User Guide

AWS CloudTrail

Version 1.0

AWS CloudTrail: User Guide

Table of Contents

What Is AWS CloudTrail?

AWS CloudTrail is an AWS service that helps you enable operational and risk auditing, governance, and compliance of your AWS account. Actions taken by a user, role, or an AWS service are recorded as events in CloudTrail. Events include actions taken in the AWS Management Console, AWS Command Line Interface, and AWS SDKs and APIs.

CloudTrail is active in your AWS account when you create it. When activity occurs in your AWS account, that activity is recorded in a CloudTrail event.

CloudTrail provides three ways to record events:

- **Event history** – The **Event history** provides a viewable, searchable, downloadable, and immutable record of the past 90 days of management events in an AWS Region. You can search events by filtering on a single attribute. You automatically have access to the **Event history** when you create your account. For more information, see Working with CloudTrail Event history.

  There are no CloudTrail charges for viewing the **Event history**.
- **CloudTrail Lake** – AWS CloudTrail Lake is a managed data lake for capturing, storing, accessing, and analyzing user and API activity on AWS for audit and security purposes. CloudTrail Lake converts existing events in row-based JSON format to Apache ORC format. ORC is a columnar storage format that is optimized for fast retrieval of data. Events are aggregated into *event data stores*, which are immutable collections of events based on criteria that you select by applying advanced event selectors. You can keep the event data in an event data store for up to 3,653 days (about 10 years) if you choose the **One-year extendable retention pricing** option, or up to 2,557 days (about 7 years) if you choose the **Seven-year retention pricing** option. You can create an event data store for a single AWS account or for multiple AWS accounts by using AWS Organizations. You can import any existing CloudTrail logs from your S3 buckets into an existing or new event data store. You can also visualize top CloudTrail event trends with Lake dashboards. For more information, see Working with AWS CloudTrail Lake.

  CloudTrail Lake event data stores and queries incur charges. When you create an event data store, you choose the pricing option you want to use for the event data store. The pricing option determines the cost for ingesting and storing events, and the default and maximum retention period for the event data store. When you run queries in Lake, you pay based upon the amount of data scanned. For information about CloudTrail pricing and managing Lake costs, see AWS CloudTrail Pricing and Managing CloudTrail Lake costs.

- **Trails** – *Trails* capture a record of AWS activities, delivering and storing these events in an Amazon S3 bucket, with optional delivery to CloudWatch Logs and Amazon EventBridge. You can input these events into your security monitoring solutions. You can also use your own third-party solutions or solutions such as Amazon Athena to search and analyze your CloudTrail logs. You can create trails for a single AWS account or for multiple AWS accounts by using AWS Organizations. You can log Insights events to analyze your management events for anomalous behavior in API call volumes and error rates. For more information, see Creating a trail for your AWS account.

  You can deliver one copy of your ongoing management events to your S3 bucket at no charge from CloudTrail by creating a trail, however, there are Amazon S3 storage charges. For more information about CloudTrail pricing, see AWS CloudTrail Pricing. For information about Amazon S3 pricing, see Amazon S3 Pricing.

Visibility into your AWS account activity is a key aspect of security and operational best practices. You can use CloudTrail to view, search, download, archive, analyze, and respond to account activity across your AWS infrastructure. You can identify who or what took which action, what resources were acted upon, when the event occurred, and other details to help you analyze and respond to activity in your AWS account.

You can integrate CloudTrail into applications using the API, automate trail or event data store creation for your organization, check the status of event data stores and trails you create, and control how users view CloudTrail events.

Accessing CloudTrail

You can work with CloudTrail in any of the following ways.

Topics

CloudTrail console

Sign in to the AWS Management Console and open the CloudTrail console at https://console.aws.amazon.com/cloudtrail/.

The CloudTrail console provides a user interface for performing many CloudTrail tasks such as:

- Viewing recent events and event history for your AWS account.
- Downloading a filtered or complete file of the last 90 days of management events from **Event history**.
- Creating and editing CloudTrail trails.
- Creating and editing CloudTrail Lake event data stores.
- Running queries on event data stores.
- Configuring CloudTrail trails, including:
  - Selecting an Amazon S3 bucket for trails.
  - Setting a prefix.
  - Configuring delivery to CloudWatch Logs.
  - Using AWS KMS keys for encryption of trail data.
  - Enabling Amazon SNS notifications for log file delivery on trails.
  - Adding and managing tags for your trails.
- Configuring CloudTrail Lake event data stores, including:
  - Integrating event data stores with CloudTrail partners or with your own applications, to log events from sources outside of AWS.
  - Federating event data stores to run queries from Amazon Athena.
  - Using AWS KMS keys for encryption of event data store data.
  - Adding and managing tags for your event data stores.

For more information about the AWS Management Console, see AWS Management Console.

AWS CLI

The AWS Command Line Interface is a unified tool that you can use to interact with CloudTrail from the command line. For more information, see the AWS Command Line Interface User Guide.

For a complete list of CloudTrail CLI commands, see cloudtrail and cloudtrail-data in the *AWS CLI Command Reference*.

CloudTrail APIs

In addition to the console and the CLI, you can also use the CloudTrail RESTful APIs to program CloudTrail directly. For more information, see the AWS CloudTrail API Reference and the CloudTrail-Data API Reference.

AWS SDKs

As an alternative to using the CloudTrail API, you can use one of the AWS SDKs. Each SDK consists of libraries and sample code for various programming languages and platforms. The SDKs provide a convenient way to create programmatic access to CloudTrail. For example, you can use the SDKs to sign requests cryptographically, manage errors, and retry requests automatically. For more information, see the Tools to Build on AWS page.

How CloudTrail works

You automatically have access to the CloudTrail **Event history** when you create your AWS account. The **Event history** provides a viewable, searchable, downloadable, and immutable record of the past 90 days of recorded management events in an AWS Region.

For an ongoing record of events in your AWS account past 90 days, create a trail or a CloudTrail Lake event data store.

Topics

- CloudTrail Event history
- CloudTrail Lake and event data stores
- CloudTrail trails
- CloudTrail Insights events
- CloudTrail channels

CloudTrail Event history

You can easily view the last 90 days of management events in the CloudTrail console by going to the **Event history** page. You can also view the event history by running the **aws cloudtrail lookup-**

events command, or the `LookupEvents` API operation. You can search events in **Event history** by filtering for events on a single attribute. For more information, see Working with CloudTrail Event history.

The **Event history** is not connected to any trails or event data stores that exist in your account and is not affected by configuration changes you make to your trails and event data stores.

There are no CloudTrail charges for viewing the **Event history** page or running the `lookup-events` command.

CloudTrail Lake and event data stores

You can create an event data store to log CloudTrail events (management events, data events), CloudTrail Insights events, AWS Audit Manager evidence, AWS Config configuration items, or events outside of AWS.

Event data stores can log events from the current AWS Region, or from all AWS Regions in your AWS account. Event data stores that you are using to log **Integration** events from outside AWS must be for a single Region only; they cannot be multi-Region event data stores.

If you have created an organization in AWS Organizations, you can create an *organization event data store* that logs all events for all AWS accounts in that organization. Organization event data stores can apply to all AWS Regions, or the current Region. Organization event data stores must be created using the management account or delegated administrator account, and when specified as applying to an organization, are automatically applied to all member accounts in the organization. Member accounts cannot see the organization event data store, nor can they modify or delete it. Organization event data stores cannot be used to collect events from outside of AWS. For more information, see Understanding organization event data stores.

By default, all events in an event data store are encrypted by CloudTrail. When you configure an event data store, you can choose to use your own AWS KMS key. Using your own KMS key incurs AWS KMS costs for encryption and decryption. After you associate an event data store with a KMS key, the KMS key cannot be removed or changed. For more information, see Encrypting CloudTrail log files with AWS KMS keys (SSE-KMS).

The following table provides information about tasks you can perform on event data stores.

Task	Description
View Lake dashboards	You can use CloudTrail Lake dashboards to visualize the events in event data stores that collect management events, S3 data events, or Insights events.
Log management events	Configure your event data store to log read-only, write-only, or all management events. By default, event data stores log management events.
Log data events	Configure your event data store to log data events. You can use advanced event selectors to filter on the `eventName` , `readOnly`, and `resources.ARN` fields to log only those events of interest.
Log Insights events	Configure your event data stores to log Insights events to help you identify and respond to unusual activity associated with management API calls. For more information, see Logging Insights events. Additional charges apply for Insights events. You will be charged separately if you enable Insights for both trails and event data stores. For more information, see AWS CloudTrail Pricing.
Copy trail events	You can copy trail events to a new or existing event data store to create a point-in-time snapshot of events logged to the trail.
Enable federation on an event data store	You can federate an event data store to see the metadata associated with the event data store in the AWS Glue Data Catalog and run SQL queries on the event data using Amazon Athena. The table metadata stored in the AWS Glue Data Catalog lets the Athena query engine know how to find, read, and process the data that you want to query.
Stop or start event ingestion on an event data store	You can stop and start event ingestion on event data stores that collect CloudTrail management and data events, or AWS Config configuration items.

Task	Description
Create an integration with an event source outside of AWS	You can use CloudTrail Lake *integrations* to log and store user activity data from outside of AWS; from any source in your hybrid environments, such as in-house or SaaS applicati ons hosted on-premises or in the cloud, virtual machines, or containers. For information about available integration partners, see AWS CloudTrail Lake Integrations.
View Lake sample queries in the CloudTrail console	The CloudTrail console provides a number of sample queries that can help you get started writing your own queries.
Create or edit a query	Queries in CloudTrail are authored in SQL. You can build a query on the CloudTrail Lake **Editor** tab by writing the query in SQL from scratch, or by opening a saved or sample query and editing it.
Save query results to an S3 bucket	When you run a query, you can save the query results to an S3 bucket.
Download saved query results	You can download a CSV file containing your saved CloudTrail Lake query results.
Validate saved query results	You can use CloudTrail query results integrity validation to determine whether the query results were modified, deleted, or unchanged after CloudTrail delivered the query results to the S3 bucket.

For more information about CloudTrail Lake, see Working with AWS CloudTrail Lake.

CloudTrail Lake event data stores and queries incur charges. When you create an event data store, you choose the pricing option you want to use for the event data store. The pricing option determines the cost for ingesting and storing events, and the default and maximum retention period for the event data store. When you run queries in Lake, you pay based upon the amount of data scanned. For information about CloudTrail pricing and managing Lake costs, see AWS CloudTrail Pricing and Managing CloudTrail Lake costs.

CloudTrail trails

A *trail* is a configuration that enables delivery of events to an Amazon S3 bucket that you specify. You can also deliver and analyze events in a trail with Amazon CloudWatch Logs and Amazon EventBridge.

Trails can log CloudTrail management events, data events, and Insights events.

You can create two types of trails for an AWS account: multi-Region trails and single-Region trails.

Multi-Region trails

When you create a multi-Region trail, CloudTrail records events in all AWS Regions in the AWS partition in which you are working and delivers the CloudTrail event log files to an S3 bucket that you specify. If an AWS Region is added after you create a multi-Region trail, that new Region is automatically included, and events in that Region are logged. Creating a multi-Region trail is a recommended best practice since you capture activity in all Regions in your account. All trails you create using the CloudTrail console are multi-Region. You can convert a single-Region trail to a multi-Region trail by using the AWS CLI. For more information, see Creating a trail in the console and Converting a trail that applies to one Region to apply to all Regions.

Single-Region trails

When you create a single-Region trail, CloudTrail records the events in that Region only. It then delivers the CloudTrail event log files to an Amazon S3 bucket that you specify. You can only create a single-Region trail by using the AWS CLI. If you create additional single trails, you can have those trails deliver CloudTrail event log files to the same S3 bucket or to separate buckets. This is the default option when you create a trail using the AWS CLI or the CloudTrail API. For more information, see Creating, updating, and managing trails with the AWS CLI.

> **Note**
>
> For both types of trails, you can specify an Amazon S3 bucket from any Region.

If you have created an organization in AWS Organizations, you can create an *organization trail* that logs all events for all AWS accounts in that organization. Organization trails can apply to all AWS Regions, or the current Region. Organization trails must be created using the management account or delegated administrator account, and when specified as applying to an organization, are automatically applied to all member accounts in the organization. Member accounts can see the organization trail, but cannot modify or delete it. By default, member accounts do not have access to the log files for an organization trail in the Amazon S3 bucket.

By default, when you create a trail in the CloudTrail console, your event log files are encrypted with a KMS key. If you choose not to enable **SSE-KMS encryption**, your event logs are encrypted using Amazon S3 server-side encryption (SSE). You can store your log files in your bucket for as long as you want. You can also define Amazon S3 lifecycle rules to archive or delete log files automatically. If you want notifications about log file delivery and validation, you can set up Amazon SNS notifications.

CloudTrail publishes log files multiple times an hour, about every 5 minutes. These log files contain API calls from services in the account that support CloudTrail. For more information, see CloudTrail supported services and integrations.

> **Note**
>
> CloudTrail typically delivers logs within an average of about 5 minutes of an API call. This time is not guaranteed. Review the AWS CloudTrail Service Level Agreement for more information.
> If you misconfigure your trail (for example, the S3 bucket is unreachable), CloudTrail will attempt to redeliver the log files to your S3 bucket for 30 days, and these attempted-to-deliver events will be subject to standard CloudTrail charges. To avoid charges on a misconfigured trail, you need to delete the trail.
> CloudTrail captures actions made directly by the user or on behalf of the user by an AWS service. For example, an AWS CloudFormation `CreateStack` call can result in additional API calls to Amazon EC2, Amazon RDS, Amazon EBS, or other services as required by the AWS CloudFormation template. This behavior is normal and expected. You can identify if the action was taken by an AWS service with the `invokedby` field in the CloudTrail event.

The following table provides information about tasks you can perform on trails.

Task	Description
Logging management events	Configure your trails to log read-only, write-only, or all management events.

Task	Description
Log data events	You can use advanced event selectors to create fine-grained selectors to log only those data events of interest. When you use advanced event selectors, you can filter on the `eventName` field to include or exclude logging of specific API calls, which can help control costs.
Log Insights events	Configure your trails to log Insights events to help you identify and respond to unusual activity associated with management API calls. Additional charges apply for Insights events. You will be charged separately if you enable Insights for both trails and event data stores. For more information, see AWS CloudTrail Pricing.
View Insights events	After you enable CloudTrail Insights on a trail, you can view up to 90 days of Insights events by using the CloudTrail console or the AWS CLI.
Download Insights events	After you enable CloudTrail Insights on a trail, you can download a CSV or JSON file containing up to the past 90 days of Insights events for your trail.
Copy trail events to CloudTrail Lake	You can copy existing trail events to a CloudTrail Lake event data store to create a point-in-time snapshot of events logged to the trail.

Task	Description
Create and subscribe to an Amazon SNS topic	Subscribe to a topic to receive notifications about log file delivery to your bucket. Amazon SNS can notify you in multiple ways, including programmatically with Amazon Simple Queue Service. **Note** If you want to receive SNS notificat ions about log file deliveries from all Regions, specify only one SNS topic for your trail. If you want to programma tically process all events, see Using the CloudTrail Processing Library.
View your log files	Find and download your log files from the S3 bucket.
Monitor events with CloudWatch Logs	You can configure your trail to send events to CloudWatch Logs. You can then use CloudWatch Logs to monitor your account for specific API calls and events. **Note** If you configure a trail that applies to all Regions to send events to a CloudWatch Logs log group, CloudTrai l sends events from all Regions to a single log group.
Enable log encryption	Log file encryption provides an extra layer of security for your log files.

Task	Description
Enable log file integrity	Log file integrity validation helps you verify that log files have remained unchanged since CloudTrail delivered them.
Share log files with other AWS accounts	You can share log files between accounts.
Aggregate logs from multiple accounts	You can aggregate log files from multiple accounts to a single bucket.
Work with partner solutions	Analyze your CloudTrail output with a partner solution that integrates with CloudTrail. Partner solutions offer a broad set of capabilit ies, such as change tracking, troubleshooting, and security analysis.

You can deliver one copy of your ongoing management events to your S3 bucket at no charge from CloudTrail by creating a trail, however, there are Amazon S3 storage charges. For more information about CloudTrail pricing, see AWS CloudTrail Pricing. For information about Amazon S3 pricing, see Amazon S3 Pricing.

CloudTrail Insights events

AWS CloudTrail Insights help AWS users identify and respond to unusual activity associated with API calls and API error rates by continuously analyzing CloudTrail management events. CloudTrail Insights analyzes your normal patterns of API call volume and API error rates, also called the *baseline*, and generates Insights events when the call volume or error rates are outside normal patterns. Insights events on API call volume are generated for `write` management APIs, and Insights events on API error rate are generated for both `read` and `write` management APIs.

By default, CloudTrail trails and event data stores don't log Insights events. You must configure your trail or event data store to log Insights events. For more information, see Logging Insights events with the AWS Management Console and Logging Insights events with the AWS Command Line Interface.

Additional charges apply for Insights events. You will be charged separately if you enable Insights for both trails and event data stores. For more information, see AWS CloudTrail Pricing.

Viewing Insights events for trails and event data stores

CloudTrail supports Insights events for both trails and event data stores, however, there are some differences in how you view and access Insights events.

Viewing Insights events for trails

If you have Insights events enabled on a trail, and CloudTrail detects unusual activity, Insights events are logged to a different folder or prefix in the destination S3 bucket for your trail. You can also see the type of insight and the incident time period when you view Insights events on the CloudTrail console. For more information, see Viewing CloudTrail Insights events for trails with the console.

After you enable CloudTrail Insights for the first time on a trail, it can take up to 36 hours for CloudTrail to deliver the first Insights event, if unusual activity is detected.

Viewing Insights events for event data stores

To log Insights events in CloudTrail Lake, you need a destination event data store that logs Insights events and a source event data store that enables Insights and logs management events. For more information, see Create an event data store for Insights events with the console.

After you enable CloudTrail Insights for the first time on the source event data store, it can take up to 7 days for CloudTrail to deliver the first Insights event to the destination event data store, if unusual activity is detected.

If you have CloudTrail Insights enabled on a source event data store and CloudTrail detects unusual activity, CloudTrail delivers Insights events to your destination event data store. You can then query your destination event data store to get information about your Insights events and can optionally save the query results to an S3 bucket. For more information, see Create or edit a query with the CloudTrail console and View sample queries with the CloudTrail console.

You can view the **Insights Events** dashboard to visualize the Insights events in your destination event data store. For more information about Lake dashboards, see View CloudTrail Lake dashboards with the CloudTrail console.

CloudTrail channels

CloudTrail supports two types of *channels*:

Channels for CloudTrail Lake integrations with event sources outside of AWS

CloudTrail Lake uses *channels* to bring events from outside of AWS into CloudTrail Lake from external partners that work with CloudTrail, or from your own sources. When you create a channel, you choose one or more event data stores to store events that arrive from the channel source. You can change the destination event data stores for a channel as needed, as long as the destination event data stores are set to log activity events. When you create a channel for events from an external partner, you provide a channel ARN to the partner or source application. The resource policy attached to the channel allows the source to transmit events through the channel. For more information, see Create an integration with an event source outside of AWS and `CreateChannel` in the *AWS CloudTrail API Reference*.

Service-linked channels

AWS services can create a service-linked channel to receive CloudTrail events on your behalf. The AWS service creating the service-linked channel configures advanced event selectors for the channel and specifies whether the channel applies to all Regions, or the current Region.

You can use the CloudTrail console or AWS CLI to view information about any CloudTrail service-linked channels created by AWS services.

CloudTrail concepts

This section summarizes basic concepts related to CloudTrail.

Concepts:

CloudTrail events

An event in CloudTrail is the record of an activity in an AWS account. This activity can be an action taken by an IAM identity, or service that is monitorable by CloudTrail. CloudTrail events provide a history of both API and non-API account activity made through the AWS Management Console, AWS SDKs, command line tools, and other AWS services.

CloudTrail log files aren't an ordered stack trace of the public API calls, so events don't appear in any specific order.

CloudTrail logs three types of events:

- Management events
- Data events
- Insights events

All event types use a CloudTrail JSON log format.

By default, trails and event data stores log management events, but not data or Insights events.

For information about how AWS services integrate with CloudTrail, see AWS service topics for CloudTrail.

Management events

Management events provide information about management operations that are performed on resources in your AWS account. These are also known as *control plane operations*.

Example management events include:

- Configuring security (for example, AWS Identity and Access Management `AttachRolePolicy` API operations).
- Registering devices (for example, Amazon EC2 `CreateDefaultVpc` API operations).
- Configuring rules for routing data (for example, Amazon EC2 `CreateSubnet` API operations).
- Setting up logging (for example, AWS CloudTrail `CreateTrail` API operations).

Management events can also include non-API events that occur in your account. For example, when a user signs in to your account, CloudTrail logs the `ConsoleLogin` event. For more information, see Non-API events captured by CloudTrail.

By default, CloudTrail trails and CloudTrail Lake event data stores log management events. For more information about logging management events, see Logging management events.

Data events

Data events provide information about the resource operations performed on or in a resource. These are also known as *data plane operations*. Data events are often high-volume activities.

Example data events include:

- Amazon S3 object-level API activity (for example, `GetObject`, `DeleteObject`, and `PutObject` API operations) on objects in S3 buckets.
- AWS Lambda function execution activity (the `Invoke` API).
- CloudTrail `PutAuditEvents` activity on a CloudTrail Lake channel that is used to log events from outside AWS.
- Amazon SNS `Publish` and `PublishBatch` API operations on topics.

The following table shows the data event types available for trails and event data stores. The **Data event type (console)** column shows the appropriate selection in the console. The **resources.type value** column shows the `resources.type` value that you would specify to include data events of that type in your trail or event data store using the AWS CLI or CloudTrail APIs.

For trails, you can use basic or advanced event selectors to log data events for Amazon S3 objects in general purpose buckets, Lambda functions, and DynamoDB tables (shown in the first three rows of the table). You can use only advanced event selectors to log the data event types shown in the remaining rows.

For event data stores, you can use only advanced event selectors to include data events.

AWS service	Description	Data event type (console)	resources.type value
Amazon DynamoDB	Amazon DynamoDB item-level API activity on tables (for example, `PutItem`, `DeleteItem` , and	**DynamoDB**	`AWS::DynamoDB::Table`

AWS service	Description	Data event type (console)	resources.type value
	`UpdateItem` API operations). **Note** For tables with streams enabled, the `resources` field in the data event contains both `AWS::DynamoDB::Stream` and `AWS::DynamoDB::Table` . If you specify `AWS::DynamoDB::Table` for the `resources.type` , it will log both DynamoDB table and DynamoDB streams events by default. To exclude streams		

AWS service	Description	Data event type (console)	resources.type value
	events, add a filter on the `eventName` field.		
AWS Lambda	AWS Lambda function execution activity (the `Invoke` API).	**Lambda**	`AWS::Lambda::Function`
Amazon S3	Amazon S3 object-level API activity (for example, `GetObject`, `DeleteObject`, and `PutObject` API operations) on objects in general purpose buckets.	**S3**	`AWS::S3::Object`
AWS AppConfig	AWS AppConfig API activity for configuration operations such as calls to `StartConfigurationSession` and `GetLatestConfiguration`.	**AWS AppConfig**	`AWS::AppConfig::Configuration`

AWS service	Description	Data event type (console)	resources.type value
AWS B2B Data Interchange	B2B Data Interchange API activity for Transformer operations such as calls to `GetTransformerJob` and `StartTransformerJob` .	**B2B Data Interchange**	`AWS::B2BI::Transformer`
Amazon Bedrock	Amazon Bedrock API activity on an agent alias.	**Bedrock agent alias**	`AWS::Bedrock::AgentAlias`
Amazon Bedrock	Amazon Bedrock API activity on a flow alias.	**Bedrock flow alias**	`AWS::Bedrock::FlowAlias`
Amazon Bedrock	Amazon Bedrock API activity on a guardrail .	**Bedrock guardrail**	`AWS::Bedrock::Guardrail`
Amazon Bedrock	Amazon Bedrock API activity on a knowledge base.	**Bedrock knowledge base**	`AWS::Bedrock::KnowledgeBase`
Amazon CloudFront	CloudFront API activity on a KeyValueStore.	**CloudFront KeyValueStore**	`AWS::CloudFront::KeyValueStore`
AWS Cloud Map	AWS Cloud Map API activity on a namespace.	**AWS Cloud Map namespace**	`AWS::ServiceDiscovery::Namespace`

AWS service	Description	Data event type (console)	resources.type value
AWS Cloud Map	AWS Cloud Map API activity on a service.	**AWS Cloud Map service**	`AWS::ServiceDiscovery::Service`
AWS CloudTrail	CloudTrail `PutAuditEvents` activity on a CloudTrail Lake channel that is used to log events from outside AWS.	**CloudTrail channel**	`AWS::CloudTrail::Channel`
Amazon CloudWatch	Amazon CloudWatch API activity on metrics.	**CloudWatch metric**	`AWS::CloudWatch::Metric`
Amazon CodeWhisperer	Amazon CodeWhisperer API activity on a customization.	**CodeWhisperer customization**	`AWS::CodeWhisperer::Customization`
Amazon CodeWhisperer	Amazon CodeWhisperer API activity on a profile.	**CodeWhisperer**	`AWS::CodeWhisperer::Profile`
Amazon Cognito	Amazon Cognito API activity on Amazon Cognito identity pools.	**Cognito Identity Pools**	`AWS::Cognito::IdentityPool`
Amazon DynamoDB	Amazon DynamoDB API activity on streams.	**DynamoDB Streams**	`AWS::DynamoDB::Stream`

AWS service	Description	Data event type (console)	resources.type value
Amazon Elastic Block Store	Amazon Elastic Block Store (EBS) direct APIs, such as `PutSnapshotBlock` , `GetSnapshotBlock` , and `ListChangedBlocks` on Amazon EBS snapshots.	**Amazon EBS direct APIs**	`AWS::EC2::Snapshot`
Amazon EMR	Amazon EMR API activity on a write-ahead log workspace.	**EMR write-ahead log workspace**	`AWS::EMRWAL::Workspace`
Amazon FinSpace	Amazon FinSpace API activity on environments.	**FinSpace**	`AWS::FinSpace::Environment`
AWS Glue	AWS Glue API activity on tables that were created by Lake Formation.	**Lake Formation**	`AWS::Glue::Table`
Amazon GuardDuty	Amazon GuardDuty API activity for a detector.	**GuardDuty detector**	`AWS::GuardDuty::Detector`
AWS HealthImaging	AWS HealthImaging API activity on data stores.	**MedicalImaging data store**	`AWS::MedicalImaging::Datastore`

AWS service	Description	Data event type (console)	resources.type value
AWS IoT	AWS IoT API activity on certificates.	**IoT certificate**	`AWS::IoT::Certificate`
AWS IoT	AWS IoT API activity on things.	**IoT thing**	`AWS::IoT::Thing`
AWS IoT Greengrass Version 2	Greengrass API activity from a Greengrass core device on a component version. **Note** Greengrass doesn't log access denied events.	**IoT Greengrass component version**	`AWS::GreengrassV2::ComponentVersion`
AWS IoT Greengrass Version 2	Greengrass API activity from a Greengrass core device on a deployment. **Note** Greengrass doesn't log access denied events.	**IoT Greengrass deployment**	`AWS::GreengrassV2::Deployment`

AWS service	Description	Data event type (console)	resources.type value
AWS IoT SiteWise	IoT SiteWise API activity on assets.	**IoT SiteWise asset**	`AWS::IoTSiteWise::Asset`
AWS IoT SiteWise	IoT SiteWise API activity on time series.	**IoT SiteWise time series**	`AWS::IoTSiteWise::TimeSeries`
AWS IoT TwinMaker	IoT TwinMaker API activity on an entity.	**IoT TwinMaker entity**	`AWS::IoTTwinMaker::Entity`
AWS IoT TwinMaker	IoT TwinMaker API activity on a workspace.	**IoT TwinMaker workspace**	`AWS::IoTTwinMaker::Workspace`
Amazon Kendra Intelligent Ranking	Amazon Kendra Intelligent Ranking API activity on rescore execution plans.	**Kendra Ranking**	`AWS::KendraRanking::ExecutionPlan`
Amazon Keyspaces (for Apache Cassandra)	Amazon Keyspaces API activity on a table.	**Cassandra table**	`AWS::Cassandra::Table`
Amazon Kinesis Data Streams	Kinesis Data Streams API activity on streams.	**Kinesis stream**	`AWS::Kinesis::Stream`
Amazon Kinesis Data Streams	Kinesis Data Streams API activity on stream consumers.	**Kinesis stream consumer**	`AWS::Kinesis::StreamConsumer`

AWS service	Description	Data event type (console)	resources.type value
Amazon Kinesis Video Streams	Kinesis Video Streams API activity on video streams, such as calls to `GetMedia` and `PutMedia`.	**Kinesis video stream**	`AWS::KinesisVideo::Stream`
Amazon Machine Learning	Machine Learning API activity on ML models.	**Maching Learning MlModel**	`AWS::MachineLearning::MlModel`
Amazon Managed Blockchain	Amazon Managed Blockchain API activity on a network.	**Managed Blockchain network**	`AWS::ManagedBlockchain::Network`
Amazon Managed Blockchain	Amazon Managed Blockchain JSON-RPC calls on Ethereum nodes, such as `eth_getBalance` or `eth_getBlockByNumber` .	**Managed Blockchain**	`AWS::ManagedBlockchain::Node`
Amazon Neptune Graph	Data API activities, for example queries, algorithms, or vector search, on a Neptune Graph.	**Neptune Graph**	`AWS::NeptuneGraph::Graph`
AWS Payment Cryptography	AWS Payment Cryptography API activity on aliases.	**Payment Cryptography Alias**	`AWS::PaymentCryptography::Alias`

AWS service	Description	Data event type (console)	resources.type value
AWS Payment Cryptography	AWS Payment Cryptography API activity on keys.	**Payment Cryptography Key**	`AWS::PaymentCryptography::Key`
AWS Private CA	AWS Private CA Connector for Active Directory API activity.	**AWS Private CA Connector for Active Directory**	`AWS::PCAConnectorAD::Connector`
AWS Private CA	AWS Private CA Connector for SCEP API activity.	**AWS Private CA Connector for SCEP**	`AWS::PCAConnectorSCEP::Connector`
Amazon Q Apps	Data API activity on Amazon Q Apps.	**Amazon Q Apps**	`AWS::QApps:QApp`
Amazon Q Business	Amazon Q Business API activity on an application.	**Amazon Q Business application**	`AWS::QBusiness::Application`
Amazon Q Business	Amazon Q Business API activity on a data source.	**Amazon Q Business data source**	`AWS::QBusiness::DataSource`
Amazon Q Business	Amazon Q Business API activity on an index.	**Amazon Q Business index**	`AWS::QBusiness::Index`

AWS service	Description	Data event type (console)	resources.type value
Amazon Q Business	Amazon Q Business API activity on a web experience.	**Amazon Q Business web experience**	`AWS::QBusiness::WebExperience`
Amazon RDS	Amazon RDS API activity on a DB Cluster.	**RDS Data API - DB Cluster**	`AWS::RDS::DBCluster`
Amazon S3	Amazon S3 API activity on access points.	**S3 Access Point**	`AWS::S3::AccessPoint`
Amazon S3	Amazon S3 object-level API activity (for example, `GetObject` , `DeleteObject` , and `PutObject` API operations) on objects in directory buckets.	**S3 Express**	`AWS::S3Express::Object`
Amazon S3	Amazon S3 Object Lambda access points API activity, such as calls to `CompleteMultipartUpload` and `GetObject` .	**S3 Object Lambda**	`AWS::S3ObjectLambda::AccessPoint`
Amazon S3 on Outposts	Amazon S3 on Outposts object-level API activity.	**S3 Outposts**	`AWS::S3Outposts::Object`

AWS service	Description	Data event type (console)	resources.type value
Amazon SageMaker	Amazon SageMaker InvokeEndpointWithResponseStream activity on endpoints.	**SageMaker endpoint**	AWS::SageMaker::Endpoint
Amazon SageMaker	Amazon SageMaker API activity on feature stores.	**SageMaker feature store**	AWS::SageMaker::FeatureGroup
Amazon SageMaker	Amazon SageMaker API activity on experiment trial components.	**SageMaker metrics experiment trial component**	AWS::SageMaker::ExperimentTrialComponent
Amazon SNS	Amazon SNS Publish API operations on platform endpoints.	**SNS platform endpoint**	AWS::SNS::PlatformEndpoint
Amazon SNS	Amazon SNS Publish and PublishBatch API operations on topics.	**SNS topic**	AWS::SNS::Topic
Amazon SQS	Amazon SQS API activity on messages.	**SQS**	AWS::SQS::Queue
AWS Step Functions	Step Functions API activity on a state machine.	**Step Functions state machine**	AWS::StepFunctions::StateMachine

AWS service	Description	Data event type (console)	resources.type value
AWS Supply Chain	AWS Supply Chain API activity on an instance.	**Supply Chain**	`AWS::SCN::Instance`
Amazon SWF	Amazon SWF API activity on domains.	**SWF domain**	`AWS::SWF::Domain`
AWS Systems Manager	Systems Manager API activity on control channels.	**Systems Manager**	`AWS::SSMMessages::ControlChannel`
AWS Systems Manager	Systems Manager API activity on managed nodes.	**Systems Manager managed node**	`AWS::SSM::ManagedNode`
Amazon Timestream	Amazon Timestream `Query` API activity on databases.	**Timestream database**	`AWS::Timestream::Database`
Amazon Timestream	Amazon Timestream `Query` API activity on tables.	**Timestream table**	`AWS::Timestream::Table`
Amazon Verified Permissions	Amazon Verified Permissions API activity on a policy store.	**Amazon Verified Permissions**	`AWS::VerifiedPermissions::PolicyStore`
Amazon WorkSpaces Thin Client	WorkSpaces Thin Client API activity on a Device.	**Thin Client Device**	`AWS::ThinClient::Device`

AWS service	Description	Data event type (console)	resources.type value
Amazon WorkSpaces Thin Client	WorkSpaces Thin Client API activity on an Environment.	**Thin Client Environment**	`AWS::ThinClient::Environment`
AWS X-Ray	X-Ray API activity on traces.	**X-Ray trace**	`AWS::XRay::Trace`

Data events are not logged by default when you create a trail or event data store. To record CloudTrail data events, you must explicitly add the supported resources or resource types for which you want to collect activity. For more information about logging data events, see Logging data events.

Additional charges apply for logging data events. For CloudTrail pricing, see AWS CloudTrail Pricing.

Insights events

CloudTrail Insights events capture unusual API call rate or error rate activity in your AWS account by analyzing CloudTrail management activity. Insights events provide relevant information, such as the associated API, error code, incident time, and statistics, that help you understand and act on unusual activity. Unlike other types of events captured in a CloudTrail trail or event data store, Insights events are logged only when CloudTrail detects changes in your account's API usage or error rate logging that differ significantly from the account's typical usage patterns.

Examples of activity that might generate Insights events include:

- Your account typically logs no more than 20 Amazon S3 `deleteBucket` API calls per minute, but your account starts to log an average of 100 `deleteBucket` API calls per minute. An Insights event is logged at the start of the unusual activity, and another Insights event is logged to mark the end of the unusual activity.
- Your account typically logs 20 calls per minute to the Amazon EC2 `AuthorizeSecurityGroupIngress` API, but your account starts to log zero calls to `AuthorizeSecurityGroupIngress`. An Insights event is logged at the start of the unusual

activity, and ten minutes later, when the unusual activity ends, another Insights event is logged to mark the end of the unusual activity.

- Your account typically logs less than one `AccessDeniedException` error in a seven-day period on the AWS Identity and Access Management API, `DeleteInstanceProfile`. Your account starts to log an average of 12 `AccessDeniedException` errors per minute on the `DeleteInstanceProfile` API call. An Insights event is logged at the start of the unusual error rate activity, and another Insights event is logged to mark the end of the unusual activity.

These examples are provided for illustration purposes only. Your results may vary depending on your use case.

To log CloudTrail Insights events, you must explicitly enable Insights events on a new or existing trail or event data store. For more information about creating a trail, see Creating a trail with the CloudTrail console. For more information about creating an event data store, see Create an event data store for Insights events with the console.

Additional charges apply for Insights events. You will be charged separately if you enable Insights for both trails and event data stores. For more information, see AWS CloudTrail Pricing.

Event history

CloudTrail event history provides a viewable, searchable, downloadable, and immutable record of the past 90 days of CloudTrail management events in an AWS Region. You can use this history to gain visibility into actions taken in your AWS account in the AWS Management Console, AWS SDKs, command line tools, and other AWS services. You can customize your view of event history in the CloudTrail console by selecting which columns are displayed. For more information, see Working with CloudTrail Event history.

Trails

A trail is a configuration that enables delivery of CloudTrail events to an S3 bucket, with optional delivery to CloudWatch Logs and Amazon EventBridge. You can use a trail to choose the CloudTrail events you want delivered, encrypt your CloudTrail event log files with an AWS KMS key, and set up Amazon SNS notifications for log file delivery. For more information about how to create and manage a trail, see Creating a trail for your AWS account.

Multi-Region and single-Region trails

You can create two types of trails for an AWS account: multi-Region trails and single-Region trails.

Multi-Region trails

When you create a multi-Region trail, CloudTrail records events in all AWS Regions in the AWS partition in which you are working and delivers the CloudTrail event log files to an S3 bucket that you specify. If an AWS Region is added after you create a multi-Region trail, that new Region is automatically included, and events in that Region are logged. Creating a multi-Region trail is a recommended best practice since you capture activity in all Regions in your account. All trails you create using the CloudTrail console are multi-Region. You can convert a single-Region trail to a multi-Region trail by using the AWS CLI. For more information, see Creating a trail in the console and Converting a trail that applies to one Region to apply to all Regions.

Single-Region trails

When you create a single-Region trail, CloudTrail records the events in that Region only. It then delivers the CloudTrail event log files to an Amazon S3 bucket that you specify. You can only create a single-Region trail by using the AWS CLI. If you create additional single trails, you can have those trails deliver CloudTrail event log files to the same S3 bucket or to separate buckets. This is the default option when you create a trail using the AWS CLI or the CloudTrail API. For more information, see Creating, updating, and managing trails with the AWS CLI.

Note

For both types of trails, you can specify an Amazon S3 bucket from any Region.

A multi-Region trail has the following advantages:

- The configuration settings for the trail apply consistently across all AWS Regions.
- You receive CloudTrail events from all AWS Regions in a single Amazon S3 bucket and, optionally, in a CloudWatch Logs log group.
- You manage trail configuration for all AWS Regions from one location.

When you apply a trail to all AWS Regions, CloudTrail uses the trail that you create in a particular Region to create trails with identical configurations in all other Regions in the AWS partition in which you are working.

This has the following effects:

- CloudTrail delivers log files for account activity from all AWS Regions to the single Amazon S3 bucket that you specify, and, optionally, to a CloudWatch Logs log group.
- If you configured an Amazon SNS topic for the trail, SNS notifications about log file deliveries in all AWS Regions are sent to that single SNS topic.

Regardless of whether a trail is multi-Region or single-Region, events sent to Amazon EventBridge are received in each Region's event bus, rather than in one single event bus.

Multiple trails per Region

If you have different but related user groups, such as developers, security personnel, and IT auditors, you can create multiple trails per Region. This allows each group to receive its own copy of the log files.

CloudTrail supports five trails per Region. A multi-Region trail counts as one trail per Region.

The following is an example of a Region with five trails:

- You create two trails in the US West (N. California) Region that apply to this Region only.
- You create two more multi-Region trails in US West (N. California) Region.
- You create another multi-Region trail in the Asia Pacific (Sydney) Region. This trail also exists as a trail in the US West (N. California) Region.

You can view a list of trails in an AWS Region in the **Trails** page of the CloudTrail console. For more information, see Updating a trail with the CloudTrail console. For CloudTrail pricing, see AWS CloudTrail Pricing.

Organization trails

An organization trail is a configuration that enables delivery of CloudTrail events in the management account and all member accounts in an AWS Organizations organization to the same Amazon S3 bucket, CloudWatch Logs, and Amazon EventBridge. Creating an organization trail helps you define a uniform event logging strategy for your organization.

All organization trails created using the console are multi-Region organization trails that log events from the enabled AWS Regions in each member account in the organization. To log events in all AWS partitions in your organization, create a multi-Region organization trail in each partition. You can create either a single-Region or multi-Region organization trail by using the AWS CLI. If you create a single-Region trail, you log activity only in the trail's AWS Region (also referred to as the *Home* Region).

Although most AWS Regions are enabled by default for your AWS account, you must manually enable certain Regions (also referred to as *opt-in Regions*). For information about which Regions are enabled by default, see Considerations before enabling and disabling Regions in the *AWS Account Management Reference Guide*. For the list of Regions CloudTrail supports, see CloudTrail supported Regions.

When you create an organization trail, a copy of the trail with the name that you give it is created in the member accounts that belongs to your organization.

- If the organization trail is for a **single-Region** and the trail's home Region **is not an opt-Region**, a copy of the trail is created in the organization trail's home Region in each member account.
- If the organization trail is for a **single-Region** and the trail's home Region **is an opt-Region**, a copy of the trail is created in the organization trail's home Region in the member accounts that have enabled that Region.
- If the organization trail is **multi-Region** and the trail's home Region **is not an opt-in Region**, a copy of the trail is created in each enabled AWS Region in each member account. When a member account enables an opt-in Region, a copy of the multi-Region trail is created in the newly opted in Region for the member account after activation of that Region is complete.
- If the organization trail is **multi-Region** and the home Region **is an opt-in Region**, member accounts will not send activity to the organization trail unless they opt into the AWS Region where the multi-Region trail was created. For example, if you create a multi-Region trail and choose the Europe (Spain) Region as the home Region for the trail, only member accounts that enabled the Europe (Spain) Region for their account will send their account activity to the organization trail.

Note

CloudTrail creates organization trails in member accounts even if a resource validation fails. Examples of validation failures include:

- an incorrect Amazon S3 bucket policy
- an incorrect Amazon SNS topic policy
- inability to deliver to a CloudWatch Logs log group
- insufficient permission to encrypt using a KMS key

A member account with CloudTrail permissions can see any validation failures for an organization trail by viewing the trail's details page on the CloudTrail console, or by running the AWS CLI get-trail-status command.

Users with CloudTrail permissions in member accounts will be able to see organization trails (including the trail ARN) when they log into the AWS CloudTrail console from their AWS accounts, or when they run AWS CLI commands such as `describe-trails` (although member accounts must use the ARN for the organization trail, and not the name, when using the AWS CLI). However, users in member accounts will not have sufficient permissions to delete organization trails, turn logging on or off, change what types of events are logged, or otherwise alter organization trails in any way. For more information about AWS Organizations, see Organizations Terminology and Concepts. For more information about creating and working with organization trails, see Creating a trail for an organization.

CloudTrail Lake and event data stores

CloudTrail Lake lets you run fine-grained SQL-based queries on your events, and log events from sources outside AWS, including from your own applications, and from partners who are integrated with CloudTrail. You do not need to have a trail configured in your account to use CloudTrail Lake.

Events are aggregated into event data stores, which are immutable collections of events based on criteria that you select by applying advanced event selectors. You can keep the event data in an event data store for up to 3,653 days (about 10 years) if you choose the **One-year extendable retention pricing** option, or up to 2,557 days (about 7 years) if you choose the **Seven-year retention pricing** option. You can save Lake queries for future use, and view results of queries for up to seven days. You can also save query results to an S3 bucket. CloudTrail Lake can also store events from an organization in AWS Organizations in an event data store, or events from multiple Regions and accounts. CloudTrail Lake is part of an auditing solution that helps you perform security investigations and troubleshooting. For more information, see Working with AWS CloudTrail Lake and CloudTrail Lake concepts and terminology.

CloudTrail Insights

CloudTrail Insights help AWS users identify and respond to unusual volumes of API calls or errors logged on API calls by continuously analyzing CloudTrail management events. An Insights event is a record of unusual levels of `write` management API activity, or unusual levels of errors returned on management API activity. By default, trails and event data stores don't log CloudTrail Insights events. In the console, you can choose to log Insights events when you create or update a trail or event data store. When you use the CloudTrail API, you can log Insights events by editing the settings of an existing trail or event data store with the `PutInsightSelectors` API. Additional charges apply for logging CloudTrail Insights events. You will be charged separately if you enable Insights for both trails and event data stores. For more information, see Logging Insights events and AWS CloudTrail Pricing.

Tags

A tag is a customer-defined key and optional value that can be assigned to AWS resources, such as CloudTrail trails, event data stores, and channels, S3 buckets used to store CloudTrail log files, AWS Organizations organizations and organizational units, and many more. By adding the same tags to trails and to the S3 buckets you use to store log files for trails, you can make it easier to manage, search for, and filter these resources with AWS Resource Groups. You can implement tagging strategies to help you consistently, effectively, and easily find and manage your resources. For more information, see Best Practices for Tagging AWS Resources.

AWS Security Token Service and CloudTrail

AWS Security Token Service (AWS STS) is a service that has a global endpoint and also supports Region-specific endpoints. An endpoint is a URL that is the entry point for web service requests. For example, `https://cloudtrail.us-west-2.amazonaws.com` is the US West (Oregon) regional entry point for the AWS CloudTrail service. Regional endpoints help reduce latency in your applications.

When you use an AWS STS Region-specific endpoint, the trail in that Region delivers only the AWS STS events that occur in that Region. For example, if you are using the endpoint `sts.us-west-2.amazonaws.com`, the trail in us-west-2 delivers only the AWS STS events that originate from us-west-2. For more information about AWS STS regional endpoints, see Activating and Deactivating AWS STS in an AWS Region in the *IAM User Guide*.

For a complete list of AWS regional endpoints, see AWS Regions and Endpoints in the *AWS General Reference*. For details about events from the global AWS STS endpoint, see Global service events.

Global service events

⚠ **Important**

As of November 22, 2021, AWS CloudTrail changed how trails capture global service events. Now, events created by Amazon CloudFront, AWS Identity and Access Management, and AWS STS are recorded in the Region in which they were created, the US East (N. Virginia) Region, us-east-1. This makes how CloudTrail treats these services consistent with that of other AWS global services. To continue receiving global service events outside of US East (N. Virginia), be sure to convert *single-Region trails* using global service events outside of US East (N. Virginia) into *multi-Region trails*. For more information about capturing global service events, see Enabling and disabling global service event logging later in this section. In contrast, the **Event history** in the CloudTrail console and the **aws cloudtrail lookup-events** command will show these events in the AWS Region where they occurred.

For most services, events are recorded in the Region where the action occurred. For global services such as AWS Identity and Access Management (IAM), AWS STS, and Amazon CloudFront, events are delivered to any trail that includes global services.

For most global services, events are logged as occurring in US East (N. Virginia) Region, but some global service events are logged as occurring in other Regions, such as US East (Ohio) Region or US West (Oregon) Region.

To avoid receiving duplicate global service events, remember the following:

- Global service events are delivered by default to trails that are created using the CloudTrail console. Events are delivered to the bucket for the trail.
- If you have multiple single Region trails, consider configuring your trails so that global service events are delivered in only one of the trails. For more information, see Enabling and disabling global service event logging.
- If you change the configuration of a trail from logging all Regions to logging a single Region, global service event logging is turned off automatically for that trail. Similarly, if you change the configuration of a trail from logging a single Region to logging all Regions, global service event logging is turned on automatically for that trail.

  For more information about changing global service event logging for a trail, see Enabling and disabling global service event logging.

Example:

1. You create a trail in the CloudTrail console. By default, this trail logs global service events.
2. You have multiple single Region trails.
3. You do not need to include global services for the single Region trails. Global service events are delivered for the first trail. For more information, see Creating, updating, and managing trails with the AWS CLI.

Note

When you create or update a trail with the AWS CLI, AWS SDKs, or CloudTrail API, you can specify whether to include or exclude global service events for trails. You cannot configure global service event logging from the CloudTrail console.

CloudTrail supported Regions

Note

For information about Regions supported by CloudTrail Lake, see CloudTrail Lake supported Regions.
For information about data plane endpoints, see Data plane endpoints in the *AWS General Reference*.

Region name	Region	Control plane endpoint	Protocol	Support date
US East (N. Virginia)	us-east-1	cloudtrail.us-east-1.amazonaws.com	HTTPS	11/13/2013
US East (Ohio)	us-east-2	cloudtrail.us-east-2.amazonaws.com	HTTPS	10/17/2016
US West (N. California)	us-west-1	cloudtrail.us-west-1.amazonaws.com	HTTPS	05/13/2014

Region name	Region	Control plane endpoint	Protocol	Support date
US West (Oregon)	us-west-2	cloudtrail.us-west-2.amazonaws.com	HTTPS	11/13/2013
Africa (Cape Town)	af-south-1	cloudtrail.af-south-1.amazonaws.com	HTTPS	04/22/2020
Asia Pacific (Hong Kong)	ap-east-1	cloudtrail.ap-east-1.amazonaws.com	HTTPS	04/24/2019
Asia Pacific (Hyderabad)	ap-south-2	cloudtrail.ap-south-2.amazonaws.com	HTTPS	11/22/2022
Asia Pacific (Jakarta)	ap-southeast-3	cloudtrail.ap-southeast-3.amazonaws.com	HTTPS	12/13/2021
Asia Pacific (Melbourne)	ap-southeast-4	cloudtrail.ap-southeast-4.amazonaws.com	HTTPS	01/23/2023
Asia Pacific (Mumbai)	ap-south-1	cloudtrail.ap-south-1.amazonaws.com	HTTPS	06/27/2016
Asia Pacific (Osaka)	ap-northeast-3	cloudtrail.ap-northeast-3.amazonaws.com	HTTPS	02/12/2018
Asia Pacific (Seoul)	ap-northeast-2	cloudtrail.ap-northeast-2.amazonaws.com	HTTPS	01/06/2016
Asia Pacific (Singapore)	ap-southeast-1	cloudtrail.ap-southeast-1.amazonaws.com	HTTPS	06/30/2014
Asia Pacific (Sydney)	ap-southeast-2	cloudtrail.ap-southeast-2.amazonaws.com	HTTPS	05/13/2014
Asia Pacific (Tokyo)	ap-northeast-1	cloudtrail.ap-northeast-1.amazonaws.com	HTTPS	06/30/2014

Region name	Region	Control plane endpoint	Protocol	Support date
Canada (Central)	ca-central-1	cloudtrail.ca-central-1.amazonaws.com	HTTPS	12/08/2016
Canada West (Calgary)	ca-west-1	cloudtrail.ca-west-1.amazonaws.com	HTTPS	12/20/2023
China (Beijing)	cn-north-1	cloudtrail.cn-north-1.amazonaws.com.cn	HTTPS	03/01/2014
China (Ningxia)	cn-northwest-1	cloudtrail.cn-northwest-1.amazonaws.com.cn	HTTPS	12/11/2017
Europe (Frankfurt)	eu-central-1	cloudtrail.eu-central-1.amazonaws.com	HTTPS	10/23/2014
Europe (Ireland)	eu-west-1	cloudtrail.eu-west-1.amazonaws.com	HTTPS	05/13/2014
Europe (London)	eu-west-2	cloudtrail.eu-west-2.amazonaws.com	HTTPS	12/13/2016
Europe (Milan)	eu-south-1	cloudtrail.eu-south-1.amazonaws.com	HTTPS	04/27/2020
Europe (Paris)	eu-west-3	cloudtrail.eu-west-3.amazonaws.com	HTTPS	12/18/2017
Europe (Spain)	eu-south-2	cloudtrail.eu-south-2.amazonaws.com	HTTPS	11/16/2022
Europe (Stockholm)	eu-north-1	cloudtrail.eu-north-1.amazonaws.com	HTTPS	12/11/2018
Europe (Zurich)	eu-central-2	cloudtrail.eu-central-2.amazonaws.com	HTTPS	11/09/2022

Region name	Region	Control plane endpoint	Protocol	Support date
Israel (Tel Aviv)	il-central-1	cloudtrail.il-central-1.amazonaws.com	HTTPS	07/31/2023
Middle East (Bahrain)	me-south-1	cloudtrail.me-south-1.amazonaws.com	HTTPS	07/29/2019
Middle East (UAE)	me-central-1	cloudtrail.me-central-1.amazonaws.com	HTTPS	08/30/2022
South America (São Paulo)	sa-east-1	cloudtrail.sa-east-1.amazonaws.com	HTTPS	06/30/2014
AWS GovCloud (US-East)	us-gov-east-1	cloudtrail.us-gov-east-1.amazonaws.com	HTTPS	11/12/2018
AWS GovCloud (US-West)	us-gov-west-1	cloudtrail.us-gov-west-1.amazonaws.com	HTTPS	08/16/2011

For more information about using CloudTrail in the AWS GovCloud (US) Regions, see Service Endpoints in the *AWS GovCloud (US) User Guide*.

For more information about using CloudTrail in the China (Beijing) Region, see Endpoints and ARNs for AWS in China in the *Amazon Web Services General Reference*.

CloudTrail supported services and integrations

CloudTrail supports logging events for many AWS services. You can find the specifics for each supported service in that service's guide. For a list of service-specific topics, see AWS service topics for CloudTrail. In addition, some AWS services can be used to analyze and act upon data collected in CloudTrail logs.

Note

To see the list of supported Regions for each service, see Service endpoints and quotas in the *Amazon Web Services General Reference*.

Topics

- AWS service integrations with CloudTrail logs
- CloudTrail integration with Amazon EventBridge
- CloudTrail integration with AWS Organizations
- AWS service topics for CloudTrail
- CloudTrail unsupported services

AWS service integrations with CloudTrail logs

Note

You can also use CloudTrail Lake to query and analyze your events. CloudTrail Lake queries offer a deeper and more customizable view of events than simple key and value lookups in **Event history**, or running `LookupEvents`. CloudTrail Lake users can run complex Standard Query Language (SQL) queries across multiple fields in a CloudTrail event. For more information, see Working with AWS CloudTrail Lake and Copying trail events to CloudTrail Lake.
CloudTrail Lake event data stores and queries incur CloudTrail charges. For more information about CloudTrail Lake pricing, see AWS CloudTrail Pricing.

You can configure other AWS services to further analyze and act upon the event data collected in CloudTrail logs. For more information, see the following topics.

AWS Service	Topic	Description
Amazon Athena	Querying AWS CloudTrail Logs	Using Athena with CloudTrai l logs is a powerful way to enhance your analysis of

AWS Service	Topic	Description
		AWS service activity. For example, you can use queries to identify trends and further isolate activity by attribute, such as source IP address or user. You can automatically create tables for querying logs directly from the CloudTrail console, and use those tables to run queries in Athena. For more information, see Creating a Table for CloudTrail Logs in the CloudTrail Console in the Amazon Athena User Guide. **Note** Running queries in Amazon Athena incurs additional costs. For more information, see Amazon Athena Pricing.

AWS Service	Topic	Description
Amazon CloudWatch Logs	Monitoring CloudTrail Log Files with Amazon CloudWatch Logs	You can configure CloudTrail with CloudWatch Logs to monitor your trail logs and be notified when specific activity occurs. For example, you can define CloudWatch Logs metric filters that will trigger CloudWatch alarms and send notifications to you when those alarms are triggered. **Note** Standard pricing for Amazon CloudWatch and Amazon CloudWatch Logs applies. For more information, see Amazon CloudWatch Pricing.

CloudTrail integration with Amazon EventBridge

Amazon EventBridge is an AWS service that delivers a near real-time stream of system events that describe changes in AWS resources. In EventBridge, you can create rules that responds to events recorded by CloudTrail. For more information, see Create a rule in Amazon EventBridge.

You can deliver events that you are subscribed to on your trail to EventBridge by creating a rule with the EventBridge console.

From the EventBridge console:

- Choose the `AWS API Call via CloudTrail` detail-type to deliver CloudTrail data and management events with an `eventType` of `AwsApiCall`. To record events with a detail-type value of AWS API Call via CloudTrail, you must have a trail that is currently logging management or data events.

- Choose the AWS Console Sign In via CloudTrail detail-type to deliver AWS Management Console sign-in events. To record events with a detail-type of AWS Console Sign In via CloudTrail, you must have a trail that is currently logging management events.
- Choose the AWS Insight via CloudTrail detail-type to deliver Insights events. To record events with a detail-type value of AWS Insight via CloudTrail, you must have a trail that is currently logging Insights events. For information about logging Insights events, see Logging Insights events.

For more information about how to create a trail, see Creating a trail with the CloudTrail console.

CloudTrail integration with AWS Organizations

The management account for an AWS Organizations organization can add a delegated administrator to manage the organization's CloudTrail resources. You can create an organization trail or organization event data store in the management account or delegated administrator account for an organization that collects all event data for all AWS accounts in an organization in AWS Organizations. Creating an organization trail helps you define a uniform event logging strategy for your organization.

An organization trail is applied automatically to each AWS account in your organization. Users in member accounts can see these trails but cannot modify them, and by default cannot see the log files created for the organization trail. For more information, see Creating a trail for an organization.

AWS service topics for CloudTrail

You can learn more about how the events for individual AWS services are recorded in CloudTrail logs, including example events for that service in log files. For more information about how specific AWS services integrate with CloudTrail, see the topic about integration in the individual guide for that service.

Services that are still in preview, or not yet released for general availability (GA), or which don't have public APIs, are not considered supported. CloudTrail does not currently log Amazon VPC endpoint policy-specific events.

Note

To see the list of supported Regions for each service, see Service endpoints and quotas in the *Amazon Web Services General Reference*.
For information about which services log data events, see Data events.

AWS Service	CloudTrail Topics	Support began
Amazon API Gateway	Log API management calls to Amazon API Gateway Using AWS CloudTrail	07/09/2015
Amazon AppFlow	Logging Amazon AppFlow API calls with AWS CloudTrail	04/22/2020
Amazon AppStream 2.0	Logging Amazon AppStream 2.0 API Calls with AWS CloudTrail	04/25/2019
Amazon Athena	Logging Amazon Athena API Calls with AWS CloudTrail	05/19/2017
Amazon Aurora	Monitoring Amazon Aurora API calls in AWS CloudTrail	08/31/2018
Amazon Bedrock	Log Amazon Bedrock API calls using AWS CloudTrail	10/23/2023
Amazon Braket	Amazon Braket API logging with CloudTrail	08/12/2020
Amazon Chime	Log Amazon Chime Administration Calls Using AWS CloudTrail	09/27/2017
Amazon Cloud Directory	Logging Cloud Directory API Calls Using AWS CloudTrail	01/26/2017

AWS Service	CloudTrail Topics	Support began
Amazon CloudFront	Using AWS CloudTrail to Capture Requests Sent to the CloudFront API	05/28/2014
Amazon CloudSearch	Logging Amazon CloudSearch Configuration Service Calls Using AWS CloudTrail	10/16/2014
Amazon CloudWatch	Logging Amazon CloudWatch API Calls in AWS CloudTrail	04/30/2014
Amazon CloudWatch Logs	Logging Amazon CloudWatch Logs API Calls in AWS CloudTrail	03/10/2016
Amazon CodeCatalyst	Logging CodeCatalyst API calls in connected AWS accounts using AWS CloudTrail	12/01/2022
Amazon CodeGuru Reviewer	Logging Amazon CodeGuru Reviewer API Calls with AWS CloudTrail	12/02/2019
Amazon Cognito	Logging Amazon Cognito API Calls with AWS CloudTrail	02/18/2016
Amazon Comprehend	Logging Amazon Comprehend API Calls with AWS CloudTrail	01/17/2018
Amazon Comprehend Medical	Logging Amazon Comprehend Medical API Calls by Using AWS CloudTrail	11/27/2018
Amazon Connect	Logging Amazon Connect API Calls with AWS CloudTrail	12/11/2019

AWS Service	CloudTrail Topics	Support began
Amazon Data Firehose	Monitoring Amazon Data Firehose API Calls with AWS CloudTrail	03/17/2016
Amazon Data Lifecycle Manager	Logging Amazon Data Lifecycle Manager API Calls Using AWS CloudTrail	07/24/2018
Amazon Detective	Logging Amazon Detective API calls with AWS CloudTrail	03/31/2020
Amazon DevOps Guru	Logging Amazon DevOps Guru API calls with AWS CloudTrail	05/04/2021
Amazon DocumentDB (with MongoDB compatibility)	Logging Amazon DocumentDB API Calls with AWS CloudTrail	01/09/2019
Amazon DynamoDB	Logging DynamoDB Operations By Using AWS CloudTrail	05/28/2015
Amazon EC2	Log Amazon EC2 API calls using AWS CloudTrail	11/13/2013
Amazon EC2 Auto Scaling	Logging Auto Scaling API Calls By Using CloudTrail	07/16/2014
Amazon EC2 Capacity Blocks	Logging Capacity Blocks API calls with AWS CloudTrail	10/31/2023
Amazon EC2 Image Builder	Logging EC2 Image Builder API calls using CloudTrail	12/02/2019

AWS Service	CloudTrail Topics	Support began
Amazon Elastic Block Store (Amazon EBS) EBS direct APIs	Logging API Calls Using AWS CloudTrail Log API Calls for the EBS direct APIs with AWS CloudTrail	Amazon EBS: 11/13/2013 EBS direct APIs: 06/30/2020
Amazon Elastic Container Registry (Amazon ECR)	Logging Amazon ECR API Calls By Using AWS CloudTrail	12/21/2015
Amazon Elastic Container Service (Amazon ECS)	Logging Amazon ECS API Calls By Using AWS CloudTrail	04/09/2015
Amazon Elastic File System (Amazon EFS)	Logging Amazon EFS API Calls with AWS CloudTrail	06/28/2016
Amazon Elastic Kubernetes Service (Amazon EKS)	Logging Amazon EKS API Calls with AWS CloudTrail	06/05/2018
Amazon Elastic Transcoder	Logging Amazon Elastic Transcoder API Calls with AWS CloudTrail	10/27/2014
Amazon ElastiCache	Logging Amazon ElastiCache API Calls Using AWS CloudTrail	09/15/2014
Amazon EMR	Logging Amazon EMR API Calls in AWS CloudTrail	04/04/2014
Amazon EMR on EKS	Logging Amazon EMR on EKS API calls using AWS CloudTrail	12/09/2020
Amazon EventBridge	Logging Amazon EventBridge API calls using AWS CloudTrail	07/11/2019
Amazon FinSpace	Querying AWS CloudTrail logs	10/18/2022

AWS Service	CloudTrail Topics	Support began
Amazon Forecast	Logging Amazon Forecast API Calls with AWS CloudTrail	11/28/2018
Amazon Fraud Detector	Logging Amazon Fraud Detector API Calls with AWS CloudTrail	01/09/2020
Amazon FSx for Lustre	Logging Amazon FSx for Lustre API Calls with AWS CloudTrail	01/11/2019
Amazon FSx for Windows File Server	Monitoring with AWS CloudTrail	11/28/2018
Amazon GameLift	Logging Amazon GameLift API Calls with AWS CloudTrail	01/27/2016
Amazon GuardDuty	Logging Amazon GuardDuty API Calls with AWS CloudTrail	02/12/2018
Amazon Inspector	Logging Amazon Inspector API calls using AWS CloudTrail	11/29/2021
Amazon Inspector Classic	Logging Amazon Inspector Classic API calls with AWS CloudTrail	04/20/2016
Amazon Inspector Scan	Amazon Inspector Scan information in CloudTrail	11/27/2023
Amazon Interactive Video Service	Logging Amazon IVS API Calls with AWS CloudTrail	07/15/2020

AWS Service	CloudTrail Topics	Support began
Amazon Kendra	Logging Amazon Kendra API calls with AWS CloudTrail and Logging Amazon Kendra Intelligent Ranking API calls with AWS CloudTrail logs	05/11/2020
Amazon Keyspaces (for Apache Cassandra)	Logging Amazon Keyspaces API calls with AWS CloudTrail	01/13/2020
Amazon Managed Service for Apache Flink	Logging Managed Service for Apache Flink API calls with AWS CloudTrail	03/22/2019
Amazon Kinesis Data Streams	Logging Amazon Kinesis Data Streams API Calls Using AWS CloudTrail	04/25/2014
Amazon Kinesis Video Streams	Logging Kinesis Video Streams API Calls with AWS CloudTrail	05/24/2018
Amazon Lex	Logging Amazon Lex API Calls with CloudTrail	08/15/2017
Amazon Lightsail	Logging Lightsail API Calls with AWS CloudTrail	12/23/2016
Amazon Location Service	Logging and monitoring with AWS CloudTrail	12/15/2020
Amazon Lookout for Equipment	Monitoring Amazon Lookout for Equipment	12/01/2020
Amazon Lookout for Metrics	Viewing Amazon Lookout for Metrics API activity in AWS CloudTrail	12/08/2020

AWS Service	CloudTrail Topics	Support began
Amazon Lookout for Vision	Logging Amazon Lookout for Vision calls with AWS CloudTrail	12/01/2020
Amazon Machine Learning	Logging Amazon ML API Calls By Using AWS CloudTrail	12/10/2015
Amazon Macie	Log Amazon Macie API calls using AWS CloudTrail	05/13/2020
Amazon Managed Blockchain	Logging Amazon Managed Blockchain API calls using AWS CloudTrail Logging Ethereum for Managed Blockchain API calls using AWS CloudTrail (Preview)	04/01/2019
Amazon Managed Grafana	Logging Amazon Managed Grafana API calls using AWS CloudTrail	12/15/2020
Amazon Managed Service for Prometheus	Logging Amazon Managed Service for Prometheus API calls using AWS CloudTrail	12/15/2020
Amazon Managed Streaming for Apache Kafka	Logging API Calls with AWS CloudTrail	12/11/2018
Amazon Managed Workflows for Apache Airflow	Viewing audit logs in AWS CloudTrail	11/24/2020
Amazon MemoryDB for Redis	Logging Amazon MemoryDB for Redis API calls with AWS CloudTrail	08/19/2021

AWS Service	CloudTrail Topics	Support began
Amazon MQ	Logging Amazon MQ API Calls Using AWS CloudTrail	07/19/2018
Amazon Neptune	Logging Amazon Neptune API Calls Using AWS CloudTrail	05/30/2018
Amazon Nimble Studio	Logging Nimble Studio calls using AWS CloudTrail	06/19/2023
Amazon One Enterprise	Logging Amazon One Enterprise API calls using AWS CloudTrail	11/27/2023
Amazon OpenSearch Service	Monitoring Amazon OpenSearch Service API calls with AWS CloudTrail	10/01/2015
Amazon Personalize	Logging Amazon Personalize API Calls with AWS CloudTrail	11/28/2018
Amazon Pinpoint	Logging Amazon Pinpoint API Calls with AWS CloudTrail	02/06/2018
Amazon Pinpoint SMS and Voice API	Logging Amazon Pinpoint API Calls with AWS CloudTrail	11/16/2018
Amazon Polly	Logging Amazon Polly API Calls with AWS CloudTrail	11/30/2016
Amazon Q Business	Logging Amazon Q Business API calls using AWS CloudTrail	11/28/2023
Amazon Q Developer	Logging Amazon Q Developer API calls using AWS CloudTrail	11/28/2023
Amazon Quantum Ledger Database (Amazon QLDB)	Logging Amazon QLDB API Calls with AWS CloudTrail	09/10/2019

AWS Service	CloudTrail Topics	Support began
Amazon QuickSight	Logging Operations with CloudTrail	04/28/2017
Amazon Relational Database Service (Amazon RDS)	Logging Amazon RDS API Calls Using AWS CloudTrail	11/13/2013
Amazon RDS Performance Insights	Logging Amazon RDS API Calls Using AWS CloudTrail The Amazon RDS Performan ce Insights API is a subset of the Amazon RDS API.	06/21/2018
Amazon Redshift	Logging Amazon Redshift API Calls with AWS CloudTrail	06/10/2014
Amazon Rekognition	Logging Amazon Rekognition API Calls Using AWS CloudTrai l	04/6/2018
Amazon Route 53	Using AWS CloudTrail to Capture Requests Sent to the Route 53 API	02/11/2015
Amazon Route 53 Application Recovery Controller	Logging Amazon Route 53 Application Recovery Controller API calls using AWS CloudTrail	07/27/2021
Amazon S3	Logging Amazon S3 API Calls By Using AWS CloudTrail	Management events: 09/01/2015 Data events: 11/21/2016
Amazon S3 Glacier	Logging S3 Glacier API Calls By Using AWS CloudTrail	12/11/2014

AWS Service	CloudTrail Topics	Support began
Amazon SageMaker	Logging Amazon SageMaker API Calls with AWS CloudTrail	01/11/2018
Amazon Security Lake	Logging Amazon Security Lake API calls using CloudTrail	05/30/2023
Amazon Simple Email Service (Amazon SES)	Logging Amazon SES API Calls By Using AWS CloudTrail	05/07/2015
Amazon Simple Notification Service (Amazon SNS)	Logging Amazon SNS API Calls using AWS CloudTrail	10/09/2014
Amazon Simple Queue Service (Amazon SQS)	Logging Amazon SQS API Actions Using AWS CloudTrail	07/16/2014
Amazon Simple Workflow Service (Amazon SWF)	Recording API calls with AWS CloudTrail	Management events: 05/13/2014 Data events: 02/14/2024
Amazon Textract	Logging Amazon Textract API Calls with AWS CloudTrail	05/29/2019
Amazon Timestream	Logging Timestream API calls with AWS CloudTrail	09/30/2020
Amazon Transcribe	Logging Amazon Transcribe API Calls with AWS CloudTrail	06/28/2018
Amazon Translate	Logging Amazon Translate API Calls with AWS CloudTrail	04/04/2018
Amazon Verified Permissions	Logging Amazon Verified Permissions API calls using AWS CloudTrail	06/13/2023

AWS Service	CloudTrail Topics	Support began
Amazon Virtual Private Cloud (Amazon VPC)	Logging API Calls Using AWS CloudTrail The Amazon VPC API is a subset of the Amazon EC2 API.	11/13/2013
Amazon VPC Lattice	CloudTrail logs	03/31/2023
Amazon VPC Reachability Analyzer	Logging Reachability Analyzer API calls using AWS CloudTrail	11/27/2023
Amazon WorkDocs	Logging Amazon WorkDocs API Calls By Using AWS CloudTrail	08/27/2014
Amazon WorkMail	Logging Amazon WorkMail API Calls Using AWS CloudTrail	12/12/2017
Amazon WorkSpaces	Logging Amazon WorkSpaces API Calls by Using CloudTrail	04/09/2015
Amazon WorkSpaces Thin Client	Logging Amazon WorkSpaces Thin Client API calls using AWS CloudTrail	11/26/2023
Amazon WorkSpaces Web	Logging Amazon WorkSpaces Web API calls using AWS CloudTrail	11/30/2021
Application Auto Scaling	Logging Application Auto Scaling API calls with AWS CloudTrail	10/31/2016
AWS Amplify	Logging Amplify API calls using AWS CloudTrail	11/30/2020

AWS Service	CloudTrail Topics	Support began
AWS App Mesh	Logging App Mesh API Calls with AWS CloudTrail	AWS App Mesh 10/30/2019 App Mesh Envoy Management Service 03/18/2022
AWS App Runner	Logging App Runner API calls with AWS CloudTrail	05/18/2021
AWS AppConfig	Logging AWS AppConfig API calls using AWS CloudTrail	Management events: 07/31/2020 Data events: 01/04/2024
AWS AppFabric	Logging AWS AppFabric API calls using AWS CloudTrail	06/27/2023
AWS Application Cost Profiler	AWS Application Cost Profiler API Reference	05/13/2021
AWS Application Discovery Service	Logging Application Discovery Service API Calls with AWS CloudTrail	05/12/2016
AWS Application Transform ation Service	(Backend service used by AWS tools, such as AWS Microserv ice Extractor for .NET)	08/26/2023
AWS AppSync	Logging AWS AppSync API Calls with AWS CloudTrail	02/13/2018
AWS Artifact	Logging AWS Artifact API calls with AWS CloudTrail	01/27/2023
AWS Audit Manager	Logging AWS Audit Manager API calls with AWS CloudTrail	12/07/2020
AWS Auto Scaling	Logging AWS Auto Scaling API Calls By Using CloudTrail	08/15/2018

AWS Service	CloudTrail Topics	Support began
AWS B2B Data Interchange	Logging AWS B2B Data Interchange API calls using AWS CloudTrail	12/01/2023
AWS Backup	Logging AWS Backup API Calls with AWS CloudTrail	02/04/2019
AWS Batch	Logging AWS Batch API Calls with AWS CloudTrail	1/10/2018
AWS Billing and Cost Management	Logging AWS Billing and Cost Management API Calls with AWS CloudTrail	06/07/2018
AWS Billing Conductor	Logging AWS Billing Conductor API calls using AWS CloudTrail	03/12/2024
AWS BugBust	Logging BugBust API calls using CloudTrail	06/24/2021
AWS Certificate Manager	Using AWS CloudTrail	03/25/2016
AWS Clean Rooms	Logging AWS Clean Rooms API calls using AWS CloudTrail	03/21/2023
AWS Cloud Map	Logging AWS Cloud Map API Calls with AWS CloudTrail	11/28/2018
AWS Cloud9	Logging AWS Cloud9 API Calls with AWS CloudTrail	01/21/2019
AWS CloudFormation	Logging AWS CloudFormation API Calls in AWS CloudTrail	04/02/2014
AWS CloudHSM	Logging AWS CloudHSM API Calls By Using AWS CloudTrail	01/08/2015

AWS Service	CloudTrail Topics	Support began
AWS CloudShell	Logging and monitoring in AWS CloudShell	12/15/2020
AWS CloudTrail	AWS CloudTrail API Reference (All CloudTrail API calls are logged by CloudTrail.)	11/13/2013
AWS CodeArtifact	Logging CodeArtifact API calls with AWS CloudTrail	06/10/2020
AWS CodeBuild	Logging AWS CodeBuild API Calls with AWS CloudTrail	12/01/2016
AWS CodeCommit	Logging AWS CodeCommit API Calls with AWS CloudTrail	01/11/2017
AWS CodeDeploy	Monitoring Deployments with AWS CloudTrail	12/16/2014
AWS CodePipeline	Logging CodePipeline API calls with AWS CloudTrail	07/09/2015
AWS CodeStar	Logging AWS CodeStar API Calls with AWS CloudTrail	06/14/2017
AWS CodeStar Notifications	Logging AWS CodeStar Notifications API Calls with AWS CloudTrail	11/05/2019
AWS Config	Logging AWS Config API Calls By with AWS CloudTrail	02/10/2015
AWS Control Catalog	Logging AWS Control Catalog API calls using AWS CloudTrail	04/08/2024
AWS Control Tower	Logging AWS Control Tower Actions with AWS CloudTrail	08/12/2019

AWS Service	CloudTrail Topics	Support began
AWS Data Pipeline	Logging AWS Data Pipeline API Calls by using AWS CloudTrail	12/02/2014
AWS Database Migration Service (AWS DMS)	Logging AWS Database Migration Service API Calls Using AWS CloudTrail	02/04/2016
AWS DataSync	Logging AWS DataSync API Calls with AWS CloudTrail	11/26/2018
AWS Deadline Cloud	Logging calls with CloudTrail	04/02/2024
AWS Device Farm	Logging AWS Device Farm API Calls By Using AWS CloudTrail	07/13/2015
AWS Direct Connect	Logging AWS Direct Connect API Calls in AWS CloudTrail	03/08/2014
AWS Directory Service	Logging AWS Directory Service API Calls by Using CloudTrail	05/14/2015
AWS Elastic Beanstalk (Elastic Beanstalk)	Using Elastic Beanstalk API Calls with AWS CloudTrail	03/31/2014
AWS Elastic Disaster Recovery	Logging AWS Elastic Disaster Recovery API calls using AWS CloudTrail	11/17/2021
AWS Elemental MediaConnect	Logging AWS Elemental MediaConnect API Calls with AWS CloudTrail	11/27/2018
AWS Elemental MediaConvert	Logging AWS Elemental MediaConvert API Calls with CloudTrail	11/27/2017

AWS Service	CloudTrail Topics	Support began
AWS Elemental MediaLive	Logging MediaLive API Calls with AWS CloudTrail	01/19/2019
AWS Elemental MediaPackage	Logging AWS Elemental MediaPackage API Calls with AWS CloudTrail	12/21/2018
AWS Elemental MediaStore	Logging AWS Elemental MediaStore API Calls with CloudTrail	11/27/2017
AWS Elemental MediaTailor	Logging AWS Elemental MediaTailor API Calls with AWS CloudTrail	02/11/2019
AWS Entity Resolution	Logging AWS Entity Resolution API calls using AAWS CloudTrail	07/26/2023
AWS Fault Injection Service	Log API calls with AWS CloudTrail	03/15/2021
AWS Firewall Manager	Logging AWS Firewall Manager API Calls with AWS CloudTrail	04/05/2018
AWS Global Accelerator	Logging AWS Global Accelerator API Calls with AWS CloudTrail	11/26/2018
AWS Glue	Logging AWS Glue Operations Using AWS CloudTrail	11/07/2017
AWS Ground Station	Logging AWS Ground Station API Calls with AWS CloudTrail	05/31/2019

AWS Service	CloudTrail Topics	Support began
AWS Health	Logging AWS Health API Calls with AWS CloudTrail	11/21/2016
AWS Health Dashboard	Logging AWS Health API Calls with AWS CloudTrail	12/01/2016
AWS HealthImaging	Logging AWS HealthImaging API calls using AWS CloudTrail	07/26/2023
AWS HealthLake	Logging AWS HealthLake API calls with AWS CloudTrail	12/07/2020
AWS HealthOmics	Logging AWS HealthOmics API calls using AWS CloudTrail	11/29/2022
AWS IAM Identity Center	Logging IAM Identity Center API Calls with AWS CloudTrail	12/07/2017
AWS Identity and Access Management (IAM)	Logging IAM Events with AWS CloudTrail	11/13/2013
AWS IoT	Logging AWS IoT API Calls with AWS CloudTrail	04/11/2016
AWS IoT 1-Click	Logging AWS IoT 1-Click API Calls with AWS CloudTrail	05/14/2018
AWS IoT Analytics	Logging AWS IoT Analytics API calls with AWS CloudTrail	04/23/2018
AWS IoT Events	Logging AWS IoT Events API Calls with AWS CloudTrail	06/11/2019
AWS IoT Greengrass	Logging AWS IoT Greengrass API Calls with AWS CloudTrail	10/29/2018
AWS IoT Greengrass V2	Log AWS IoT Greengrass V2 API calls with AWS CloudTrail	12/14/2020

AWS Service	CloudTrail Topics	Support began
AWS IoT SiteWise	Logging AWS IoT SiteWise API calls with AWS CloudTrail	04/29/2020
AWS Key Management Service (AWS KMS)	Logging AWS KMS API Calls using AWS CloudTrail	11/12/2014
AWS Lake Formation	Logging AWS Lake Formation API Calls Using AWS CloudTrail	08/09/2019
AWS Lambda	Logging AWS Lambda API Calls By Using AWS CloudTrail	Management events: 04/09/2015 Data events: 11/30/2017
AWS Launch Wizard	Logging AWS Launch Wizard API calls using AWS CloudTrail	11/08/2023
AWS License Manager	Logging AWS License Manager API Calls with AWS CloudTrail	03/01/2019
AWS Mainframe Modernization	Logging AWS Mainframe Modernization API calls using AWS CloudTrail	06/08/2022
AWS Managed Services	Log management in AMS Accelerate	12/21/2016
AWS Marketplace Agreements	Logging Agreements API Calls using AWS CloudTrail	09/01/2023
AWS Marketplace Deployment Service	Logging AWS Marketplace Deployment Service calls with CloudTrail	11/29/2023

AWS Service	CloudTrail Topics	Support began
AWS Marketplace Discovery	Logging AWS Marketplace Discovery API calls using AWS CloudTrail	12/15/2022
AWS Marketplace Metering Service	Logging AWS Marketplace API Calls with AWS CloudTrail	08/22/2018
AWS Migration Hub	Logging AWS Migration Hub API Calls with AWS CloudTrail	08/14/2017
AWS Network Firewall	Logging calls to the AWS Network Firewall API with AWS CloudTrail	11/17/2020
AWS OpsWorks for Chef Automate	Logging AWS OpsWorks for Chef Automate API Calls with AWS CloudTrail	07/16/2018
AWS OpsWorks for Puppet Enterprise	Logging OpsWorks for Puppet Enterprise API Calls with AWS CloudTrail	07/16/2018
AWS OpsWorks Stacks	Logging AWS OpsWorks Stacks API Calls with AWS CloudTrail	06/04/2014
AWS Organizations	Logging AWS Organizations API calls with AWS CloudTrail	02/27/2017
AWS Outposts	Logging AWS Outposts API calls with AWS CloudTrail	02/04/2020
AWS Panorama	AWS Panorama API Reference	10/20/2021
AWS Payment Cryptography	Logging AWS Payment Cryptography API calls using AWS CloudTrail	06/08/2023

AWS Service	CloudTrail Topics	Support began
AWS Private 5G	Logging AWS Private 5G API calls using AWS CloudTrail	08/11/2022
AWS Private Certificate Authority (AWS Private CA)	Using CloudTrail	04/04/2018
AWS Proton	Logging and monitoring in AWS Proton	06/09/2021
AWS re:Post Private	Logging AWS re:Post Private API calls using AWS CloudTrail	11/26/2023
AWS Resilience Hub	AWS CloudTrail	11/10/2021
AWS Resource Access Manager (AWS RAM)	Logging AWS RAM API Calls with AWS CloudTrail	11/20/2018
AWS Resource Explorer	Logging AWS Resource Explorer API calls using AWS CloudTrail	11/07/2022
AWS Resource Groups	Logging and monitoring in Resource Groups	06/29/2018
AWS RoboMaker	Logging AWS RoboMaker API Calls with AWS CloudTrail	01/16/2019
AWS Secrets Manager	Monitor the Use of Your AWS Secrets Manager Secrets	04/05/2018
AWS Security Hub	Logging AWS Security Hub API Calls with AWS CloudTrail	11/27/2018
AWS Security Token Service (AWS STS)	Logging IAM Events with AWS CloudTrail The IAM topic includes information for AWS STS.	11/13/2013

AWS Service	CloudTrail Topics	Support began
AWS Serverless Application Repository	Logging AWS Serverless Application Repository API Calls with AWS CloudTrail	02/20/2018
AWS Service Catalog	Logging Service Catalog API Calls with AWS CloudTrail	07/06/2016
AWS Shield	Logging Shield Advanced API Calls with AWS CloudTrail	02/08/2018
AWS Snowball Edge	Logging AWS Snowball Edge API Calls with AWS CloudTrail	01/25/2019
AWS Step Functions	Logging AWS Step Functions API Calls with AWS CloudTrail	12/01/2016
AWS Storage Gateway	Logging Storage Gateway API Calls by Using AWS CloudTrail	12/16/2014
AWS Support	Logging AWS Support API calls with AWS CloudTrail	04/21/2016
AWS Support Recommend ations (Preview)	Logging AWS Support Recommendations API calls with AWS CloudTrail	05/22/2024
AWS Systems Manager	Logging AWS Systems Manager API Calls with AWS CloudTrail	11/29/2017
AWS Systems Manager Incident Manager	Logging AWS Systems Manager Incident Manager API calls using AWS CloudTrail	05/10/2021
AWS Telco Network Builder (AWS TNB)	Logging AWS Telco Network Builder API calls using AWS CloudTrail	02/21/2023

AWS Service	CloudTrail Topics	Support began
AWS Transfer for SFTP	Logging AWS Transfer for SFTP API Calls with AWS CloudTrail	01/08/2019
AWS Transit Gateway	Logging API Calls for Your Transit Gateway Using AWS CloudTrail	11/26/2018
AWS Trusted Advisor	Logging AWS Trusted Advisor console actions with AWS CloudTrail	10/22/2020
AWS Verified Access	Log AWS Verified Access API calls using AWS CloudTrail	04/27/2023
AWS WAF	Logging AWS WAF API Calls with AWS CloudTrail	04/28/2016
AWS Well-Architected Tool	Logging AWS Well-Architected Tool API Calls with AWS CloudTrail	12/15/2020
AWS X-Ray	Logging AWS X-Ray API Calls With CloudTrail	04/25/2018
Elastic Load Balancing	AWS CloudTrail Logging for Your Classic Load Balancer and AWS CloudTrail Logging for Your Application Load Balancer	04/04/2014
FreeRTOS Over-the-Air Updates (OTA)	Logging AWS IoT OTA API Calls with AWS CloudTrail	05/22/2019
Service Quotas	Logging Service Quotas API calls using AWS CloudTrail	06/24/2019

CloudTrail unsupported services

Services that are still in preview, or not yet released for general availability (GA), or which don't have public APIs, are not considered supported.

Additionally, the following AWS services and events are not supported:

- AWS Import/Export
- Amazon VPC endpoint policy-specific events

For a list of supported AWS services, see AWS service topics for CloudTrail.

Quotas in AWS CloudTrail

The following table describes quotas (formerly referred to as limits) within CloudTrail. CloudTrail has no adjustable quotas. For information about other quotas in AWS, see AWS service quotas.

Resource	Default quota	Comments
Trails per Region	5	This quota cannot be increased.
Get, describe, and list APIs	10 transactions per second (TPS)	The maximum number of operation requests you can make per second without being throttled. The `CancelQuery` , `LookupEvents` , `ListInsightsMetricData` , `PutAuditEvents` , and `StartQuery` APIs are not included in this category.
CancelQuery, StartQuery APIs	3 transactions per second (TPS)	The maximum number of operation requests you can make per second without being throttled.

Resource	Default quota	Comments
		This quota cannot be increased.
LookupEvents API	2 transactions per second (TPS)	The maximum number of operation requests you can make per second without being throttled. This quota cannot be increased.
ListInsightsMetricData API	1 transaction per second (TPS)	The maximum number of operation requests you can make per second without being throttled. This quota cannot be increased.
PutAuditEvents API	100 transactions per second (TPS)	The maximum number of operation requests you can make per second without being throttled. This quota cannot be increased.
All other APIs	1 transaction per second (TPS)	The maximum number of operation requests you can make per second without being throttled. This quota cannot be increased.

Resource	Default quota	Comments
Event data stores	10	The maximum number of event data stores that you can have in any one AWS Region. This includes single-Region event data stores for the Region as well as any multi-Region event data stores across all AWS Regions. This includes event data stores in any lifecycle stage. This quota cannot be increased.
Channels	25	This quota applies to channels used for CloudTrail Lake integrations with event sources outside of AWS, and does not apply to service-linked channels. This quota cannot be increased.
Concurrent queries	10	The maximum number of queued or running queries that you can run simultaneously in CloudTrail Lake. This quota cannot be increased.

Resource	Default quota	Comments
Events per PutAuditEvents request	100	You can add up to 100 activity events (or up to 1 MB) per `PutAuditEvents` request. This quota cannot be increased.
Event selectors	5 per trail	This quota cannot be increased.
Advanced event selectors	500 conditions across all advanced event selectors	If a trail or event data store uses advanced event selectors , a maximum of 500 total values for all conditions in all advanced event selectors is allowed. Unless a trail or event data store logs data events on all resources , such as all S3 buckets or all Lambda functions, you are limited to 250 data resources. Data resources can be distributed across event selectors, but the overall total cannot exceed 250. This quota cannot be increased.

Resource	Default quota	Comments
Data resources in event selectors	250 across all event selectors in a trail	If you choose to limit data events by using event selectors or advanced event selectors, the total number of data resources cannot exceed 250 across all event selectors in a trail. The limit of number of resources on an individua l event selector is configura ble up to 250. This upper limit is allowed only if the total number of data resources does not exceed 250 across all event selectors. Examples: • A trail with 5 event selectors, each configured with 50 data resources, is allowed. (5*50=250) • A trail with 5 event selectors, 3 of which are configured with 50 data resources, 1 of which is configured with 99 data resources, and 1 of which is configured with 1 data resource, is also allowed. ((3*50)+1+99=250) • A trail configured with 5 event selectors, all of which are configured with 100 data resources, is not allowed. (5*100=500)

Resource	Default quota	Comments
		Event selectors apply only to trails. For event data stores, you must use advanced event selectors. This quota cannot be increased. The quota does not apply if you choose to log data events on all resources, such as all S3 buckets or all Lambda functions.
Event size	All event versions: events over 256 KB cannot be sent to CloudWatch Logs Event version 1.05 and newer: total event size limit of 256 KB	Amazon CloudWatch Logs and Amazon EventBridge each allow a maximum event size of 256 KB. CloudTrail does not send events over 256 KB to CloudWatch Logs or EventBridge. Starting with event version 1.05, events have a maximum size of 256 KB. This is to help prevent exploitation by malicious actors, and allow events to be consumed by other AWS services, such as CloudWatch Logs and EventBridge.

Resource	Default quota	Comments
CloudTrail file size sent to Amazon S3	50 MB ZIP file, after compression	For both management and data events, CloudTrail sends events to S3 in maximum 50 MB (compressed) ZIP files. If enabled on the trail, log delivery notifications are sent by Amazon SNS after CloudTrail sends ZIP files to S3.

Getting started with AWS CloudTrail tutorials

If you're new to AWS CloudTrail, these tutorials can help you learn how to use its features. To use CloudTrail features, you need to have adequate permissions. This page describes the managed policies available for CloudTrail and provides information about how you can grant permissions.

Examples:

- Grant permissions to use CloudTrail
- View event history
- Create a trail to log management events
- Create an event data store for S3 data events
- View CloudTrail Lake dashboards

Grant permissions to use CloudTrail

To create, update, and manage CloudTrail resources like trails, event data stores, and channels, you need to grant permissions to use CloudTrail. This section provides information about the managed policies available for CloudTrail.

> **Note**
>
> The permissions you grant to users to perform CloudTrail administration tasks aren't the same as the permissions that CloudTrail requires to deliver log files to Amazon S3 buckets or send notifications to Amazon SNS topics. For more information about those permissions, see Amazon S3 bucket policy for CloudTrail.
> If you configure integration with Amazon CloudWatch Logs, CloudTrail also requires a role that it can assume to deliver events to an Amazon CloudWatch Logs log group. You must create the role that CloudTrail uses. For more information, see Granting permission to view and configure Amazon CloudWatch Logs information on the CloudTrail console and Sending events to CloudWatch Logs.

The following AWS managed policies are available for CloudTrail:

- **AWSCloudTrail_FullAccess** – This policy provides full access to CloudTrail actions on CloudTrail resources, such as trails, event data stores, and channels. This policy provides the required permissions to create, update, and delete CloudTrail trails, event data stores, and channels.

  This policy also provides permissions to manage the Amazon S3 bucket, the log group for CloudWatch Logs, and an Amazon SNS topic for a trail. However, the `AWSCloudTrail_FullAccess` managed policy doesn't provide permissions to delete the Amazon S3 bucket, the log group for CloudWatch Logs, or an Amazon SNS topic. For information about managed policies for other AWS services, see the *AWS Managed Policy Reference Guide*.

  > **Note**
  >
  > The **AWSCloudTrail_FullAccess** policy isn't intended to be shared broadly across your AWS account. Users with this role can turn off or reconfigure the most sensitive and important auditing functions in their AWS accounts. For this reason, you must only apply this policy to account administrators. You must closely control and monitor use of this policy.

- **AWSCloudTrail_ReadOnlyAccess** – This policy grants permissions to view the CloudTrail console, including recent events and event history. This policy also allows you to view existing trails, event data stores, and channels. Roles and users with this policy can download the event history, but they can't create or update trails, event data stores, or channels.

To provide access, add permissions to your users, groups, or roles:

- Users and groups in AWS IAM Identity Center:

  Create a permission set. Follow the instructions in Create a permission set in the *AWS IAM Identity Center User Guide*.

- Users managed in IAM through an identity provider:

  Create a role for identity federation. Follow the instructions in Creating a role for a third-party identity provider (federation) in the *IAM User Guide*.

- IAM users:
  - Create a role that your user can assume. Follow the instructions in Creating a role for an IAM user in the *IAM User Guide*.

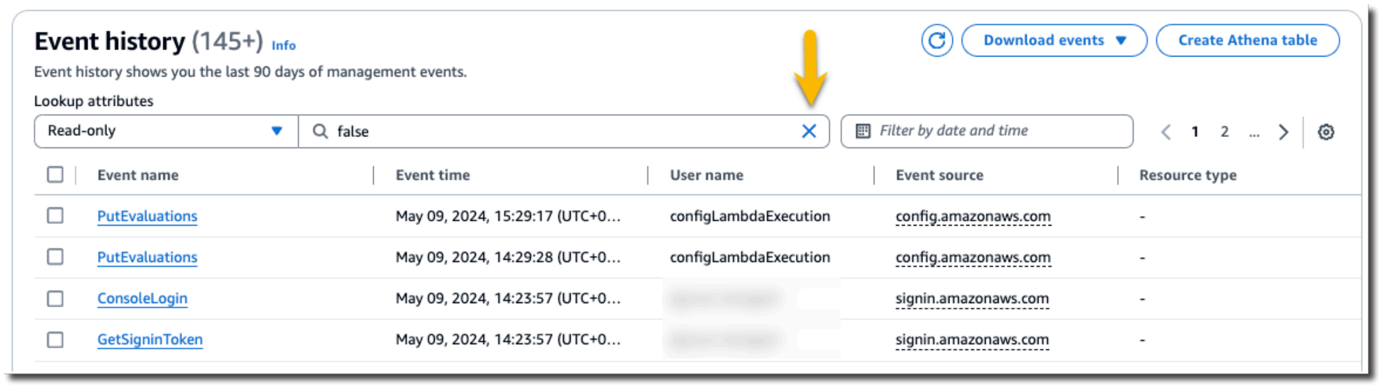
- (Not recommended) Attach a policy directly to a user or add a user to a user group. Follow the instructions in Adding permissions to a user (console) in the *IAM User Guide*.

View event history

This section describes how to use the CloudTrail **Event history** page on the CloudTrail console to view the last 90 days of management events for your AWS account for the current AWS Region.

To view the Event history

1. Sign in to the AWS Management Console and open the CloudTrail console at https://console.aws.amazon.com/cloudtrail/.
2. In the navigation pane, choose **Event history**. You see a filtered list of events, with the most recent events showing first. The default filter for events is **Read only**, set to **false**. You can clear that filter by choosing **X** at the right of the filter. You can search events in **Event history** by filtering for events on a single attribute



Event history (145+) Info

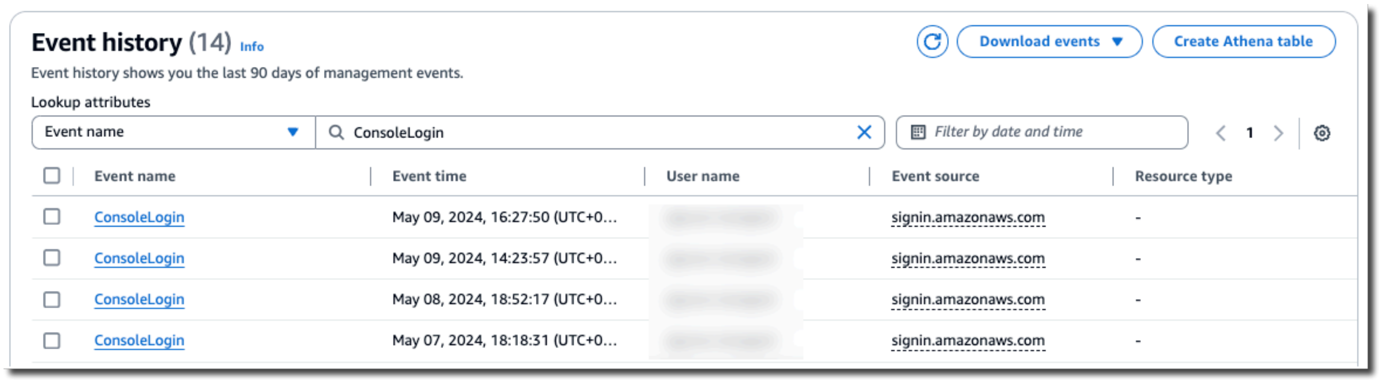
Event history shows you the last 90 days of management events.

Lookup attributes

Read-only ▼ | false ✕ | Filter by date and time

□	Event name	Event time	User name	Event source	Resource type
□	PutEvaluations	May 09, 2024, 15:29:17 (UTC+0...	configLambdaExecution	config.amazonaws.com	-
□	PutEvaluations	May 09, 2024, 14:29:28 (UTC+0...	configLambdaExecution	config.amazonaws.com	-
□	ConsoleLogin	May 09, 2024, 14:23:57 (UTC+0...	[redacted]	signin.amazonaws.com	-
□	GetSigninToken	May 09, 2024, 14:23:57 (UTC+0...	[redacted]	signin.amazonaws.com	-

3. Choose an attribute to filter on and enter the full value for the attribute. CloudTrail can't filter on a partial value. For example, to view all console login events, choose the **Event name** filter, and specify **ConsoleLogin** for the attribute value.



Event history (14) Info

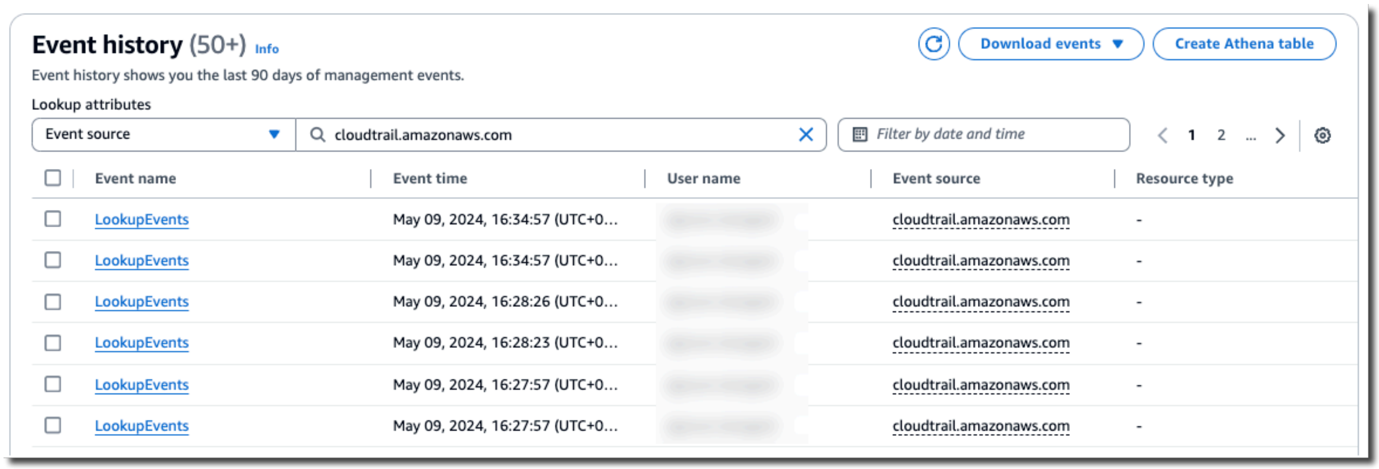
Event history shows you the last 90 days of management events.

Lookup attributes

Event name ▼ | ConsoleLogin ✕ | Filter by date and time

□	Event name	Event time	User name	Event source	Resource type
□	ConsoleLogin	May 09, 2024, 16:27:50 (UTC+0...	[redacted]	signin.amazonaws.com	-
□	ConsoleLogin	May 09, 2024, 14:23:57 (UTC+0...	[redacted]	signin.amazonaws.com	-
□	ConsoleLogin	May 08, 2024, 18:52:17 (UTC+0...	[redacted]	signin.amazonaws.com	-
□	ConsoleLogin	May 07, 2024, 18:18:31 (UTC+0...	[redacted]	signin.amazonaws.com	-

Or, to view recent CloudTrail management events, choose **Event source**, and specify `cloudtrail.amazonaws.com`. For information about the events a service logs to CloudTrail, refer to the service's API Reference.



Event history shows you the last 90 days of management events.

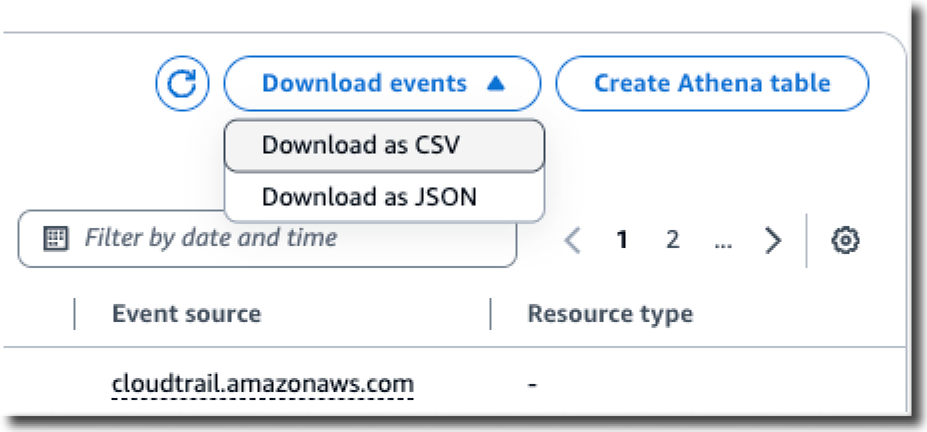
Event source: cloudtrail.amazonaws.com

☐	Event name	Event time	User name	Event source	Resource type
☐	LookupEvents	May 09, 2024, 16:34:57 (UTC+0...	[redacted]	cloudtrail.amazonaws.com	-
☐	LookupEvents	May 09, 2024, 16:34:57 (UTC+0...	[redacted]	cloudtrail.amazonaws.com	-
☐	LookupEvents	May 09, 2024, 16:28:26 (UTC+0...	[redacted]	cloudtrail.amazonaws.com	-
☐	LookupEvents	May 09, 2024, 16:28:23 (UTC+0...	[redacted]	cloudtrail.amazonaws.com	-
☐	LookupEvents	May 09, 2024, 16:27:57 (UTC+0...	[redacted]	cloudtrail.amazonaws.com	-
☐	LookupEvents	May 09, 2024, 16:27:57 (UTC+0...	[redacted]	cloudtrail.amazonaws.com	-

4. To view a specific management event, choose the event name. On the event details page, you can view details about the event, see any referenced resources, and view the event record.

5. To compare events, select up to five events by filling their check boxes in the left margin of the **Event history** table. You can view details for selected events side-by-side in the **Compare event details** table.

6. You can save event history by downloading it as a file in CSV or JSON format. Downloading your event history can take a few minutes.



For more information, see [Working with CloudTrail Event history](#).

Create a trail to log management events

For your first trail, we recommend creating a trail that logs all management events and does not log any data events or Insights events. Examples of management events include security events such as IAM `CreateUser` and `AttachRolePolicy` events, resource events such as `RunInstances` and `CreateBucket`, and many more. You will create an Amazon S3 bucket where you will store the log files for the trail as part of creating the trail in the CloudTrail console.

> **Note**
>
> AWS Control Tower sets up a new CloudTrail trail logging management events when you set up a landing zone. It is an organization-level trail, which means that it logs all management events for the management account and all member accounts in the organization. For more information, see About logging in AWS Control Tower in the *AWS CloudTrail User Guide*.
>
> This tutorial assumes you are creating your first trail. Depending on the number of trails you have in your AWS account, and how those trails are configured, the following procedure might or might not incur expenses. CloudTrail stores log files in an Amazon S3 bucket, which incurs costs. For more information about pricing, see AWS CloudTrail Pricing and Amazon S3 Pricing.

To create a trail

1. Sign in to the AWS Management Console and open the CloudTrail console at https://console.aws.amazon.com/cloudtrail/.
2. In the **Region** selector, choose the AWS Region where you want your trail to be created. This is the home Region for the trail.

   > **Note**
   >
   > The home Region is the only AWS Region where you can update the trail after it is created.

3. On the CloudTrail service home page, the **Trails** page, or the **Trails** section of the **Dashboard** page, choose **Create trail**.

4. In **Trail name**, give your trail a name, such as *`management-events`*. As a best practice, use a name that quickly identifies the purpose of the trail. In this case, you're creating a trail that logs management events.

5. Leave the default setting for **Enable for all accounts in my organization**. This option won't be available to change unless you have accounts configured in Organizations.

6. For **Storage location**, choose **Create new S3 bucket** to create a bucket. When you create a bucket, CloudTrail creates and applies the required bucket policies. If you choose to create a new S3 bucket, your IAM policy needs to include permission for the `s3:PutEncryptionConfiguration` action because by default server-side encryption is enabled for the bucket. Give your bucket a name that makes it easy to identify.

   To make it easier to find your logs, create a new folder (also known as a *prefix*) in an existing bucket to store your CloudTrail logs.

   > **Note**
   >
   > The name of your Amazon S3 bucket must be globally unique. For more information, see Bucket naming rules in the *Amazon Simple Storage Service User Guide*.

7. Clear the check box to disable **Log file SSE-KMS encryption**. By default, your log files are encrypted with SSE-S3 encryption. For more information about this setting, see Using server-side encryption with Amazon S3 managed keys (SSE-S3).

8. Leave default settings in **Additional settings**.

9. Leave the default settings for **CloudWatch Logs**. For now, do not send logs to Amazon CloudWatch Logs.

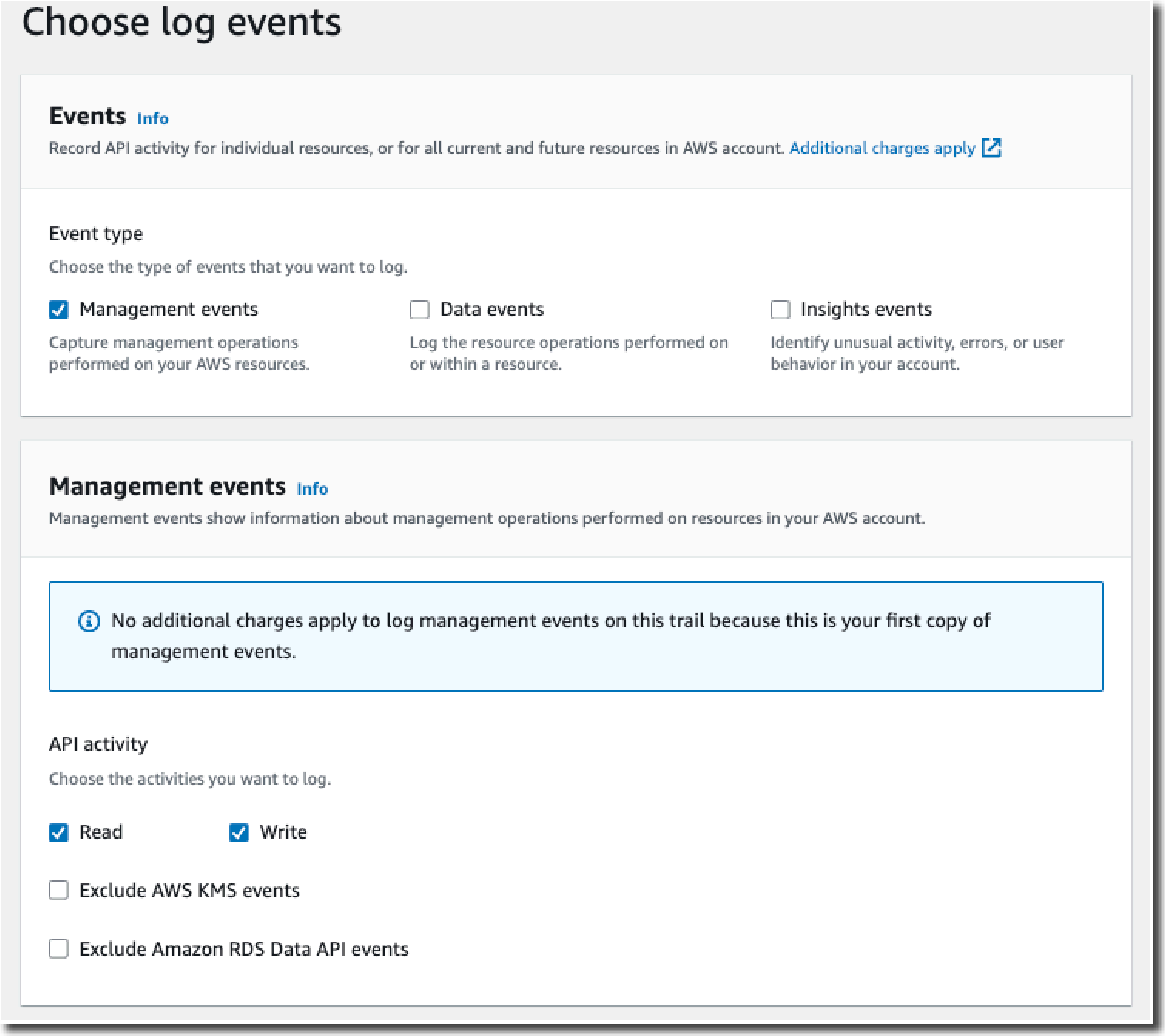
10. (Optional) In **Tags**, add one or more custom tags (key-value pairs) to your trail. Tags can help you identify your CloudTrail trails and other resources, such as the Amazon S3 buckets that contain CloudTrail log files. For example, you could attach a tag with the name **`Compliance`** and the value **`Auditing`**.

    > **Note**
    >
    > Though you can add tags to trails when you create them in the CloudTrail console, and you can create an Amazon S3 bucket to store your log files in the CloudTrail console, you cannot add tags to the Amazon S3 bucket from the CloudTrail console. For more

information about viewing and changing the properties of an Amazon S3 bucket, including adding tags to a bucket, see the *Amazon S3 User Guide*.

When you are finished creating tags, choose **Next**.

11. On the **Choose log events** page, select event types to log. For this trail, keep the default, **Management events**. In the **Management events** area, choose to log both **Read** and **Write** events, if they are not already selected. Leave the check boxes for **Exclude AWS KMS events** and **Exclude Amazon RDS Data API events** empty, to log all management events.



Choose log events

Events Info

Record API activity for individual resources, or for all current and future resources in AWS account. Additional charges apply

Event type

Choose the type of events that you want to log.

☑ Management events
Capture management operations performed on your AWS resources.

□ Data events
Log the resource operations performed on or within a resource.

□ Insights events
Identify unusual activity, errors, or user behavior in your account.

Management events Info

Management events show information about management operations performed on resources in your AWS account.

No additional charges apply to log management events on this trail because this is your first copy of management events.

API activity

Choose the activities you want to log.

☑ Read ☑ Write

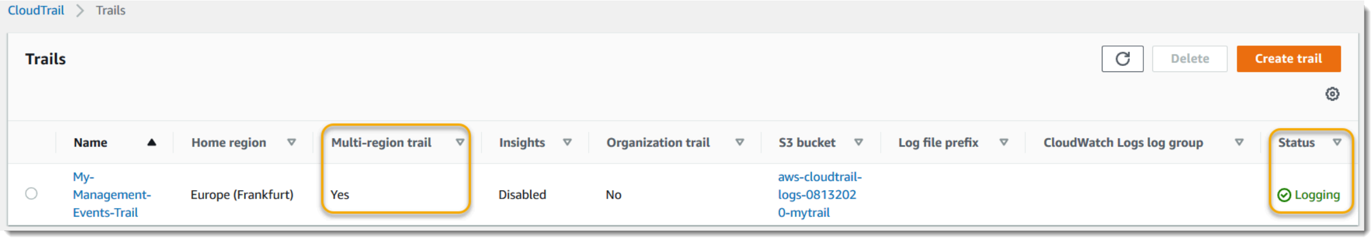
□ Exclude AWS KMS events

□ Exclude Amazon RDS Data API events

12. Leave default settings for **Data events** and **Insights events**. This trail will not log any data or CloudTrail Insights events. Choose **Next**.

13. On the **Review and create** page, review the settings you've chosen for your trail. Choose **Edit** for a section to go back and make changes. When you are ready to create your trail, choose **Create trail**.

14. The **Trails** page shows your new trail in the table. Note that the trail is set to **Multi-region trail** by default, and that logging is turned on for the trail by default.



	Name	Home region	Multi-region trail	Insights	Organization trail	S3 bucket	Log file prefix	CloudWatch Logs log group	Status
○	My-Management-Events-Trail	Europe (Frankfurt)	Yes	Disabled	No	aws-cloudtrail-logs-08132020-mytrail			✓ Logging

For more information about trails, see Working with CloudTrail trails.

View your log files

Within an average of about 5 minutes of creating your first trail, CloudTrail delivers the first set of log files to the Amazon S3 bucket for your trail. You can look at these files and learn about the information they contain.

> **Note**
>
> CloudTrail typically delivers logs within an average of about 5 minutes of an API call. This time is not guaranteed. Review the AWS CloudTrail Service Level Agreement for more information.
> If you misconfigure your trail (for example, the S3 bucket is unreachable), CloudTrail will attempt to redeliver the log files to your S3 bucket for 30 days, and these attempted-to-deliver events will be subject to standard CloudTrail charges. To avoid charges on a misconfigured trail, you need to delete the trail.

To view your log files

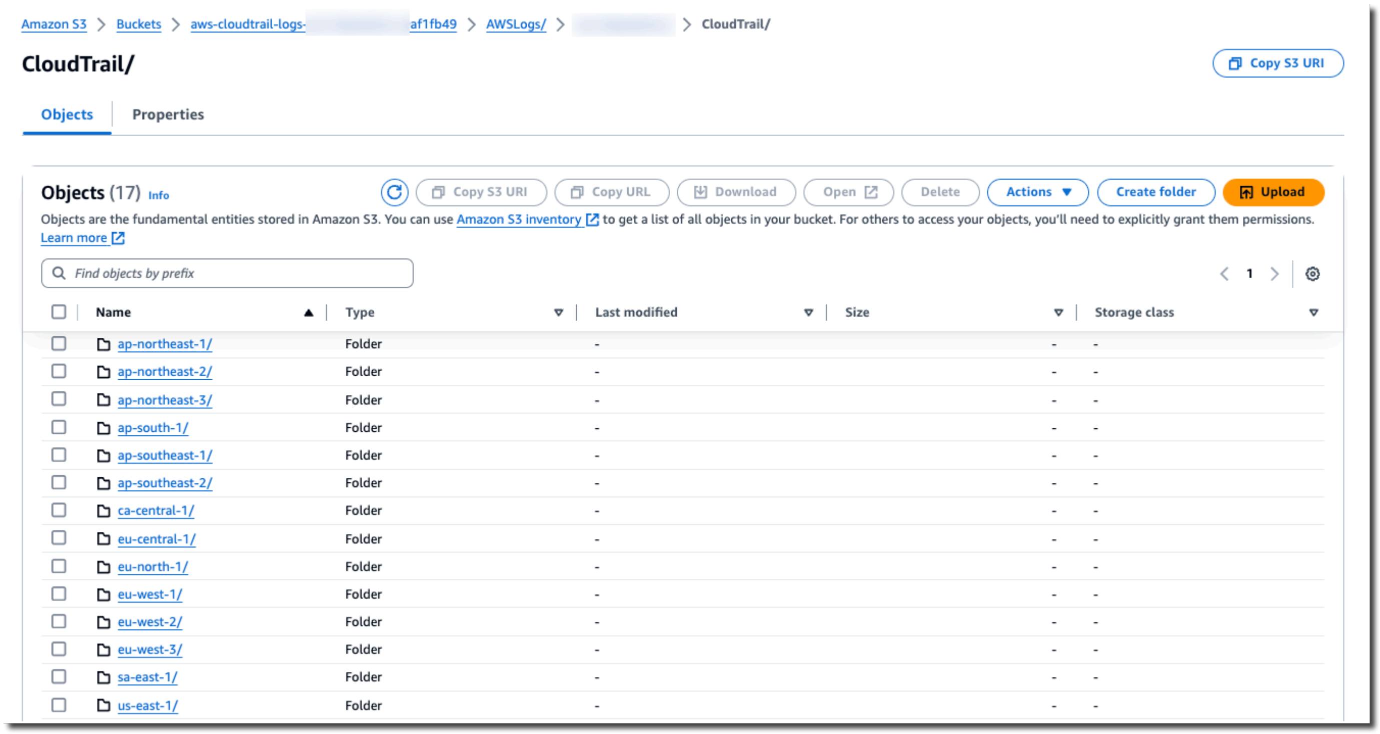
1. Sign in to the AWS Management Console and open the CloudTrail console at https://console.aws.amazon.com/cloudtrail/.

2. In the navigation pane, choose **Trails**. On the **Trails** page, find the name of the trail you just created (in the example, *management-events*).

3. In the row for the trail, choose the value for the S3 bucket.

4. The Amazon S3 console opens and shows two folders for the bucket: `CloudTrail-Digest` and `CloudTrail`. Choose the **CloudTrail** folder to view the log files.

5. If you created a multi-Region trail, there is a folder for each AWS Region. Choose the folder for the AWS Region where you want to review log files. For example, if you want to review the log files for the US East (Ohio) Region, choose **us-east-2**.



Amazon S3 > Buckets > aws-cloudtrail-logs-[redacted]af1fb49 > AWSLogs/ > [redacted] > CloudTrail/

CloudTrail/

Objects (17)

Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

	Name	Type	Last modified	Size	Storage class
□	ap-northeast-1/	Folder	-	-	-
□	ap-northeast-2/	Folder	-	-	-
□	ap-northeast-3/	Folder	-	-	-
□	ap-south-1/	Folder	-	-	-
□	ap-southeast-1/	Folder	-	-	-
□	ap-southeast-2/	Folder	-	-	-
□	ca-central-1/	Folder	-	-	-
□	eu-central-1/	Folder	-	-	-
□	eu-north-1/	Folder	-	-	-
□	eu-west-1/	Folder	-	-	-
□	eu-west-2/	Folder	-	-	-
□	eu-west-3/	Folder	-	-	-
□	sa-east-1/	Folder	-	-	-
□	us-east-1/	Folder	-	-	-

6. Navigate the bucket folder structure to the year, the month, and the day where you want to review logs of activity in that Region. In that day, there are a number of files. The name of the files begin with your AWS account ID, and end with the extension `.gz`. For example, if your account ID is *123456789012*, you would see files with names similar to this: *123456789012*_CloudTrail_*us-east-2_20240512T0000Z_EXAMPLE*.json.gz.

   To view these files, you can download them, unzip them, and then view them in a plain-text editor or a JSON file viewer. Some browsers also support viewing .gz and JSON files directly. We recommend using a JSON viewer, as it makes it easier to parse the information in CloudTrail log files.

Create an event data store for S3 data events

You can create an event data store to log CloudTrail events (management events, data events), CloudTrail Insights events, AWS Audit Manager evidence, AWS Config configuration items, or non-AWS events.

When you create an event data store for data events, you choose the AWS services and resource types for which you want to log data events. For information about AWS services that log data events, see Data events.

This walkthrough shows you how to create an event data store for Amazon S3 data events. In this tutorial, instead of logging all Amazon S3 data events, we'll choose a custom log selector template to log events only when an object is deleted from a specific S3 bucket.

To create an event data store for S3 data events

1. Sign in to the AWS Management Console and open the CloudTrail console at https://console.aws.amazon.com/cloudtrail/.
2. From the navigation pane, under **Lake**, choose **Event data stores**.
3. Choose **Create event data store**.
4. On the **Configure event data store** page, in **General details**, give your event data store a name, such as *s3-data-events-eds*. As a best practice, use a name that quickly identifies the purpose of the event data store. For information about CloudTrail naming requirements, see Naming requirements for CloudTrail resources, S3 buckets, and KMS keys.
5. Choose the **Pricing option** that you want to use for your event data store. The pricing option determines the cost for ingesting and storing events, and the default and maximum retention periods for your event data store. For more information, see AWS CloudTrail Pricing and Managing CloudTrail Lake costs.

   The following are the available options:

   - **One-year extendable retention pricing** - Generally recommended if you expect to ingest less than 25 TB of event data per month and want a flexible retention period of up to 10 years. For the first 366 days (the default retention period), storage is included at no additional charge with ingestion pricing. After 366 days, extended retention is available at pay-as-you-go pricing. This is the default option.
     - **Default retention period:** 366 days

    - **Maximum retention period:** 3,653 days
- **Seven-year retention pricing** - Recommended if you expect to ingest more than 25 TB of event data per month and need a retention period of up to 7 years. Retention is included with ingestion pricing at no additional charge.
    - **Default retention period:** 2,557 days
    - **Maximum retention period:** 2,557 days

6. Specify a retention period for the event data store. Retention periods can be between 7 days and 3,653 days (about 10 years) for the **One-year extendable retention pricing** option, or between 7 days and 2,557 days (about seven years) for the **Seven-year retention pricing** option.

   CloudTrail Lake determines whether to retain an event by checking if the `eventTime` of the event is within the specified retention period. For example, if you specify a retention period of 90 days, CloudTrail will remove events when their `eventTime` is older than 90 days.
7. (Optional) In **Encryption**. choose whether you want to encrypt the event data store using your own KMS key. By default, all events in an event data store are encrypted by CloudTrail using a KMS key that AWS owns and manages for you.

   To enable encryption using your own KMS key, choose **Use my own AWS KMS key**. Choose **New** to have an AWS KMS key created for you, or choose **Existing** to use an existing KMS key. In **Enter KMS alias**, specify an alias, in the format `alias/MyAliasName`. Using your own KMS key requires that you edit your KMS key policy to allow CloudTrail logs to be encrypted and decrypted. For more information, see Configure AWS KMS key policies for CloudTrail. CloudTrail also supports AWS KMS multi-Region keys. For more information about multi-Region keys, see Using multi-Region keys in the *AWS Key Management Service Developer Guide*.

   Using your own KMS key incurs AWS KMS costs for encryption and decryption. After you associate an event data store with a KMS key, the KMS key cannot be removed or changed.

   > **Note**
   >
   > To enable AWS Key Management Service encryption for an organization event data store, you must use an existing KMS key for the management account.
8. (Optional) If you want to query against your event data using Amazon Athena, choose **Enable** in **Lake query federation**. Federation lets you view the metadata associated with the event data store in the AWS Glue Data Catalog and run SQL queries against the event data in Athena.

The table metadata stored in the AWS Glue Data Catalog lets the Athena query engine know how to find, read, and process the data that you want to query. For more information, see Federate an event data store.

To enable Lake query federation, choose **Enable** and then do the following:

a. Choose whether you want to create a new role or use an existing IAM role. AWS Lake Formation uses this role to manage permissions for the federated event data store. When you create a new role using the CloudTrail console, CloudTrail automatically creates a role with the required permissions. If you choose an existing role, be sure the policy for the role provides the required minimum permissions.

b. If you are creating a new role, enter a name to identify the role.

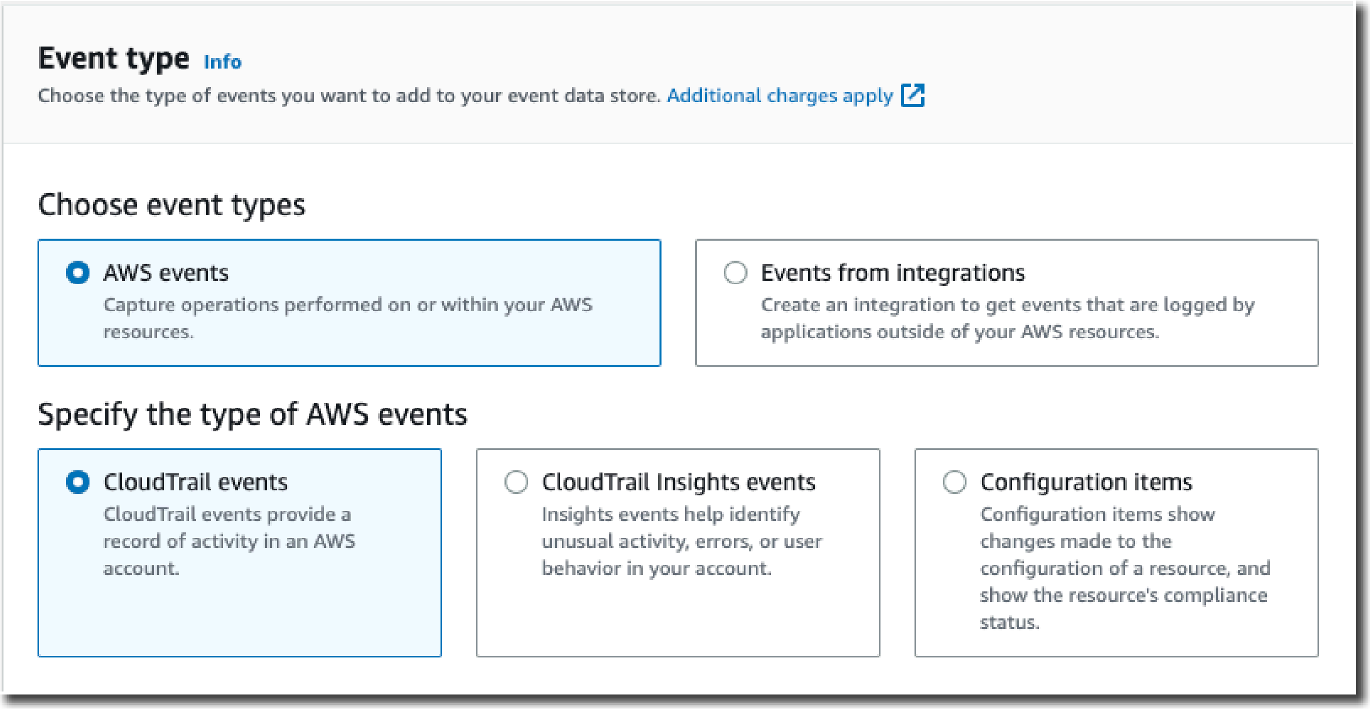
c. If you are using an existing role, choose the role you want to use. The role must exist in your account.

9. (Optional) In **Tags**, add one or more custom tags (key-value pairs) to your event data store. Tags can help you identify your CloudTrail event data stores. For example, you could attach a tag with the name **`stage`** and the value **`prod`**. You can use tags to limit access to your event data store. You can also use tags to track the query and ingestion costs for your event data store.

   For information about how to use tags to track costs, see Creating user-defined cost allocation tags for CloudTrail Lake event data stores. For information about how to use IAM policies to authorize access to an event data store based on tags, see Examples: Denying access to create or delete event data stores based on tags. For information about how you can use tags in AWS, see Tagging your AWS resources in the *Tagging AWS Resources User Guide*.

10. Choose **Next** to configure the event data store.

11. On the **Choose events** page, leave the default selections for **Event type**.



Event type Info

Choose the type of events you want to add to your event data store. Additional charges apply

Choose event types

● AWS events
Capture operations performed on or within your AWS resources.

○ Events from integrations
Create an integration to get events that are logged by applications outside of your AWS resources.

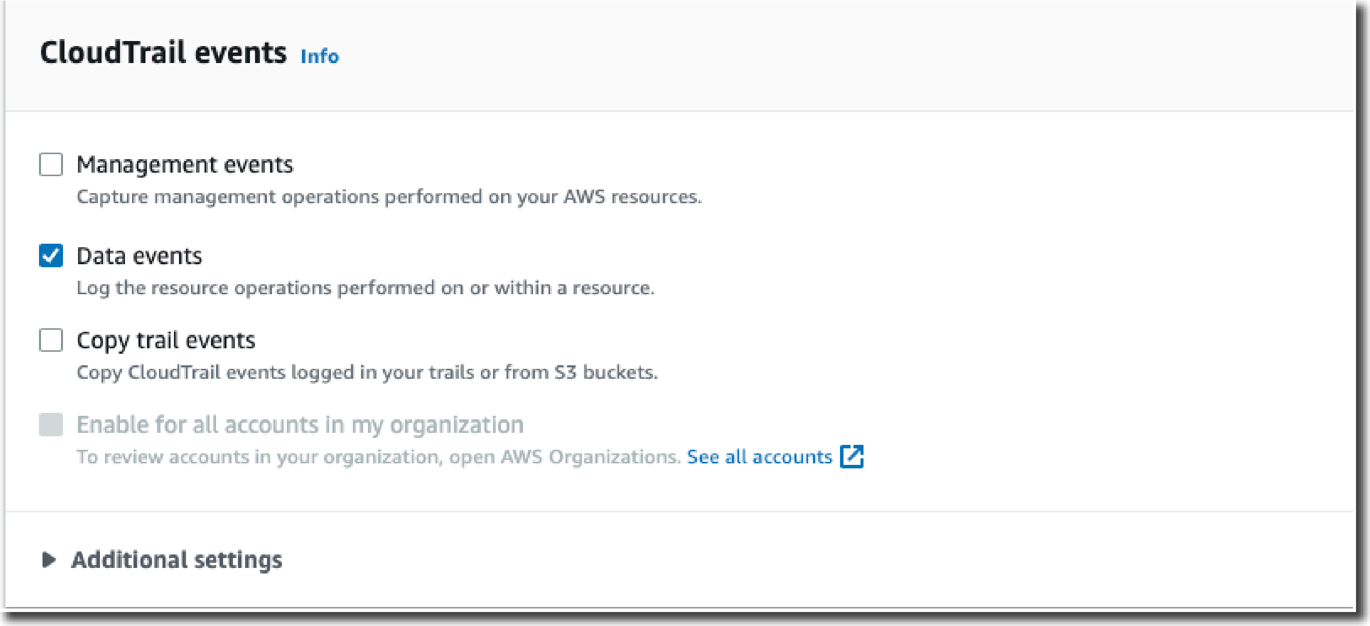
Specify the type of AWS events

● CloudTrail events
CloudTrail events provide a record of activity in an AWS account.

○ CloudTrail Insights events
Insights events help identify unusual activity, errors, or user behavior in your account.

○ Configuration items
Configuration items show changes made to the configuration of a resource, and show the resource's compliance status.

12. For **CloudTrail events**, choose **Data events** and deselect **Management events**. For more information about data events, see [Logging data events](#).



CloudTrail events Info

□ Management events
Capture management operations performed on your AWS resources.

☑ Data events
Log the resource operations performed on or within a resource.

□ Copy trail events
Copy CloudTrail events logged in your trails or from S3 buckets.

□ Enable for all accounts in my organization
To review accounts in your organization, open AWS Organizations. See all accounts

▶ Additional settings

13. Leave the default setting for **Copy trail events**. You'd use this option to copy existing trail events to your event data store. For more information, see [Copy trail events to an event data store](#).

14. Choose **Enable for all accounts in my organization** if this is an organization event data store. This option won't be available to change unless you have accounts configured in AWS Organizations.

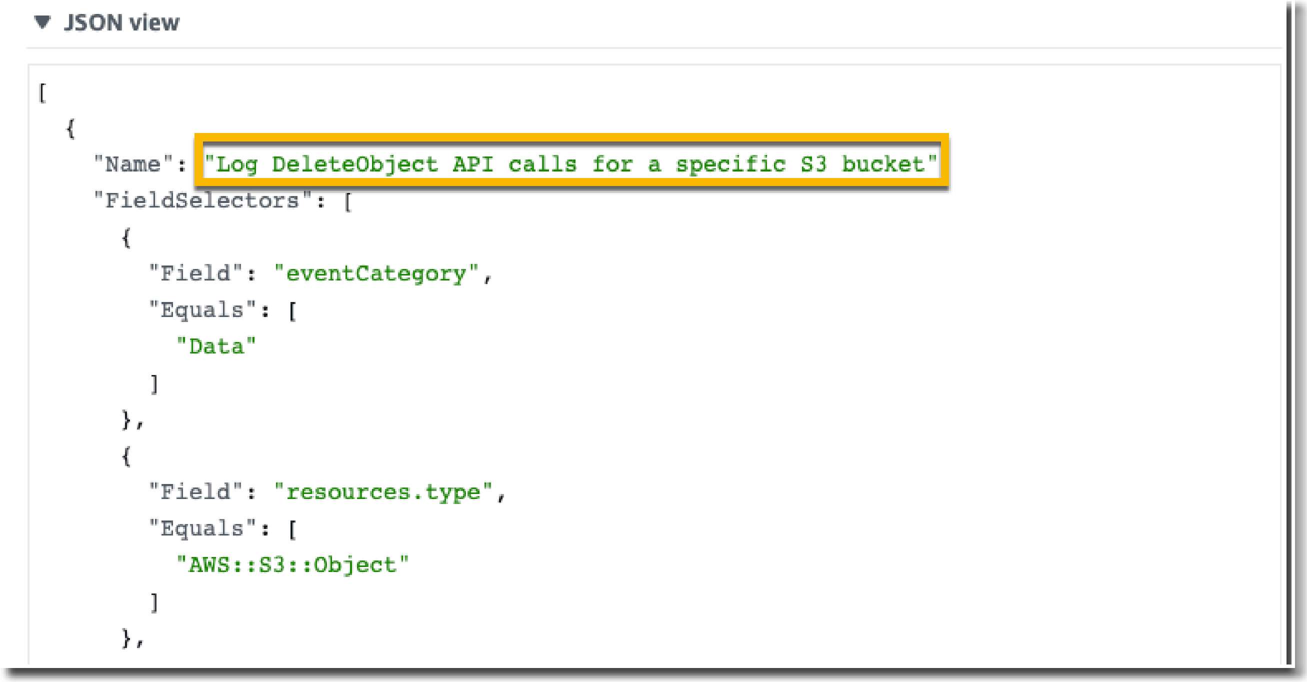
15. For **Additional settings** leave the default selections. By default, an event data store collects events for all AWS Regions and starts ingesting events when it's created.

16. For **Data events**, make the following selections:

    a. In **Data event type**, choose **S3**. The data event type identifies the AWS service and resource on which data events are logged.

    b. In **Log selector template**, choose **Custom**. Choosing **Custom** lets you define a custom event selector to filter on the `eventName`, `resources.ARN`, and `readOnly` fields. For information about these fields, see AdvancedFieldSelector in the *AWS CloudTrail API Reference*.

    c. (Optional) In **Selector name**, enter a name to identify your selector. The selector name is a descriptive name for an advanced event selector, such as "Log DeleteObject API calls for a specific S3 bucket". The selector name is listed as `Name` in the advanced event selector and is viewable if you expand the **JSON view**.

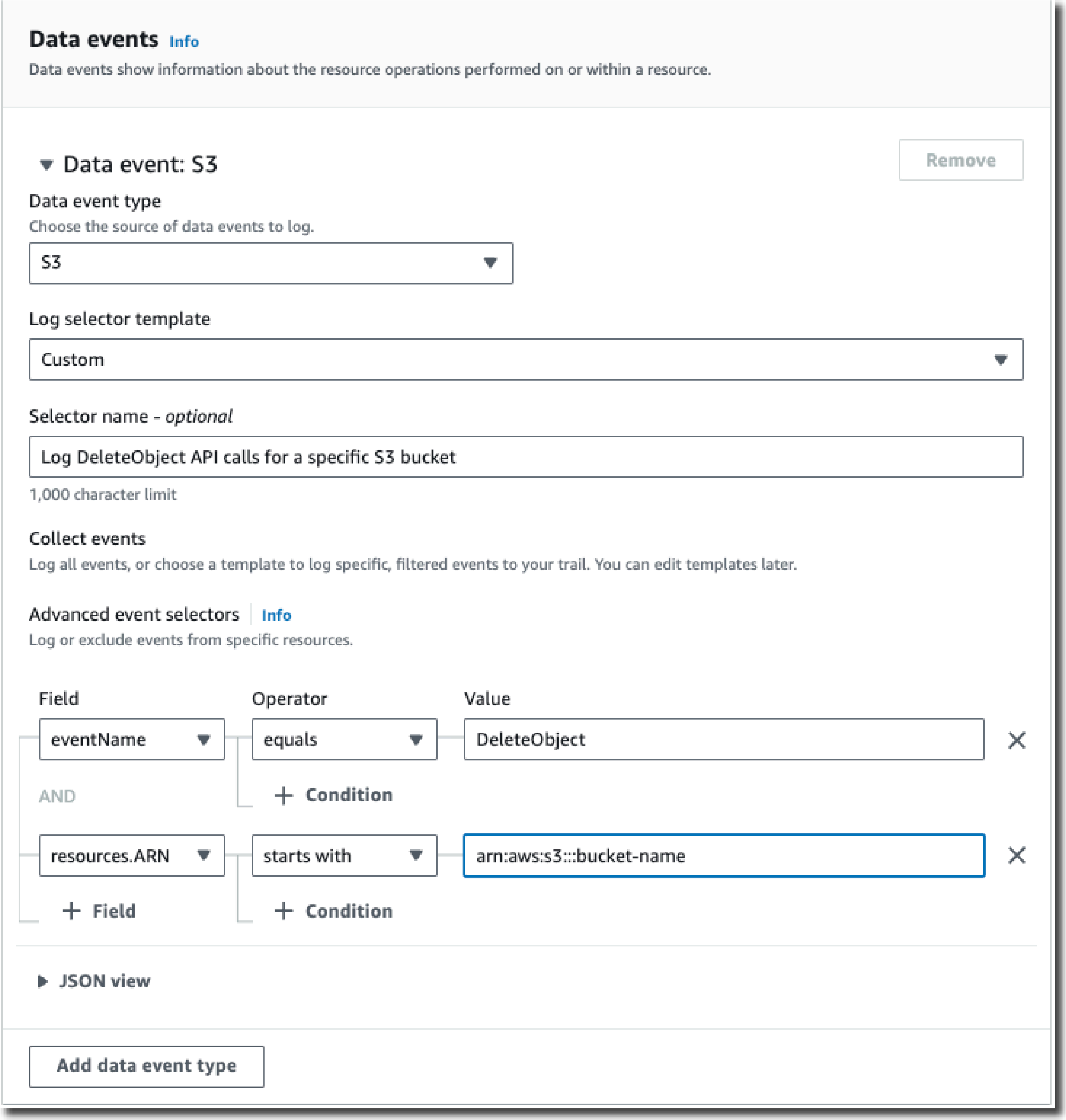


```
[
  {
    "Name": "Log DeleteObject API calls for a specific S3 bucket"
    "FieldSelectors": [
      {
        "Field": "eventCategory",
        "Equals": [
          "Data"
        ]
      },
      {
        "Field": "resources.type",
        "Equals": [
          "AWS::S3::Object"
        ]
      },
```

    d. In **Advanced event selectors**, we'll build the custom event selector to filter on the `eventName` and `resources.ARN` fields. Advanced event selectors for an event data store work the same as advanced event selectors that you apply to a trail. For more information about how to build advanced event selectors, see Logging data events with advanced event selectors.

i. For **Field** choose **eventName**. For **Operator**, choose **equals**. For **Value**, enter **`DeleteObject`**. Choose **+ Field** to filter on another field.

ii. For **Field**, choose **resources.ARN**. For **Operator**, choose **StartsWith**. For **Value**, enter the ARN for your bucket (for example, arn:aws:s3:::*`DOC-EXAMPLE-BUCKET`*). For information about how to get the ARN, see Amazon S3 resources in the *Amazon Simple Storage Service User Guide*.



Data events Info

Data events show information about the resource operations performed on or within a resource.

▼ Data event: S3

Remove

Data event type

Choose the source of data events to log.

S3

Log selector template

Custom

Selector name - *optional*

Log DeleteObject API calls for a specific S3 bucket

1,000 character limit

Collect events

Log all events, or choose a template to log specific, filtered events to your trail. You can edit templates later.

Advanced event selectors Info

Log or exclude events from specific resources.

Field	Operator	Value
eventName	equals	DeleteObject
AND resources.ARN	starts with	arn:aws:s3:::bucket-name

17. Choose **Next** to review your choices.

18. On the **Review and create** page, review your choices. Choose **Edit** to make changes to a section. When you're ready to create the event data store, choose **Create event data store**.

19. The new event data store is visible in the **Event data stores** table on the **Event data stores** page.

    From this point forward, the event data store captures events that match its advanced event selectors. Events that occurred before you created the event data store are not in the event data store, unless you opted to copy existing trail events.

You are now ready to run queries on your event data store. For information about how to view and run sample queries, see View sample queries with the CloudTrail console.

For more information about CloudTrail Lake, see Working with AWS CloudTrail Lake.

View CloudTrail Lake dashboards

This walkthrough shows you how to view CloudTrail Lake dashboards. CloudTrail Lake dashboards let you visualize the events in your event data store and see trends, such as top users and top errors.

Each dashboard consists of multiple widgets and each widget represents a SQL query. To populate the dashboard, CloudTrail runs systems-generated queries. Queries incur charges based upon the amount of data scanned.

> **Note**
>
> Currently, dashboards are only available for event data stores that collect CloudTrail management events, Amazon S3 data events, and Insights events.

To view Lake dashboards

1. Sign in to the AWS Management Console and open the CloudTrail console at https://console.aws.amazon.com/cloudtrail/.
2. From the navigation pane, under **Lake**, choose **Dashboard**.
3. The first time you view the **Dashboards** page, CloudTrail asks you to acknowledge the costs associated with running queries. Choose **I agree** to acknowledge the cost of running queries. This is a one time confirmation. For more information about CloudTrail pricing, see CloudTrail Pricing.

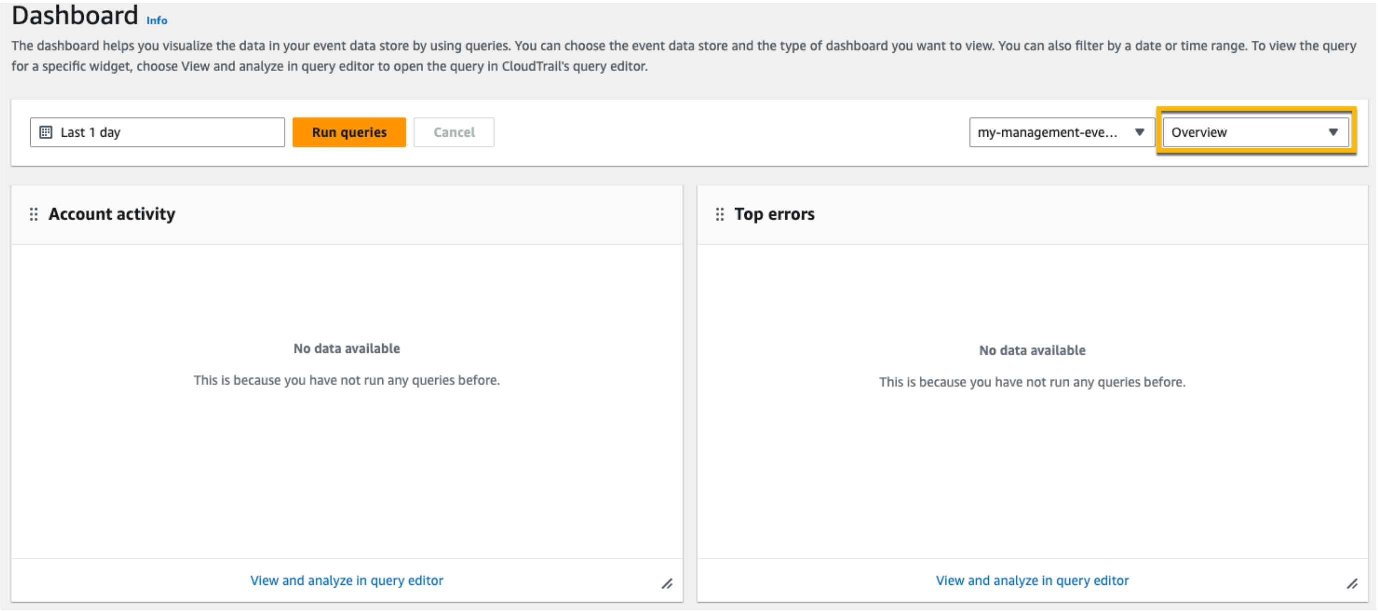
4. Choose your event data store from the list and then choose the dashboard type you want to view.

   The following are the possible dashboard types.

   - **Overview** dashboard - Shows the most active users, AWS Regions, and AWS services by event count. You can also view information about `read` and `write` management event activity, most throttled events, and the top errors. This dashboard is available for event data stores that collect management events.
   - **Management Events** dashboard - Shows console sign-in events, access denied events, destructive actions, and top errors by user. You can also view information about TLS versions and outdated TLS calls by user. This dashboard is available for event data stores that collect management events.
   - **S3 Data Events** dashboard - Shows S3 account activity, most accessed S3 objects, top S3 users, and top S3 actions. This dashboard is available for event data stores that collect Amazon S3 data events.
   - **Insights Events** dashboard - Shows the overall proportion of Insights events by Insights type, the proportion of Insights events by Insights type for the top users and services, and the number of Insights events per day. The dashboard also includes a widget that lists up to 30 days of Insights events. This dashboard is only available for event data stores that collect Insights events.

   > **Note**
   >
   > - After you enable CloudTrail Insights for the first time on the source event data store, it can take up to 7 days for CloudTrail to deliver the first Insights event, if unusual activity is detected. For more information, see Understanding Insights events delivery.
   > - The **Insights Events** dashboard only displays information about the Insights events collected by the selected event data store, which is determined by the configuration of the source event data store. For example, if you configure the source event data store to enable Insights events on `ApiCallRateInsight` but not `ApiErrorRateInsight`, you won't see information about Insights events on `ApiErrorRateInsight`.

   In this example, we've chosen the **Overview** dashboard.



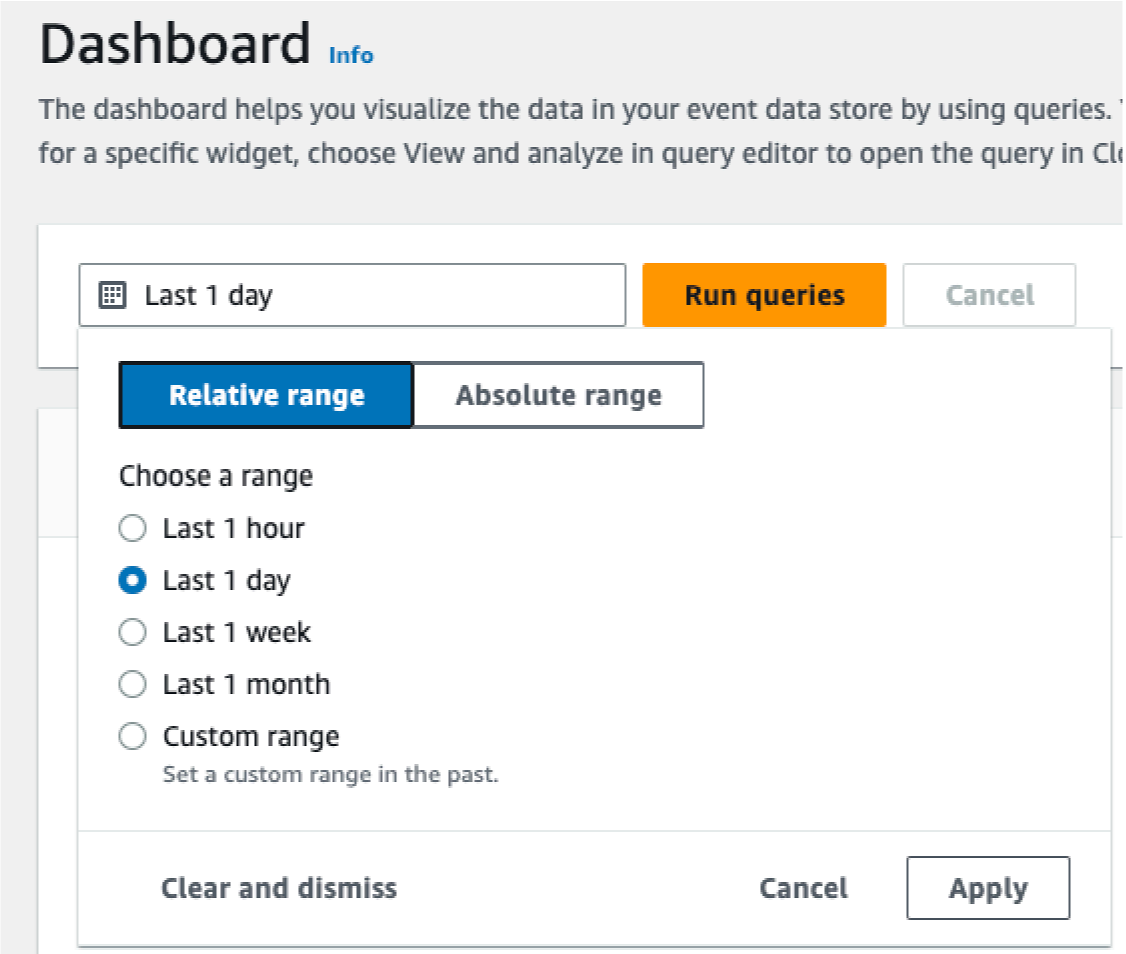
Dashboard Info

The dashboard helps you visualize the data in your event data store by using queries. You can choose the event data store and the type of dashboard you want to view. You can also filter by a date or time range. To view the query for a specific widget, choose View and analyze in query editor to open the query in CloudTrail's query editor.

Last 1 day
Run queries
Cancel
my-management-eve...
Overview
Account activity
No data available
This is because you have not run any queries before.
View and analyze in query editor
Top errors
No data available
This is because you have not run any queries before.
View and analyze in query editor

5. Choose the date field to filter on a time range and then choose **Apply**. Choose **Absolute range** to select a specific date and time range. Choose **Relative range** to select a predefined time range or a custom range. By default, the dashboard displays event data for the past 24 hours.

> **Note**
>
> Because CloudTrail queries are charged based on the amount of data scanned, you can reduce costs by filtering on a narrower time range.



Dashboard Info

The dashboard helps you visualize the data in your event data store by using queries. '
for a specific widget, choose View and analyze in query editor to open the query in Cl

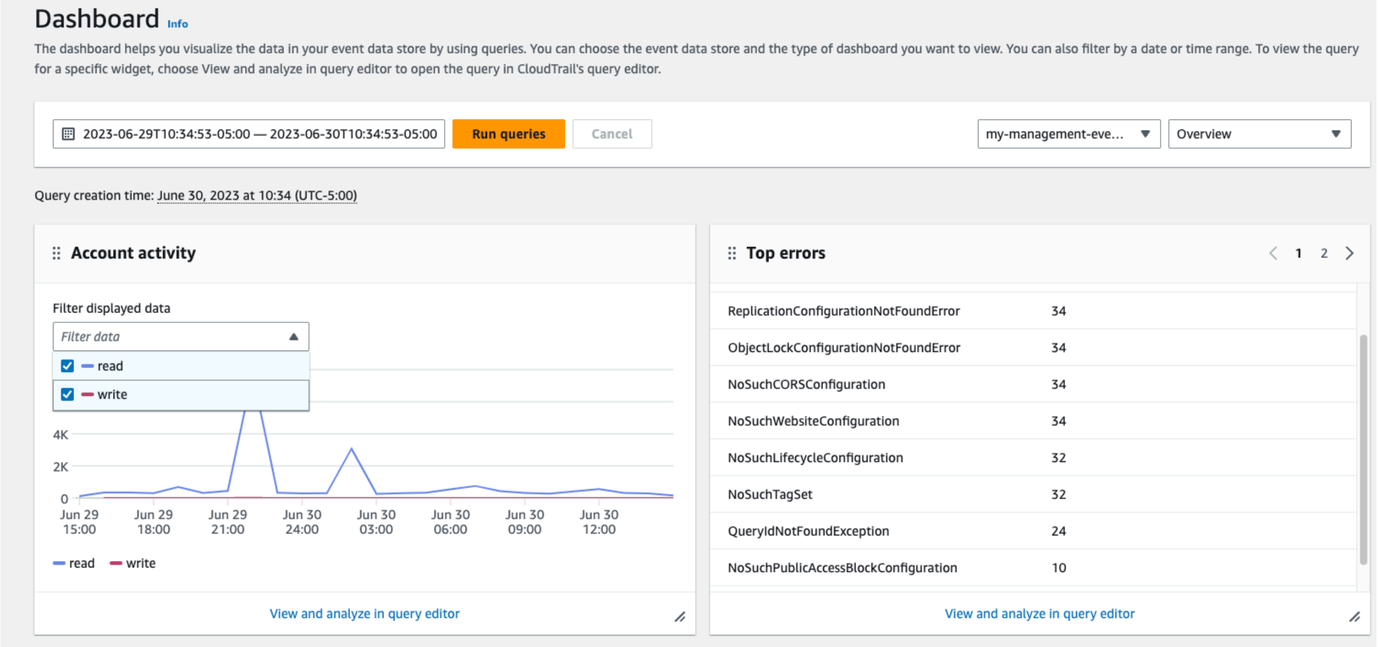
Last 1 day

Choose a range

○ Last 1 hour
● Last 1 day
○ Last 1 week
○ Last 1 month
○ Custom range
Set a custom range in the past.

6. Choose **Run queries** to populate the dashboard. Each widget individually displays the status of its associated query and presents data when its query completes.

   You can perform additional filtering on some widgets, such as **Account activity**, which lets you filter on `read` and `write` event activity.



Dashboard Info

The dashboard helps you visualize the data in your event data store by using queries. You can choose the event data store and the type of dashboard you want to view. You can also filter by a date or time range. To view the query for a specific widget, choose View and analyze in query editor to open the query in CloudTrail's query editor.

2023-06-29T10:34:53-05:00 — 2023-06-30T10:34:53-05:00 | Run queries | Cancel | my-management-eve... ▼ | Overview ▼

Query creation time: June 30, 2023 at 10:34 (UTC-5:00)

Account activity

Filter displayed data: Filter data ▲

☑ read
☑ write

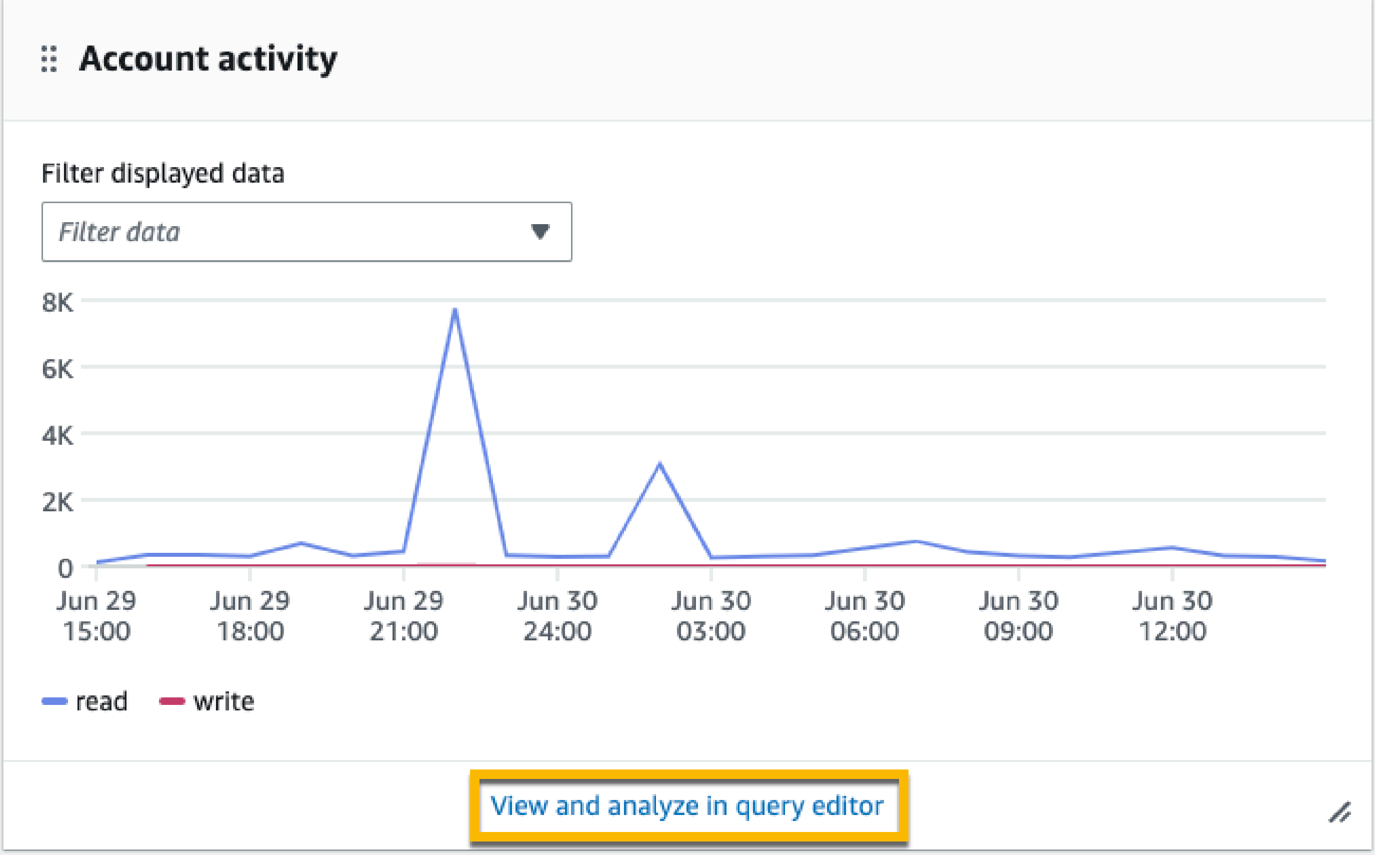
View and analyze in query editor

Top errors < 1 2 >

ReplicationConfigurationNotFoundError	34
ObjectLockConfigurationNotFoundError	34
NoSuchCORSConfiguration	34
NoSuchWebsiteConfiguration	34
NoSuchLifecycleConfiguration	32
NoSuchTagSet	32
QueryIdNotFoundException	24
NoSuchPublicAccessBlockConfiguration	10

View and analyze in query editor

7. To view the query for a widget, choose **View and analyze in query editor**.

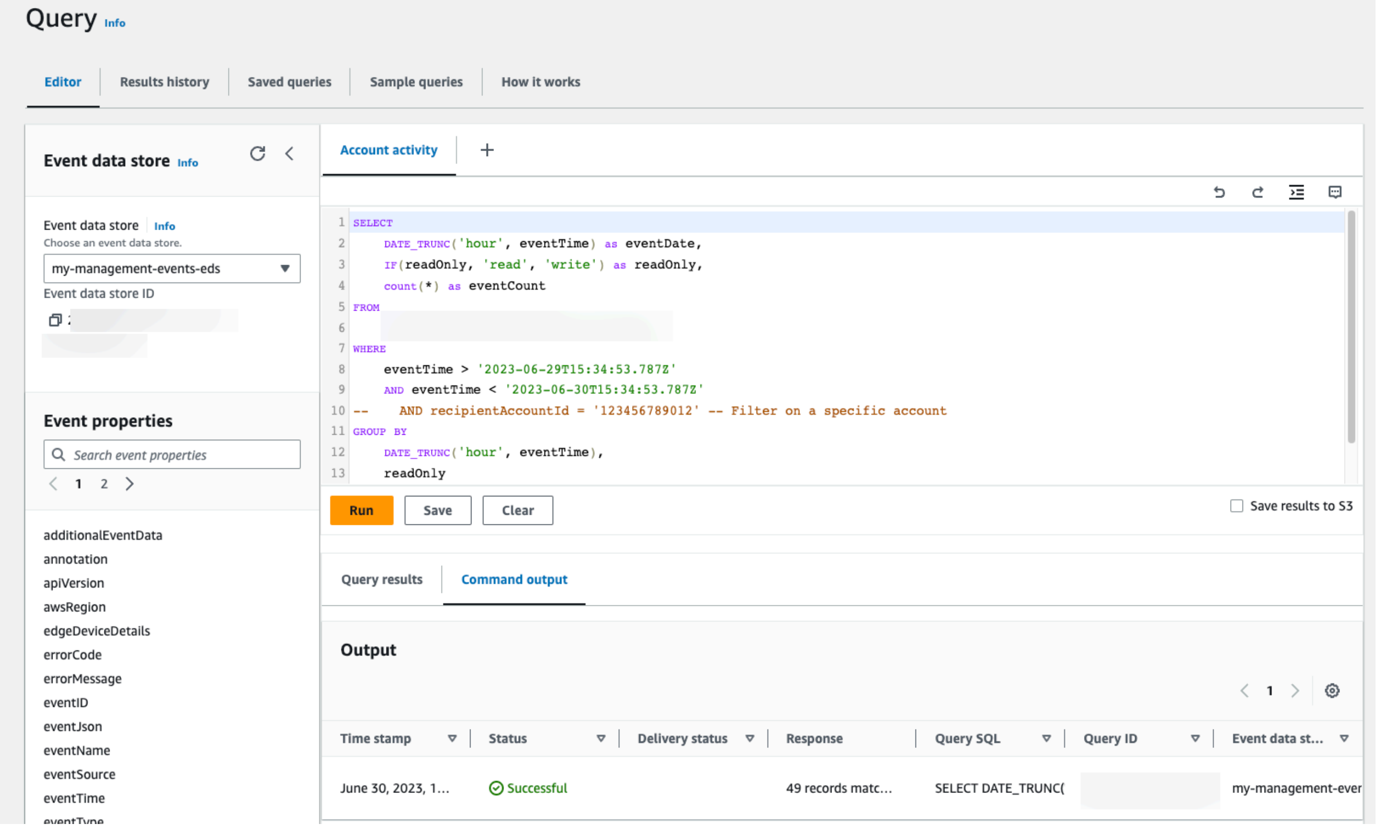


Account activity

Filter displayed data: Filter data ▼

View and analyze in query editor

Choosing **View and analyze in query editor** opens the query in CloudTrail Lake's query editor, which lets you further analyze the query results outside of the dashboard. For more information about editing a query, see Create or edit a query with the CloudTrail console. For more information about running a query and saving query results, see Run a query and save query results with the console.



Event data store Info
Choose an event data store.
my-management-events-eds

Event data store ID
[redacted]

Event properties

Search event properties

< 1 2 >

additionalEventData
annotation
apiVersion
awsRegion
edgeDeviceDetails
errorCode
errorMessage
eventID
eventJson
eventName
eventSource
eventTime
eventType

```
SELECT
    DATE_TRUNC('hour', eventTime) as eventDate,
    IF(readOnly, 'read', 'write') as readOnly,
    count(*) as eventCount
FROM
    [redacted]
WHERE
    eventTime > '2023-06-29T15:34:53.787Z'
    AND eventTime < '2023-06-30T15:34:53.787Z'
--    AND recipientAccountId = '123456789012' -- Filter on a specific account
GROUP BY
    DATE_TRUNC('hour', eventTime),
    readOnly
```

□ Save results to S3

Time stamp	Status	Delivery status	Response	Query SQL	Query ID	Event data st...
June 30, 2023, 1...	✓ Successful		49 records matc...	SELECT DATE_TRUNC(	[redacted]	my-management-ever

For more information about dashboards, see View CloudTrail Lake dashboards with the CloudTrail console.

Viewing your CloudTrail cost and usage with AWS Cost Explorer

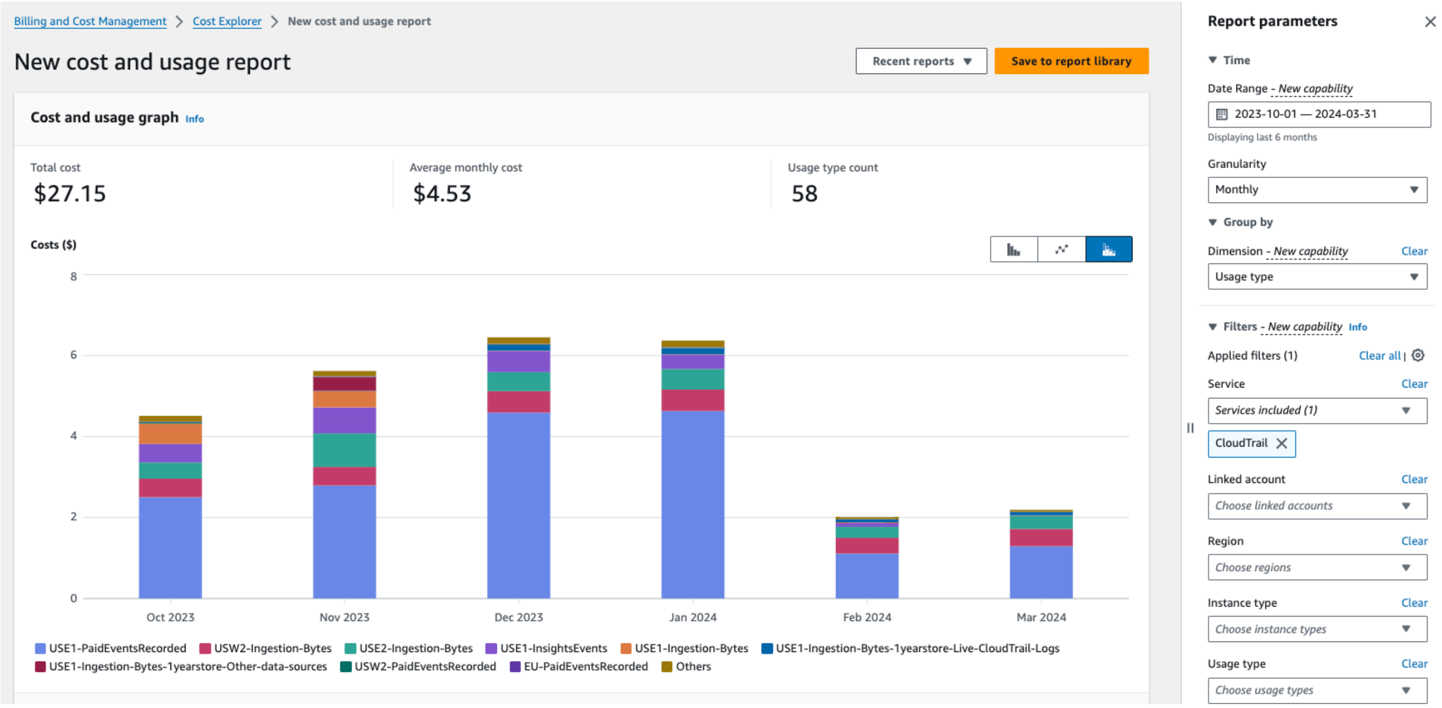
This section describes how you can view your CloudTrail costs and usage using AWS Cost Explorer. Cost Explorer gives you the ability to visualize, understand, and manage your AWS costs and usage over time.

For details about CloudTrail pricing, see AWS CloudTrail Pricing.

To view CloudTrail cost and usage with Cost Explorer

1. Sign in to the AWS Management Console and open the Cost Explorer console at https://console.aws.amazon.com/cost-management/home#/custom.
2. Under **Time**, choose the date range you want to analyze.
3. Under **Group by**, for **Dimension**, choose **Usage type**.
4. Under **Filters**, for **Service**, choose **CloudTrail**.

The following image shows an example of a cost report filtered for CloudTrail and grouped by **Usage type**.



Billing and Cost Management > Cost Explorer > New cost and usage report

New cost and usage report

Cost and usage graph

Total cost
$27.15

Average monthly cost
$4.53

Usage type count
58

Report parameters

Time

Date Range - New capability: 2023-10-01 — 2024-03-31
Displaying last 6 months

Granularity: Monthly

Group by

Dimension - New capability: Usage type

Filters - New capability

Applied filters (1)

Service: Services included (1)
CloudTrail

Linked account: Choose linked accounts

Region: Choose regions

Instance type: Choose instance types

Usage type: Choose usage types

Review the **Usage type** to see which CloudTrail features generated the most cost. Each **Usage type** begins with the code for the AWS Region where the charge was incurred.

The following table describes the CloudTrail usage types for each CloudTrail feature.

CloudTrail feature	Usage type	Description
CloudTrail trails	`*region*-FreeEventsRecorded`	The first copy of management events delivered free of charge to an AWS Region.
CloudTrail trails	`*region*-PaidEventsRecorded`	The charge for additional copies of management events delivered to an AWS Region.
CloudTrail trails	`*region*-DataEventsRecorded`	The charge for delivery of data events to an AWS Region. Data events always incur charges.
CloudTrail Lake	`*region*-Ingestion-Bytes`	The charge for ingesting events into a CloudTrail Lake event data store using the **Seven-year retention pricing** option. Ingestion pricing is based on the volume of

CloudTrail feature	Usage type	Description
		data ingested and is the same for all event types.
CloudTrail Lake	`*region*-Ingestion-Bytes-1yearstore-Live-CloudTrail-Logs`	The charge for ingesting CloudTrail data events and management events into a CloudTrail Lake event data store using the **One-year extendable retention pricing** option.

CloudTrail feature	Usage type	Description
CloudTrail Lake	`*region*-Ingestion-Bytes-1yearstore-Other-data-sources`	The charge for ingesting other event sources into a CloudTrail Lake event data store using the **One-year extendable retention pricing** option. This includes CloudTrail Insights events, configuration items from AWS Config, evidence from AWS Audit Manager, (uncompressed) historical CloudTrail logs imported from S3, and events outside of AWS.

CloudTrail feature	Usage type	Description
CloudTrail Lake	`region-QueryScanned-Bytes`	The charge for running CloudTrail Lake queries. When you run queries in CloudTrail Lake, you incur charges based on the amount of optimized and compressed data scanned.
CloudTrail Insights	`region-InsightsEvents`	The charge for CloudTrail Insights events. For Insights events, you incur charges based on the number of management events analyzed per Insight type.

Using AWS Budgets to manage costs

AWS Budgets a feature of AWS Billing and Cost Management, allows you to set custom budgets that alert you when your costs or usage exceed (or are forecasted to exceed) your budgeted amount.

Creating a budget for CloudTrail by using AWS Budgets is a recommended best practice, and can help you track your CloudTrail spending. Cost-based budgets help promote awareness of how much you might be billed for your CloudTrail use. Budget alerts notify you when your bill reaches a threshold that you define. When you receive a budget alert, you can make changes before the end of the billing cycle to manage your costs.

> **Note**
>
> Though you can apply tags to CloudTrail trails, AWS Billing cannot currently use tags applied to trails for cost allocation. Cost Explorer can show costs for CloudTrail Lake event data stores and for the CloudTrail service as a whole.

To get started with AWS Budgets, open AWS Billing and Cost Management, and then choose **Budgets** in the left navigation bar. We recommend configuring budget alerts as you create a budget to track CloudTrail spending. For more information about how to use AWS Budgets, see Managing your costs with AWS Budgets and Best practices for AWS Budgets.

Creating user-defined cost allocation tags for CloudTrail Lake event data stores

You can create user-defined cost allocation tags to track the query and ingestion costs for your CloudTrail Lake event data stores. A *user-defined cost allocation tag* is a key-value pair that you can associate with an event data store. After you activate cost allocation tags, AWS uses the tags to organize your resource costs on your cost allocation report.

- To create tags in the console, see step 9 of the To create an event data store for CloudTrail management or data events procedure.
- To create tags using the CloudTrail API, see CreateEventDataStore and AddTags in the *AWS CloudTrail API Reference*.
- To create tags using the AWS CLI, see create-event-data-store and add-tags in the *AWS CLI Command Reference*.

For more information about activating tags, see Activating user-defined cost allocation tags.

Managing CloudTrail trail costs

You can configure and manage CloudTrail trails in ways that capture the data you need while remaining cost-effective. For more information about CloudTrail pricing, see AWS CloudTrail Pricing.

Trail configuration

CloudTrail offers flexibility in how you configure trails in your account. Some decisions that you make during the setup process require that you understand the impacts to your CloudTrail bill. The following are examples of how trail configurations can influence your CloudTrail bill.

Multiple trail creation

The first copy of management events within each region is delivered free of charge. For example, if your account has 2 single-Region trails, a trail in `us-east-1` and another trail in `us-west-2`, there are no CloudTrail charges because there is only one trail logging events in each respective Region. However, if your account has a multi-Region trail and an additional single-Region trail, the single-Region trail will incur charges because the multi-Region trail is already logging events in each Region.

If you create more trails that deliver the same management events to other destinations, those subsequent deliveries incur CloudTrail costs. You can do this to allow different user groups (such as developers, security personnel, and IT auditors) to receive their own copies of log files. For data events, all deliveries incur CloudTrail costs, including the first.

As you create more trails, it is especially important to be familiar with your logs, and understand the types and volumes of events that are generated by resources in your account. This helps you anticipate the volume of events that are associated with an account, and plan for trail costs. For example, using AWS KMS-managed server-side encryption (SSE-KMS) on your S3 buckets can result in a large number of AWS KMS management events in CloudTrail. Larger volumes of events across multiple trails can also influence costs.

To help limit the number of events that are logged to your trail, you can filter out AWS KMS or Amazon RDS Data API events by choosing **Exclude AWS KMS events** or **Exclude Amazon RDS Data API events** on the **Create trail** or **Update trail** pages. When using basic event selectors, you can only filter management events. However, you can use advanced event selectors to filter both management and data events.

You can use advanced event selectors to include or exclude data events based on the `eventName`, `resources.ARN`, and `readOnly` fields, giving you the ability to log only the data events of interest. For more information, see Filtering data events by using advanced event selectors.

For more information about creating and updating a trail, see Creating a trail with the CloudTrail console or Updating a trail with the CloudTrail console in this guide.

AWS Organizations

When you set up an Organizations trail with CloudTrail, CloudTrail replicates the trail to each member account within your organization. The new trail is created *in addition to* any existing trails in member accounts. Be sure that the configuration of your organization trail matches how you want trails configured for all accounts within an organization, because the organization trail configuration propagates to all accounts.

Because Organizations creates a trail in each member account, an individual member account that creates an additional trail to collect the same management events as the Organizations trail is collecting a second copy of events. The account is charged for the second copy. Similarly, if an account has a multi-Region trail, and creates a second trail in a single Region to collect the same management events as the multi-Region trail, the trail in the single Region is delivering a second copy of events. The second copy incurs charges.

See also

- AWS CloudTrail Pricing
- Managing your costs with AWS Budgets
- Getting started with Cost Explorer
- Prepare for creating a trail for your organization

Managing CloudTrail Lake costs

AWS CloudTrail Lake event data stores and queries incur charges. You can configure event data stores in ways that capture the data you need while remaining cost-effective. For information about CloudTrail pricing, see AWS CloudTrail Pricing.

Topics

- Event data store pricing options
- Understanding CloudTrail Lake charges
- Recommendations for how you can reduce costs
- See also

Event data store pricing options

When you create an event data store, you choose the pricing option that you want to use for the event data store. The pricing option determines the cost for ingesting and storing events, and the default and maximum retention periods for the event data store.

The following table describes the available pricing options. The table shows the **Pricing option** in the console and the corresponding `BillingMode` value for the API, and lists the default and maximum retention period for each option.

Pricing option (console)	BillingMode (API)	Description
One-year extendable retention pricing	`EXTENDABLE_RETENTION_PRICING`	Recommended if you expect to ingest less than 25 TB of event data per month and want a flexible retention period of up to 10 years. This option is also recommended if your event data store collects AWS Config configuration items, Audit Manager evidence, and events from outside of AWS. For the first 366 days (the default retention period), storage is included at no additional cost with ingestion pricing. After 366 days, extended retention is available at pay-as-you-go pricing. This is the default option. **Default retention period:** 366 days **Maximum retention period:** 3,653 days
Seven-year retention pricing	`FIXED_RETENTION_PRICING`	Recommended if expect to ingest more than 25 TB of event data per month and need a retention period of up to 7 years.

Pricing option (console)	BillingMode (API)	Description
		Retention is included with ingestion pricing at no additional charge. **Default retention period:** 2,557 days **Maximum retention period:** 2,557 days

Understanding CloudTrail Lake charges

The following tables provides information about how CloudTrail Lake event data stores and queries incur charges. For information about CloudTrail pricing, see AWS CloudTrail Pricing.

Charge type	How you incur charges
Data ingestion (uncompressed data)	For CloudTrail Lake, you pay based on the uncompressed data ingested. The pricing option for the event data store determine s the cost of ingesting events: • **One-year extendable retention pricing**: Offers ingestion pricing based on event type. • **Seven-year retention pricing**: Offers ingestion pricing based on the volume of data ingested. The greatest savings are achieved when the volume of data ingested monthly exceeds 25 TB. **Copying trail events** When you copy trail events to CloudTrail Lake, CloudTrail unzips the logs that are stored in gzip (compressed) format. Then CloudTrail copies the events contained in the logs to your event data store. The size of the uncompressed data could be greater than the actual Amazon S3 storage size. To

Charge type	How you incur charges
	get a general estimate of the size of the uncompressed data, multiply the size of the logs in the S3 bucket by 10. **Note** CloudTrail will not copy an event if its event time is older than the specified retention period. To determine the appropriate retention period, take the sum of the oldest event you want to copy in days and the number of days you want to retain the events in the event data store as demonstrated in this equation: **Retention period** = *`oldest-event-in-days`* + *`number-days-to-retain`* For example, if the oldest event you're copying is 45 days old and you want to keep the events in the event data store for a further 45 days, you would set the retention period to 90 days.

Charge type	How you incur charges
Data retention (optimized and compressed data)	CloudTrail Lake converts existing events in row-based JSON format to Apache ORC format. ORC is a columnar storage format that is optimized for fast retrieval of compressed data. An event data store's *retention period* determines how long event data is kept in the event data store. CloudTrail Lake determines whether to retain an event by checking if an event's event time is within the specified retention period. For example, if you specify a retention period of 90 days, CloudTrai l will remove events when their event time is older than 90 days. For event data stores using the **Seven-year retention pricing** option, storage is included with ingestion pricing at no additional charge. For event data stores using the **One-year extendable retention pricing** option, storage is included at no charge with ingestion pricing for the first 366 days (the default retention period). After 366 days, storage is offered at pay-as-you-pricing and is charged based on the optimized and compressed data in the event data store.
Running queries in CloudTrai l Lake (optimized and compressed data)	When you run queries in CloudTrail Lake, you pay based on the amount of optimized and compressed data scanned.

Recommendations for how you can reduce costs

This section provides recommendations for how you can reduce costs when working with CloudTrail Lake.

Choose a pricing option based on the type of events your event data store will collect and your expected monthly ingestion

When creating an event data store, choose a pricing option based on the type of events your event data store will collect and your expected monthly ingestion.

If you expect to ingest less than 25 TB of event data on a monthly basis and want a flexible retention period of up to 10 years, choose the **One-year extendable retention pricing** option. We also generally recommend this option for event data stores that collect AWS Config configuration items, Audit Manager evidence, and events from outside of AWS.

If you expect to ingest more than 25 TB of event data on a monthly basis and need a 7-year retention period, choose the **Seven-year retention pricing** option.

Evaluate your event data store's monthly ingestion over time

Evaluate the historical monthly ingestion of your event data store to see if there's a pricing option better suited to your needs.

If you have an existing event data store that uses the **Seven-year retention pricing** option and you ingest less than 25 TB of data on a monthly basis, consider updating the event data store to use **One-year extendable retention pricing**. For event data stores using the **Seven-year retention pricing** option, you can change the pricing option using the CloudTrail console, AWS CLI, or UpdateEventDataStore API operation.

If you have an existing event data store that uses the **One-year extendable retention pricing** option and you ingest more than 25 TB of event data on a monthly basis, consider whether **Seven-year retention pricing** would better suit your needs. To use the new pricing option, stop ingestion on your event data store and create a new event data store with the **Seven-year retention pricing** option.

Use advanced event selectors to filter out events that aren't of interest

When configuring an event data store for CloudTrail management or data events, filter out events that aren't of interest by using advanced event selectors.

If you're creating an event data store to collect management events, you can filter out AWS Key Management Service (AWS KMS) or Amazon Relational Database Service (Amazon RDS) Data API management events. Typically, AWS KMS actions such as `Encrypt`, `Decrypt`, and `GenerateDataKey` generate more than 99 percent of events.

If you're creating an event data store to collect data events, you can use advanced event selectors to filter on the `eventName`, `resources.type`, `resources.ARN`, and `readOnly` fields. For more information, see Filtering data events by using advanced event selectors.

Choose a narrower time range when copying trail events

When copying trail events to CloudTrail Lake, specify a narrower start event time and end event time to reduce the amount of data ingested.

If you are copying trail events to CloudTrail Lake for historical analysis and do not want to ingest future events, deselect the option to ingest events so that you do not incur charges on ingesting any additional events.

Format queries to use a starting and ending `eventTime`

When you run queries in Lake, you pay based upon the amount of data scanned. You can constrain costs by specifying a starting and ending `eventTime` for the query.

See also

- AWS CloudTrail Pricing
- Supported CloudWatch metrics
- Managing your costs with AWS Budgets
- Getting started with Cost Explorer

Working with CloudTrail Event history

CloudTrail is enabled by default for your AWS account and you automatically have access to the CloudTrail **Event history**. The **Event history** provides a viewable, searchable, downloadable, and immutable record of the past 90 days of management events in an AWS Region. These events capture activity made through the AWS Management Console, AWS Command Line Interface, and AWS SDKs and APIs. The **Event history** records events in the AWS Region where the event happened. There are no CloudTrail charges for viewing the **Event history**.

You can look up events related to the creation, modification, or deletion of resources (such as IAM users or Amazon EC2 instances) in your AWS account on a by-Region basis in the CloudTrail console by viewing the **Event history** page. You can also look up these events by running the **aws cloudtrail lookup-events** command or by using the `LookupEvents` API.

You can use the **Event history** page in the CloudTrail console to view, search, download, archive, analyze, and respond to account activity across your AWS infrastructure. You can customize the view of the **Event history** in the console by selecting how many events to display on each page and which columns to display or hide. You can also compare the details of events in **Event history** side-by-side. You can programmatically look up events by using the AWS SDKs or AWS Command Line Interface.

Note

Over time, AWS services might add additional events. CloudTrail records these events in **Event history**, but a full 90-day record of activity that includes added events won't be available until 90 days after it adds the events.
The **Event history** is separate from any trails or event data stores that you create for your account. Changes you make to your event data stores or trails do not affect the **Event history**.

The sections which follow describe how to look up recent management events by using the CloudTrail console and the AWS CLI, and describe how to download a file of events. For information about using the `LookupEvents` API to retrieve information from CloudTrail events, see LookupEvents in the *AWS CloudTrail API Reference*.

Topics

- Limitations of Event history
- Viewing recent management events with the console
- Viewing recent management events with the AWS CLI

Limitations of Event history

The following limitations apply to the **Event history**.

- The **Event history** page on the CloudTrail console only shows management events. It does not show data events or Insights events.
- The **Event history** is limited to the past 90 days of events. For an ongoing record of events in your AWS account, create an event data store or a trail.
- When you download events from the **Event history** page on the CloudTrail console, you can download up to 200,000 events in a single file. If you reach the 200,000 event limit, the CloudTrail console will provide the option to download additional files.
- The **Event history** doesn't provide organization level event aggregation. To record events across your organization, create an organization event data store or trail.
- An **Event history** search is limited to a single AWS account, only returns events from a single AWS Region, and cannot query multiple attributes. You can only apply one attribute filter and a time range filter.

  You can create a CloudTrail Lake event data store to query across multiple attributes and AWS Regions. You can also query across multiple AWS accounts in an AWS Organizations organization. In CloudTrail Lake, you can query multiple event types, including management events, data events, Insights events, AWS Config configuration items, Audit Manager evidence, and non-AWS events. CloudTrail Lake queries offer a deeper and more customizable view of events than simple key and value lookups in **Event history**, or running `LookupEvents`. For more information, see Working with AWS CloudTrail Lake and Create an event data store for CloudTrail events with the console.
- You cannot exclude AWS KMS or Amazon RDS Data API events from **Event history**; settings that you apply to a trail or event data store do not apply to **Event history**.

Viewing recent management events with the console

You can use the **Event history** page in the CloudTrail console to view the last 90 days of management events in an AWS Region. You can also download a file with that information, or a subset of information based on the filter and time range you choose. You can customize your view of **Event history** by selecting how many events to display on each page and choosing which columns to display in the console. You can also look up and filter events by the resource types available for a particular service. You can select up to five events in **Event history** and compare their details side-by-side.

Event history does not show data events. To view data events, create an event data store or a trail.

After 90 days, events are no longer shown in **Event history**. You cannot manually delete events from **Event history**.

You can learn more about the specifics of how CloudTrail logs events for a specific service by consulting the documentation for that service. For more information, see AWS service topics for CloudTrail.

> **Note**
>
> For an ongoing record of activity and events past 90 days, create an event data store or a trail.

To view Event history

1. Sign in to the AWS Management Console and open the CloudTrail console at https://console.aws.amazon.com/cloudtrail/.
2. In the navigation pane, choose **Event history**. You see a filtered list of events, with the most recent events showing first. The default filter for events is **Read only**, set to **false**. You can clear that filter by choosing **X** at the right of the filter.
3. You can filter events on a single attribute, which you can choose from the drop-down list. To filter on an attribute, choose the attribute from the drop-down list and enter the full value for the attribute. For example, to view all console login events, choose the **Event name** filter, and specify **ConsoleLogin**. Or, to view recent S3 management events, choose the **Event source** filter, and specify `s3.amazonaws.com`.

4. To view a specific management event, choose the event name. On the event details page, you can view details about the event, see any referenced resources, and view the event record.
5. To compare events, select up to five events by filling their check boxes in the left margin of the **Event history** table. You can view details for the selected events side-by-side in the **Compare event details** table.
6. You can save event history by downloading it as a file in CSV or JSON format. Downloading your event history can take a few minutes.

Contents

Navigating between pages

You can navigate between pages in the **Event history** by choosing the page you want to view. You can also view the next and previous page in **Event history**.

Choose **<** to view the previous page of **Event history**.

Choose **>** to view the next page of **Event history**.

Customizing the display

You can customize the view of **Event history** in the CloudTrail console by selecting from the following preferences.

- **Page size** - Choose whether you want to display 10, 25, or 50 events on each page.
- **Wrap lines** - Wrap text so you can see all text for each event.
- **Striped rows** - Shade every other row in the table.
- **Event time display** - Choose whether to display the event time in UTC or the local time zone.

- **Select visible columns** - Select which columns to display. By default, the following columns are displayed:
  - **Event name**
  - **Event time**
  - **User name**
  - **Event source**
  - **Resource type**
  - **Resource name**

  > **Note**
  >
  > You cannot change the order of the columns, or manually delete events from **Event history**.

To customize the display

1. Sign in to the AWS Management Console and open the CloudTrail console at https://console.aws.amazon.com/cloudtrail/.
2. In the navigation pane, choose **Event history**.
3. Choose the gear icon.
4. For **Page size**, choose the number of events to display on a page.
5. Choose **Wrap lines** to see all text for each event.
6. Choose **Striped rows** to shade every other row in the table.
7. For **Event time display**, choose whether to display the event time in UTC or the local time zone. By default, UTC is selected.
8. In **Select visible columns**, select the columns you want to display. Turn off columns you do not want to display.
9. When you have finished making your changes, choose **Confirm**.

Filtering CloudTrail events

The default display of events in **Event history** uses an attribute filter to exclude read-only events from the list of displayed events. This attribute filter is named **Read-only**, and it is set to **false**.

You can remove this filter to display both read and write events. To view only **Read** events, you can change the filter value to **true**. You can also filter events by other attributes. You can additionally filter by time range.

> **Note**
>
> You can only apply one attribute filter and a time range filter. You cannot apply multiple attribute filters.

AWS access key

The AWS access key ID that was used to sign the request. If the request was made with temporary security credentials, this is the access key ID of the temporary credentials.

Event ID

The CloudTrail ID of the event. Each event has a unique ID.

Event name

The name of the event. For example, you can filter on IAM events, such as `CreatePolicy`, or Amazon EC2 events, such as `RunInstances`.

Event source

The AWS service to which the request was made, such as `iam.amazonaws.com` or `s3.amazonaws.com`. You can scroll through a list of event sources after you choose the **Event source** filter.

Read only

The read type of the event. Events are categorized as read events or write events. If set to **false**, read events are not included in the list of displayed events. By default, this attribute filter is applied and the value is set to **false**.

Resource name

The name or ID of the resource referenced by the event. For example, the resource name might be "auto-scaling-test-group" for an Auto Scaling group or "i-12345678910" for an EC2 instance.

Resource type

The type of resource referenced by the event. For example, a resource type can be `Instance` for EC2 or `DBInstance` for RDS. Resource types vary for each AWS service.

Time range

The time range in which you want to filter events. You can choose either a **Relative range** or an **Absolute range**. You can filter events for the last 90 days.

User name

The identity referenced by the event. For example, this can be a user, a role name, or a service role.

If there are no events logged for the attribute or time that you choose, the results list is empty. You can apply only one attribute filter in addition to the time range. If you choose a different attribute filter, your specified time range is preserved.

The following steps describe how to filter by attribute.

To filter by attribute

1. To filter the results by an attribute, choose an attribute from the **Lookup attributes** drop-down list, and then type or choose a value for the attribute in the text box.
2. To remove an attribute filter, choose the **X** at the right of the attribute filter box.

The following steps describe how to filter by a start and end date and time.

To filter by a start and end date and time

1. To narrow the time range for the events that you want to see, choose a time range in the time range bar. You can choose either a **Relative range** or an **Absolute range**.

   Choose **Relative range** to select from a preset value or choose a custom range. Preset values are 30 minutes, 1 hour, 12 hours, or 1 day. To specify a custom time range, choose **Custom**.

   Choose **Absolute range** to specify a specific start and end time. You can also choose between the local time zone or UTC.
2. To remove a time range filter, choose **Clear and dismiss** in the time range bar.

Viewing details for an event

1. Choose an event in the results list to show its details.

2. Resources referenced in the event are shown in the **Resources referenced** table on the event details page.
3. Some referenced resources have links. Choose the link to open the console for that resource.
4. Scroll to **Event record** on the details page to see the JSON event record, also called the event *payload*.
5. Choose **Event history** in the page breadcrumb to close the event details page and return to **Event history**.

Downloading events

You can download recorded event history as a file in CSV or JSON format. You can download up to 200,000 events in a single file. If you reach the 200,000 event limit, the CloudTrail console will provide the option to download additional files. Use filters and time ranges to reduce the size of the file you download.

> **Note**
>
> CloudTrail event history files are data files that contain information (such as resource names) that can be configured by individual users. Some data can potentially be interpreted as commands in programs used to read and analyze this data (CSV injection). For example, when CloudTrail events are exported to CSV and imported to a spreadsheet program, that program might warn you about security concerns. You should choose to disable this content to keep your system secure. Always disable links or macros from downloaded event history files.

1. Add a filter and time range for events in **Event history** that you want to download. For example, you can specify the event name, `StartInstances`, and specify a time range for the last three days of activity.
2. Choose **Download events**, and then choose **Download as CSV** or **Download as JSON**. The download starts immediately.

   > **Note**
   >
   > Your download might take some time to complete. For faster results, before you start the download process, use a more specific filter or a shorter time range to narrow

> the results. You can cancel a download. If you cancel a download, a partial download including only some event data might be on your local computer. To download the full event history, restart the download.

3. After your download is complete, open the file to view the events that you specified.
4. To cancel your download, choose **Cancel**, and then confirm by choosing **Cancel download**. If you need to restart a download, wait until the earlier download is finished canceling.

Viewing resources referenced with AWS Config

AWS Config records configuration details, relationships, and changes to your AWS resources.

On the **Resources referenced** pane, choose the

←🕒

in the **AWS Config resource timeline** column to view the resource in the AWS Config console.

If the

←🕒

icon is gray, AWS Config isn't turned on, or it's not recording the resource type. Choose the icon to go to the AWS Config console to turn on the service or start recording that resource type. For more information, see Set Up AWS Config Using the Console in the *AWS Config Developer Guide*.

If **Link not available** appears in the column, the resource can't be viewed for one of the following reasons:

- AWS Config doesn't support the resource type. For more information, see Supported Resources, Configuration Items, and Relationships in the *AWS Config Developer Guide*.
- AWS Config recently added support for the resource type, but it's not yet available from the CloudTrail console. You can look up the resource in the AWS Config console to see the timeline for the resource.
- The resource is owned by another AWS account.
- The resource is owned by another AWS service, such as a managed IAM policy.
- The resource was created and then deleted immediately.
- The resource was recently created or updated.

To grant users read-only permission to view resources in the AWS Config console, see Granting permission to view AWS Config information on the CloudTrail console.

For more information about AWS Config, see the AWS Config Developer Guide.

Viewing recent management events with the AWS CLI

You can look up CloudTrail management events for the last 90 days for the current AWS Region using the **aws cloudtrail lookup-events** command. The **aws cloudtrail lookup-events** command shows events in the AWS Region where they occurred.

Lookup supports the following attributes for management events:

- AWS access key
- Event ID
- Event name
- Event source
- Read only
- Resource name
- Resource type
- User name

All attributes are optional.

The `lookup-events` command includes the following options:

- `--max-items` *`<integer>`* – The total number of items to return in the command's output. If the total number of items available is more than the value specified, a `NextToken` is provided in the command's output. To resume pagination, provide the `NextToken` value in the starting-token argument of a sub- sequent command. Do not use the `NextToken` response element directly outside of the AWS CLI.
- `--start-time` *`<timestamp>`* – Specifies that only events that occur after or at the specified time are returned. If the specified start time is after the specified end time, an error is returned.
- `--lookup-attributes` *`<integer>`* – Contains a list of lookup attributes. Currently the list can contain only one item.

- `--generate-cli-skeleton` *<string>* – Prints a JSON skeleton to standard output without sending an API request. If provided with no value or the value input, prints a sample input JSON that can be used as an argument for `--cli-input-json`. Similarly, if provided yaml-input it will print a sample input YAML that can be used with `--cli-input-yaml`. If provided with the value output, it validates the command inputs and returns a sample output JSON for that command. The generated JSON skeleton is not stable between versions of the AWS CLI and there are no backwards compatibility guarantees in the JSON skeleton generated.
- `--cli-input-json` *<string>* – Reads arguments from the JSON string provided. The JSON string follows the format provided by the `--generate-cli-skeleton` parameter. If other arguments are provided on the command line, those values will override the JSON-provided values. It is not possible to pass arbitrary binary values using a JSON-provided value as the string will be taken literally. This may not be specified along with the `--cli-input-yaml` parameter.

For general information about using the AWS Command Line Interface, see the AWS Command Line Interface User Guide.

Contents

Prerequisites

- To run AWS CLI commands, you must install the AWS CLI. For information, see Get started with the AWS CLI.
- Make sure your AWS CLI version is greater than 1.6.6. To verify the CLI version, run **aws --version** on the command line.

- To set the account, AWS Region, and default output format for an AWS CLI session, use the **aws configure** command. For more information, see Configuring the AWS Command Line Interface.

> **Note**
>
> The CloudTrail AWS CLI commands are case-sensitive.

Getting command line help

To see the command line help for `lookup-events`, type the following command:

```
aws cloudtrail lookup-events help
```

Looking up events

> **Important**
>
> The rate of lookup requests is limited to two per second, per account, per Region. If this limit is exceeded, a throttling error occurs.

To see the ten latest events, type the following command:

```
aws cloudtrail lookup-events --max-items 10
```

A returned event looks similar to the following fictitious example, which has been formatted for readability:

```
{
    "NextToken": "kbOt5LlZe+
+mErCebpy2TgaMgmDvF1kYGFcH64JSjIbZFjsuvrSqg66b5YGssKutDYIyII4lrP4IDbeQdiObkp9YAlju3oXd12juy3CIZ
    "Events": [
        {
            "EventId": "0ebbaee4-6e67-431d-8225-ba0d81df5972",
            "Username": "root",
            "EventTime": 1424476529.0,
            "CloudTrailEvent": "{
                  \"eventVersion\":\"1.02\",
                  \"userIdentity\":{
                        \"type\":\"Root\",
                        \"principalId\":\"111122223333\",
                        \"arn\":\"arn:aws:iam::111122223333:root\",
                        \"accountId\":\"111122223333\"},
                  \"eventTime\":\"2015-02-20T23:55:29Z\",
                  \"eventSource\":\"signin.amazonaws.com\",
                  \"eventName\":\"ConsoleLogin\",
                  \"awsRegion\":\"us-east-2\",
                  \"sourceIPAddress\":\"203.0.113.4\",
                  \"userAgent\":\"Mozilla/5.0\",
                  \"requestParameters\":null,
                  \"responseElements\":{\"ConsoleLogin\":\"Success\"},
                  \"additionalEventData\":{
                         \"MobileVersion\":\"No\",
                         \"LoginTo\":\"https://console.aws.amazon.com/console/home",
                         \"MFAUsed\":\"No\"},
                  \"eventID\":\"0ebbaee4-6e67-431d-8225-ba0d81df5972\",
                  \"eventType\":\"AwsApiCall\",
                  \"recipientAccountId\":\"111122223333\"}",
            "EventName": "ConsoleLogin",
            "Resources": []
        }
    ]
}
```

For an explanation of the lookup-related fields in the output, see the section Lookup output fields later in this document. For an explanation of the fields in the CloudTrail event, see CloudTrail record contents.

Specifying the number of events to return

To specify the number of events to return, type the following command:

```
aws cloudtrail lookup-events --max-items <integer>
```

Possible values are 1 through 50. The following example returns one event.

```
aws cloudtrail lookup-events --max-items 1
```

Looking up events by time range

Events from the past 90 days are available for lookup. To specify a time range, type the following command:

```
aws cloudtrail lookup-events --start-time <timestamp> --end-time <timestamp>
```

`--start-time` *`<timestamp>`* specifies, in UTC, that only events that occur after or at the specified time are returned. If the specified start time is after the specified end time, an error is returned.

`--end-time` *`<timestamp>`* specifies, in UTC, that only events that occur before or at the specified time are returned. If the specified end time is before the specified start time, an error is returned.

The default start time is the earliest date that data is available within the last 90 days. The default end time is the time of the event that occurred closest to the current time.

All timestamps are shown in UTC.

Looking up events by attribute

To filter by an attribute, type the following command:

```
aws cloudtrail lookup-events --lookup-attributes
 AttributeKey=<attribute>,AttributeValue=<string>
```

You can specify only one attribute key/value pair for each **lookup-events** command. The following are valid values for `AttributeKey`. Value names are case sensitive.

- `AccessKeyId`
- `EventId`
- `EventName`
- `EventSource`
- `ReadOnly`
- `ResourceName`
- `ResourceType`

- Username

The maximum length for the AttributeValue is 2000 characters. The following characters ('_', ' ', ',', '\\n') count as two characters towards the 2000 character limit.

Attribute lookup examples

The following example command returns events in which the value of AccessKeyId is AKIAIOSFODNN7EXAMPLE.

```
aws cloudtrail lookup-events --lookup-attributes
 AttributeKey=AccessKeyId,AttributeValue=AKIAIOSFODNN7EXAMPLE
```

The following example command returns the event for the specified CloudTrail EventId.

```
aws cloudtrail lookup-events --lookup-attributes
 AttributeKey=EventId,AttributeValue=b5cc8c40-12ba-4d08-a8d9-2bceb9a3e002
```

The following example command returns events in which the value of EventName is RunInstances.

```
aws cloudtrail lookup-events --lookup-attributes
 AttributeKey=EventName,AttributeValue=RunInstances
```

The following example command returns events in which the value of EventSource is iam.amazonaws.com.

```
aws cloudtrail lookup-events --lookup-attributes
 AttributeKey=EventSource,AttributeValue=iam.amazonaws.com
```

The following example command returns write events. It excludes read events such as GetBucketLocation and DescribeStream.

```
aws cloudtrail lookup-events --lookup-attributes
 AttributeKey=ReadOnly,AttributeValue=false
```

The following example command returns events in which the value of ResourceName is CloudTrail_CloudWatchLogs_Role.

```
aws cloudtrail lookup-events --lookup-attributes
 AttributeKey=ResourceName,AttributeValue=CloudTrail_CloudWatchLogs_Role
```

The following example command returns events in which the value of `ResourceType` is `AWS::S3::Bucket`.

```
aws cloudtrail lookup-events --lookup-attributes
 AttributeKey=ResourceType,AttributeValue=AWS::S3::Bucket
```

The following example command returns events in which the value of `Username` is `root`.

```
aws cloudtrail lookup-events --lookup-attributes
 AttributeKey=Username,AttributeValue=root
```

Specifying the next page of results

To get the next page of results from a `lookup-events` command, type the following command:

```
aws cloudtrail lookup-events <same parameters as previous command> --next-token=<token>
```

where the value for *<token>* is taken from the first field of the output of the previous command.

When you use `--next-token` in a command, you must use the same parameters as in the previous command. For example, suppose you run the following command:

```
aws cloudtrail lookup-events --lookup-attributes
 AttributeKey=Username,AttributeValue=root
```

To get the next page of results, your next command would look like this:

```
aws cloudtrail lookup-events --lookup-attributes
 AttributeKey=Username,AttributeValue=root --next-token=kbOt5LlZe+
+mErCebpy2TgaMgmDvF1kYGFcH64JSjIbZFjsuvrSqg66b5YGssKutDYIyII4lrP4IDbeQdiObkp9YAlju3oXd12juy3CIZ
```

Getting JSON input from a file

The AWS CLI for some AWS services has two parameters, `--generate-cli-skeleton` and `--cli-input-json`, that you can use to generate a JSON template which you can modify and use as input to the `--cli-input-json` parameter. This section describes how to use these parameters with `aws cloudtrail lookup-events`. For more general information, see [AWS CLI skeletons and input files](#).

To look up CloudTrail events by getting JSON input from a file

1. Create an input template for use with `lookup-events` by redirecting the `--generate-cli-skeleton` output to a file, as in the following example.

   ```
   aws cloudtrail lookup-events --generate-cli-skeleton > LookupEvents.txt
   ```

   The template file generated (in this case, LookupEvents.txt) looks like this:

   ```

   {
       "LookupAttributes": [
           {
               "AttributeKey": "",
               "AttributeValue": ""
           }
       ],
       "StartTime": null,
       "EndTime": null,
       "MaxResults": 0,
       "NextToken": ""
   }
   ```

2. Use a text editor to modify the JSON as needed. The JSON input must contain only values that are specified.

   > **Important**
   >
   > All empty or null values must be removed from the template before you can use it.

   The following example specifies a time range and maximum number of results to return.

   ```
   {
       "StartTime": "2023-11-01",
       "EndTime": "2023-12-12",
       "MaxResults": 10
   ```

```
}
```

3. To use the edited file as input, use the syntax `--cli-input-json file://<filename>`, as in the following example:

```
aws cloudtrail lookup-events --cli-input-json file://LookupEvents.txt
```

> **Note**
>
> You can use other arguments on the same command line as `--cli-input-json` .

Lookup output fields

Events

A list of lookup events based on the lookup attribute and time range that were specified. The events list is sorted by time, with the latest event listed first. Each entry contains information about the lookup request and includes a string representation of the CloudTrail event that was retrieved.

The following entries describe the fields in each lookup event.

CloudTrailEvent

A JSON string that contains an object representation of the event returned. For information about each of the elements returned, see Record Body Contents.

EventId

A string that contains the GUID of the event returned.

EventName

A string that contains the name of the event returned.

EventSource

The AWS service that the request was made to.

EventTime

The date and time, in UNIX time format, of the event.

Resources

A list of resources referenced by the event that was returned. Each resource entry specifies a resource type and a resource name.

ResourceName

A string that contains the name of the resource referenced by the event.

ResourceType

A string that contains the type of a resource referenced by the event. When the resource type cannot be determined, null is returned.

Username

A string that contains the user name of the account for the event returned.

NextToken

A string to get the next page of results from a previous `lookup-events` command. To use the token, the parameters must be the same as those in the original command. If no `NextToken` entry appears in the output, there are no more results to return.

Working with AWS CloudTrail Lake

AWS CloudTrail Lake lets you run SQL-based queries on your events. CloudTrail Lake converts existing events in row-based JSON format to Apache ORC format. ORC is a columnar storage format that is optimized for fast retrieval of data. Events are aggregated into event data stores, which are immutable collections of events based on criteria that you select by applying advanced event selectors. You can keep the event data in an event data store for up to 3,653 days (about 10 years) if you choose the **One-year extendable retention pricing** option, or up to 2,557 days (about 7 years) if you choose the **Seven-year retention pricing** option. The selectors that you apply to an event data store control which events persist and are available for you to query. CloudTrail Lake is an auditing solution that can complement your compliance stack, and assist you with near real-time troubleshooting.

CloudTrail Lake event data stores

When you create an event data store, you choose the type of events to include in your event data store. You can create an event data store to include CloudTrail events, CloudTrail Insights events, AWS Config configuration items, AWS Audit Manager evidence, or events from outside of AWS. Each event data store can only contain a specific event category (for example, AWS Config configuration items), because the event schema is unique to the event category. You can store events from an organization in AWS Organizations in an organization event data store, including events from multiple Regions and accounts. You can also run SQL queries across multiple event data stores using the supported SQL JOIN keywords. For information about running queries across multiple event data stores, see Advanced, multi-table query support.

You can copy trail events to a new or existing event data store to create a point-in-time snapshot of events logged to the trail. For more information, see Copy trail events to an event data store.

You can federate an event data store to see the metadata associated with the event data store in the AWS Glue Data Catalog and run SQL queries on the event data using Amazon Athena. The table metadata stored in the AWS Glue Data Catalog lets the Athena query engine know how to find, read, and process the data that you want to query. For more information, see Federate an event data store.

By default, all events in an event data store are encrypted by CloudTrail. When you configure an event data store, you can choose to use your own AWS Key Management Service key. Using your own KMS key incurs AWS KMS costs for encryption and decryption. After you associate an event data store with a KMS key, the KMS key cannot be removed or changed.

You can control access to actions on event data stores by using authorization based on tags. For more information and examples, see Examples: Denying access to create or delete event data stores based on tags in this guide.

You can use CloudTrail Lake dashboards to visualize the data in your event data stores. Each dashboard consists of multiple widgets and each widget represents a SQL query. For more information about Lake dashboards, see View CloudTrail Lake dashboards with the CloudTrail console.

CloudTrail Lake event data stores incur charges. When you create an event data store, you choose the pricing option you want to use for the event data store. The pricing option determines the cost for ingesting and storing events, and the default and maximum retention period for the event data store. For information about CloudTrail pricing and managing Lake costs, see AWS CloudTrail Pricing and Managing CloudTrail Lake costs.

CloudTrail Lake supports Amazon CloudWatch metrics, which provide information about data ingested and storage bytes. For more information about supported CloudWatch metrics, see Supported CloudWatch metrics.

Note

CloudTrail typically delivers events within an average of about 5 minutes of an API call. This time is not guaranteed.

CloudTrail Lake integrations

You can use CloudTrail Lake *integrations* to log and store user activity data from outside of AWS; from any source in your hybrid environments, such as in-house or SaaS applications hosted on-premises or in the cloud, virtual machines, or containers. After you create event data stores in CloudTrail Lake and create a channel to log activity events, you call the `PutAuditEvents` API to ingest your application activity into CloudTrail. You can then use CloudTrail Lake to search, query, and analyze the data that is logged from your applications.

Integrations can also log events to your event data stores from over a dozen CloudTrail partners. In a partner integration, you create destination event data stores, a channel, and a resource policy. After you create the integration, you provide the channel ARN to the partner. There are two types of integrations: direct and solution. With direct integrations, the partner calls the `PutAuditEvents` API to deliver events to the event data store for your AWS account. With solution integrations, the application runs in your AWS account and the application calls the `PutAuditEvents` API to deliver events to the event data store for your AWS account.

For more information about integrations, see Create an integration with an event source outside of AWS.

CloudTrail Lake queries

> **Note**
>
> Introducing a preview feature for CloudTrail Lake queries that uses generative artificial intelligence (generative AI) capabilities to produce a SQL query from an English language prompt. For more information, see Create CloudTrail Lake queries from English language prompts.

CloudTrail Lake queries offer a deeper and more customizable view of events than simple key and value lookups in **Event history**, or running `LookupEvents`. An **Event history** search is limited to a single AWS account, only returns events from a single AWS Region, and cannot query multiple attributes. In contrast, CloudTrail Lake users can run complex SQL queries across multiple event fields. CloudTrail Lake supports all valid Presto `SELECT` statements and functions. For more information about the supported SQL functions and operators, see Functions and Operators on the Presto documentation website.

You can save CloudTrail Lake queries for future use, and view results of queries for up to seven days. When you run queries, you can save the query results to an Amazon S3 bucket.

The CloudTrail console provides a number of sample queries that can help you get started writing your own queries. For more information, see View sample queries with the CloudTrail console.

CloudTrail Lake queries incur charges. When you run queries in Lake, you pay based upon the amount of data scanned. For information about CloudTrail pricing and managing Lake costs, see AWS CloudTrail Pricing and Managing CloudTrail Lake costs.

Additional resources

The following resources can help you get a better understanding of what CloudTrail Lake is and how you can use it.

- Modernize Your Audit Log Management Using CloudTrail Lake (YouTube video)
- Log Activity Events from Non-AWS Sources in AWS CloudTrail Lake (YouTube video)
- Analyze Activity Logs with AWS CloudTrail Lake and Amazon Athena (YouTube video)
- Get visibility into the activity logs for your workforce and customer identities (AWS blog)
- Using AWS CloudTrail Lake to identify older TLS connections to AWS service endpoints (AWS blog)
- How Arctic Wolf uses AWS CloudTrail Lake to Simplify Security and Operations (AWS blog)
- CloudTrail Lake FAQs
- AWS CloudTrail API Reference
- AWS CloudTrail Data API Reference
- AWS CloudTrail Partner Onboarding Guide

CloudTrail Lake supported Regions

Currently, CloudTrail Lake is supported in the following AWS Regions:

Region Name	Region
US East (N. Virginia)	us-east-1
US East (Ohio)	us-east-2
US West (N. California)	us-west-1
US West (Oregon)	us-west-2
Africa (Cape Town)	af-south-1
Asia Pacific (Hong Kong)	ap-east-1
Asia Pacific (Hyderabad)	ap-south-2

Region Name	Region
Asia Pacific (Jakarta)	ap-southeast-3
Asia Pacific (Melbourne)	ap-southeast-4
Asia Pacific (Mumbai)	ap-south-1
Asia Pacific (Osaka)	ap-northeast-3
Asia Pacific (Seoul)	ap-northeast-2
Asia Pacific (Singapore)	ap-southeast-1
Asia Pacific (Sydney)	ap-southeast-2
Asia Pacific (Tokyo)	ap-northeast-1
Canada (Central)	ca-central-1
Europe (Frankfurt)	eu-central-1
Europe (Ireland)	eu-west-1
Europe (London)	eu-west-2
Europe (Milan)	eu-south-1
Europe (Paris)	eu-west-3
Europe (Spain)	eu-south-2
Europe (Stockholm)	eu-north-1
Europe (Zurich)	eu-central-2
Israel (Tel Aviv)	il-central-1
Middle East (Bahrain)	me-south-1
Middle East (UAE)	me-central-1

Region Name	Region
South America (São Paulo)	sa-east-1
AWS GovCloud (US-East)	us-gov-east-1
AWS GovCloud (US-West)	us-gov-west-1

For information about CloudTrail service endpoints, see AWS CloudTrail endpoints and quotas.

For more information about using CloudTrail in the AWS GovCloud (US) Regions, see Service Endpoints in the *AWS GovCloud (US) User Guide*.

CloudTrail Lake concepts and terminology

This section describes the key concepts and terms to help you use AWS CloudTrail Lake.

Concepts and terms

- Event data stores
- Integrations
- Queries
- Dashboard

Event data stores

Events are aggregated into *event data stores*, which are immutable collections of events based on criteria that you select by applying advanced event selectors.

You can create an event data store to log CloudTrail management events and data events, CloudTrail Insights events, AWS Audit Manager evidence, AWS Config configuration items, or events outside of AWS.

Advanced event selectors

> *Advanced event selectors* determine which events to include in an event data store. Advanced event selectors help you control costs by logging only those events that are important to you.

For management events and data events, you can use advanced event selectors to filter events. For example, if you're creating an event data store to collect management events, you can filter out AWS Key Management Service (AWS KMS) or Amazon Relational Database Service (Amazon RDS) Data API events. Typically, AWS KMS actions such as `Encrypt`, `Decrypt`, and `GenerateDataKey` generate more than 99 percent of events.

For AWS Config configuration items, Audit Manager evidence, or events outside of AWS, advanced event selectors are used only to include events of that type in the event data store.

Federation

Federation lets you see the metadata associated with an event data store in the AWS Glue Data Catalog and run SQL queries on the event data using Amazon Athena. The table metadata stored in the AWS Glue Data Catalog lets the Athena query engine know how to find, read, and process the data that you want to query.

When you enable Lake query federation, CloudTrail creates the federated resources on your behalf and registers those resources with AWS Lake Formation. After Lake federation is enabled, you can directly query your event data in Athena without needing to perform any additional steps. For more information, see Federate an event data store.

Pricing option

When you create an event data store, you choose the *pricing option* that you want to use for the event data store. The pricing option determines the cost for ingesting and storing events, and the default and maximum retention periods for the event data store. For information about pricing, see AWS CloudTrail Pricing and Managing CloudTrail Lake costs.

Retention period

An event data store's *retention period* determines how long event data is kept in the event data store. CloudTrail Lake determines whether to retain an event by checking if the `eventTime` of the event is within the specified retention period. For example, if you specify a retention period of 90 days, CloudTrail will remove events when their `eventTime` is older than 90 days.

Default retention period

An event data store's *default retention period* is the default number of days that event data is kept in the event data store. During an event data store's default retention period, storage is included with ingestion pricing at no additional charge. After the default retention period, pricing for storage is pay-as-you-go.

Maximum retention period

An event data store's *maximum retention period* represents the maximum number of days that you can keep data in an event data store.

Termination protection

By default, event data stores enable *termination protection*, which protects an event data store from being accidentally deleted. To delete an event data store with termination protection enabled, choose **Change termination protection** from the **Actions** menu on the event data store's details page. Then you can proceed with deleting the event data store. For more information, see Change termination protection with the console.

Integrations

You can use CloudTrail Lake *integrations* to log and store user activity data from the following sources:

- Outside of AWS
- Any source in your hybrid environments, such as in-house or software as a service (SaaS) applications hosted on premises or in the cloud, virtual machines, or containers

An integration requires a channel to deliver the events and an event data store to receive the events. After you set up your integration, call the PutAuditEvents API operation to ingest your application activity into CloudTrail. Then, you can use CloudTrail Lake to search, query, and analyze the data that is logged from your applications. For more information, see Create an integration with an event source outside of AWS.

Integration type

There are two types of integrations: *direct* and *solution*. With direct integrations, the partner calls the `PutAuditEvents` API operation to deliver events to the event data store for your AWS account. With solution integrations, the application runs in your AWS account and the application calls the `PutAuditEvents` API operation to deliver events to the event data store for your AWS account.

Channels

Activity events from sources outside of AWS work by using *channels* to bring events into CloudTrail Lake from external partners that work with CloudTrail, or from your own sources. When you create a channel, you choose one or more event data stores to store events that arrive from the channel source. You can change the destination event data stores for a channel as needed, as long as the destination event data stores are set to log `eventCategory="ActivityAuditLog"` events. When you create a channel for events from an external partner, you provide a channel Amazon Resource Name (ARN) to the partner or source application.

Resource-based policies

Resource-based policies are JSON policy documents that you attach to a resource. The resource-based policy attached to the channel allows the source to transmit events through the channel. If a channel doesn't have a resource policy, only the channel owner can call the `PutAuditEvents` API operation on the channel. For more information, see AWS CloudTrail resource-based policy examples.

Queries

> **Note**
>
> Introducing a preview feature for CloudTrail Lake queries that uses generative artificial intelligence (generative AI) capabilities to produce a SQL query from an English language prompt. For more information, see Create CloudTrail Lake queries from English language prompts.

Queries in CloudTrail Lake are authored in SQL. You can build a query on the CloudTrail Lake **Editor** tab by writing the query in SQL from scratch, or by opening a saved or sample query and editing it. You can't overwrite an included sample query with your changes, but you can save it as a new query. For more information, see Create or edit a query with the CloudTrail console.

CloudTrail Lake supports all valid Presto `SELECT` statements and functions. For more information about the supported SQL functions and operators, see Functions and Operators on the Presto documentation website.

Dashboard

By using the CloudTrail Lake *dashboard*, you can visualize the events in an event data store and see events trends, such as top AWS services, users, and errors. For more information, see View CloudTrail Lake dashboards with the CloudTrail console.

Dashboard type

The dashboard types available for an event data store depend upon the advanced event selectors configuration of the event data store. For example, if a dashboard type displays information about CloudTrail management events, you can only select the dashboard if the currently selected event data store collects CloudTrail management events.

The following are the available dashboard types:

- **Overview** dashboard – Shows the most active users, AWS Regions, and AWS services by event count. You can also view information about `read` and `write` management event activity, most throttled events, and the top errors. This dashboard is available for event data stores that collect management events.
- **Management Events** dashboard – Shows console sign-in events, access denied events, destructive actions, and top errors by user. You can also view information about TLS versions and outdated TLS calls by user. This dashboard is available for event data stores that collect management events.
- **S3 Data Events** dashboard – Shows Amazon S3 account activity, most accessed S3 objects, top S3 users, and top S3 actions. This dashboard is available for event data stores that collect Amazon S3 data events.
- **Insights Events** dashboard - Shows the overall proportion of Insights events by Insights type, the proportion of Insights events by Insights type for the top users and services, and the number of Insights events per day. The dashboard also includes a widget that lists up to 30 days of Insights events. This dashboard is only available for event data stores that collect Insights events.

> **Note**
>
> - After you enable CloudTrail Insights for the first time on the source event data store, it can take up to 7 days for CloudTrail to deliver the first Insights event, if unusual activity is detected. For more information, see Understanding Insights events delivery.

- The **Insights Events** dashboard only displays information about the Insights events collected by the selected event data store, which is determined by the configuration of the source event data store. For example, if you configure the source event data store to enable Insights events on `ApiCallRateInsight` but not `ApiErrorRateInsight`, you won't see information about Insights events on `ApiErrorRateInsight`.

Widgets

Widgets are the components that make up a dashboard and provide a visualization, such as a line chart or bar graph. Each widget represents an underlying query. When you choose **Run queries**, CloudTrail runs a system-generated query to populate the data for each widget.

CloudTrail Lake event data stores

Events are aggregated into event data stores, which are immutable collections of events based on criteria that you select by applying advanced event selectors.

When you create an event data store in CloudTrail Lake, you choose the type of events to include in your event data store. You can create an event data store to include CloudTrail data or management events, CloudTrail Insights events, AWS Config configuration items, or events outside of AWS. Each event data store type can only contain specific event categories (for example, AWS Config configuration items), because the event schema is unique to the event category. You can run SQL queries across multiple event data stores using the supported SQL JOIN keywords. For information about running queries across multiple event data stores, see Advanced, multi-table query support.

The following table shows the supported event categories for each event data store type. The **eventCategory** column shows the value that you would specify in the advanced event selectors to collect events of that type.

Event type (console)	eventCategory (API)	Description
CloudTrail events	`Management` `Data`	This event data store type can collect CloudTrail management and data events. For more information, see Create an event data store for CloudTrail events.

Event type (console)	eventCategory (API)	Description
CloudTrail Insights events	`Insight`	This event data store type can collect CloudTrail Insights events. To receive Insights events, you need a source event data store that logs CloudTrail management events and enables Insights. For information about creating the source and destination event data stores, see Create an event data store for CloudTrail Insights events.
Configuration items	`ConfigurationItem`	This event data store type can collect AWS Config configuration items. For more information, see Create an event data store for AWS Config configuration items.
Events from integration	`ActivityAuditLog`	This event data store type can collect non-AWS events from integrations. For more information, see Create an event data store for events outside of AWS.

You can also create an event data store for AWS Audit Manager evidence by using the Audit Manager console. For more information about aggregating evidence in CloudTrail Lake using Audit Manager, see Understanding how evidence finder works with CloudTrail Lake in the *AWS Audit Manager User Guide*.

CloudTrail Lake event data stores incur charges. When you create an event data store, you choose the pricing option you want to use for the event data store. The pricing option determines the cost for ingesting and storing events, and the default and maximum retention period for the event data store. For information about CloudTrail pricing and managing Lake costs, see AWS CloudTrail Pricing and Managing CloudTrail Lake costs.

The sections which follow describe how to create, update, and manage event data stores.

Topics

- Create, update, and manage event data stores with the console
- Create, update, and manage event data stores with the AWS CLI

- Manage event data store lifecycles
- Copy trail events to an event data store
- Federate an event data store
- Understanding organization event data stores

Create, update, and manage event data stores with the console

You can use the CloudTrail console to create, update, delete, and restore event data stores.

You can update the following settings using the CloudTrail console:

- You can change the pricing option from **Seven-year retention pricing** to **One-year extendable retention pricing**.
- You can update the retention period for the event data store. The retention period determines how long event data is kept in the event data store.
- You can convert a multi-Region event data store to a single-Region event data store, or convert a single-Region event data store to a multi-Region event data store.
- The management account for an AWS Organizations organization can convert an account-level event data store to an organization event data store, or can convert an organization event data store to an account-level event data store. This setting is not available on event data stores that collect events outside of AWS.
- You can enable or disable Lake query federation. Federating an event data store allows you to query your event data from Amazon Athena.
- You can stop event ingestion and restart event ingestion on event data stores that collect management events, data events, or AWS Config configuration items.
- You can enable or disable termination protection. Enabling termination protection protects an event data store from being accidentally deleted. Termination protection is enabled by default.
- You can restore an event data store that is pending deletion.
- You can add or remove tags. You can add up to 50 tag key pairs to help you identify, sort, and control access to your event data store.
- You can add a KMS key to encrypt your event data store. You can’t remove a KMS key from an event data store.

Using the CloudTrail console to create or update a event data stores provides the following advantages:

- If you're configuring an event data store to log data events, using the CloudTrail console allows you to view the available data event types. You can also choose the log selector template that you want to use. For each data event type, you can choose to log all data events, log all read events, log all write events, or create a custom log selector template to apply filters on the `eventnName`, `resources.ARN`, and `readOnly` fields. For more information, see Create an event data store for CloudTrail events with the console and Logging data events.
- If your event data store is already logging data events, using the console allows you to easily add logging for additional data event types.
- If you're configuring a event data store to log events outside of AWS, using the CloudTrail console lets you view information about available partners. For more information, see Create an event data store for events outside of AWS with the console.

Topics

Create an event data store for CloudTrail events with the console

Event data stores for CloudTrail events can log CloudTrail management and data events. You can keep the event data in an event data store for up to 3,653 days (about 10 years) if you choose the **One-year extendable retention pricing** option, or up to 2,557 days (about 7 years) if you choose the **Seven-year retention pricing** option..

CloudTrail Lake event data stores incur charges. When you create an event data store, you choose the pricing option you want to use for the event data store. The pricing option determines the cost for ingesting and storing events, and the default and maximum retention period for the event data store. For information about CloudTrail pricing and managing Lake costs, see AWS CloudTrail Pricing and Managing CloudTrail Lake costs.

To create an event data store for CloudTrail management or data events

Use this procedure to create an event data store that logs CloudTrail management events, data events, or both management and data events.

1. Sign in to the AWS Management Console and open the CloudTrail console at https://console.aws.amazon.com/cloudtrail/.
2. From the navigation pane, under **Lake**, choose **Event data stores**.
3. Choose **Create event data store**.
4. On the **Configure event data store** page, in **General details**, enter a name for the event data store. A name is required.
5. Choose the **Pricing option** that you want to use for your event data store. The pricing option determines the cost for ingesting and storing events, and the default and maximum retention periods for your event data store. For more information, see AWS CloudTrail Pricing and Managing CloudTrail Lake costs.

   The following are the available options:

   - **One-year extendable retention pricing** - Generally recommended if you expect to ingest less than 25 TB of event data per month and want a flexible retention period of up to 10 years. For the first 366 days (the default retention period), storage is included at no additional charge with ingestion pricing. After 366 days, extended retention is available at pay-as-you-go pricing. This is the default option.
     - **Default retention period:** 366 days
     - **Maximum retention period:** 3,653 days
   - **Seven-year retention pricing** - Recommended if you expect to ingest more than 25 TB of event data per month and need a retention period of up to 7 years. Retention is included with ingestion pricing at no additional charge.
     - **Default retention period:** 2,557 days
     - **Maximum retention period:** 2,557 days
6. Specify a retention period for the event data store. Retention periods can be between 7 days and 3,653 days (about 10 years) for the **One-year extendable retention pricing** option, or

between 7 days and 2,557 days (about seven years) for the **Seven-year retention pricing** option.

CloudTrail Lake determines whether to retain an event by checking if the `eventTime` of the event is within the specified retention period. For example, if you specify a retention period of 90 days, CloudTrail will remove events when their `eventTime` is older than 90 days.

> **Note**
>
> If you are copying trail events to this event data store, CloudTrail will not copy an event if its `eventTime` is older than the specified retention period. To determine the appropriate retention period, take the sum of the oldest event you want to copy in days and the number of days you want to retain the events in the event data store (**retention period** = *oldest-event-in-days* + *number-days-to-retain*). For example, if the oldest event you're copying is 45 days old and you want to keep the events in the event data store for a further 45 days, you would set the retention period to 90 days.

7. (Optional) To enable encryption using AWS Key Management Service, choose **Use my own AWS KMS key**. Choose **New** to have an AWS KMS key created for you, or choose **Existing** to use an existing KMS key. In **Enter KMS alias**, specify an alias, in the format `alias/`*MyAliasName*. Using your own KMS key requires that you edit your KMS key policy to allow CloudTrail logs to be encrypted and decrypted. For more information, see Configure AWS KMS key policies for CloudTrail. CloudTrail also supports AWS KMS multi-Region keys. For more information about multi-Region keys, see Using multi-Region keys in the *AWS Key Management Service Developer Guide*.

   Using your own KMS key incurs AWS KMS costs for encryption and decryption. After you associate an event data store with a KMS key, the KMS key cannot be removed or changed.

   > **Note**
   >
   > To enable AWS Key Management Service encryption for an organization event data store, you must use an existing KMS key for the management account.

8. (Optional) If you want to query against your event data using Amazon Athena, choose **Enable** in **Lake query federation**. Federation lets you view the metadata associated with the event data store in the AWS Glue Data Catalog and run SQL queries against the event data in Athena.

The table metadata stored in the AWS Glue Data Catalog lets the Athena query engine know how to find, read, and process the data that you want to query. For more information, see Federate an event data store.

To enable Lake query federation, choose **Enable** and then do the following:

a. Choose whether you want to create a new role or use an existing IAM role. AWS Lake Formation uses this role to manage permissions for the federated event data store. When you create a new role using the CloudTrail console, CloudTrail automatically creates a role with the required permissions. If you choose an existing role, be sure the policy for the role provides the required minimum permissions.

b. If you are creating a new role, enter a name to identify the role.

c. If you are using an existing role, choose the role you want to use. The role must exist in your account.

9. (Optional) In the **Tags** section, you can add up to 50 tag key pairs to help you identify, sort, and control access to your event data store. For more information about how to use IAM policies to authorize access to an event data store based on tags, see Examples: Denying access to create or delete event data stores based on tags. For more information about how you can use tags in AWS, see Tagging AWS resources in the *Tagging AWS Resources User Guide*.
10. Choose **Next** to configure the event data store.
11. On the **Choose events** page, choose **AWS events**, and then choose **CloudTrail events**.
12. For **CloudTrail events**, choose at least one event type. By default, **Management events** is selected. You can add both management and data events to your event data store. For more information about management events, see Logging management events. For more information about data events, see Logging data events.
13. (Optional) Choose **Copy trail events** if you want to copy events from an existing trail to run queries on past events. To copy trail events to an organization event data store, you must use the management account for the organization. The delegated administrator account cannot copy trail events to an organization event data store. For more information about considerations for copying trail events, see Considerations for copying trail events.
14. To have your event data store collect events from all accounts in an AWS Organizations organization, select **Enable for all accounts in my organization**. You must be signed in to the management account or delegated administrator account for the organization to create an event data store that collects events for an organization.

> **Note**
>
> To copy trail events or enable Insights events, you must be signed in to the management account for your organization.

15. Expand **Additional settings** to choose whether you want your event data store to collect events for all AWS Regions, or only the current AWS Region, and choose whether the event data store ingests events. By default, your event data store collects events from all Regions in your account and starts ingesting events when it's created.

    a. Select **Include only the current region in my event data store** to include only events that are logged in the current Region. If you do not choose this option, your event data store includes events from all Regions.

    b. Deselect **Ingest events** if you do not want the event data store to start ingesting events. For example, you may want to deselect **Ingest events**, if you are copying trail events and do not want the event data store to include any future events. By default, the event data store starts ingesting events when it's created.

16. If your event data store includes management events, you can choose from the following options. For more information about management events, see Logging management events.

    a. Choose whether you want to include **Read** events, **Write** events, or both. At least one is required.

    b. Choose whether to exclude AWS Key Management Service or Amazon RDS Data API events from your event data store.

    c. Choose whether to enable Insights. To enable Insights, you need to set up a destination event data store to collect Insights events based upon the management event activity in this event data store.

       If you choose to enable Insights, do the following.

       i. In **Enable Insights**, choose the destination event store that will log Insights events. The destination event data store will collect Insights events based upon the management event activity in this event data store. For information about how to create the destination event data store, see To create a destination event data store that logs Insights events.

    ii. Choose the Insights types. You can choose **API call rate**, **API error rate**, or both. You must be logging **Write** management events to log Insights events for **API call rate**. You must be logging **Read** or **Write** management events to log Insights events for **API error rate**.

17. To include data events in your event data store, do the following.

    a. Choose a data event type. This is the AWS service and resource on which data events are logged. To log data events for AWS Glue tables created by Lake Formation, choose **Lake Formation** for the data type.

    b. In **Log selector template**, choose a template. You can choose to log all data events, `readOnly` events, `writeOnly` events, or **Custom** to build a custom log selector.

    c. (Optional) In **Selector name**, enter a name to identify your selector. The selector name is a descriptive name for an advanced event selector, such as "Log data events for only two S3 buckets". The selector name is listed as `Name` in the advanced event selector and is viewable if you expand the **JSON view**.

    d. In **Advanced event selectors**, build expressions by choosing values for **Field**, **Operator**, and **Value**. Advanced event selectors for an event data store work the same as advanced event selectors that you apply to a trail. For more information about how to build advanced event selectors, see [Filtering data events using advanced event selectors](#).

    > ⚠ **Important**
    >
    > To exclude or include data events with advanced event selectors by using an S3 bucket ARN, always use the **Starts with** operator.

    e. Optionally, expand **JSON view** to see your advanced event selectors as a JSON block.

    f. To add another data type on which to log data events, choose **Add data event type**. Repeat steps a through this step to configure advanced event selectors for the data event type.

18. To copy existing trail events to your event data store, do the following.

    a. Choose the trail that you want to copy. By default, CloudTrail only copies CloudTrail events contained in the S3 bucket's `CloudTrail` prefix and the prefixes inside the `CloudTrail` prefix, and does not check prefixes for other AWS services. If you want to copy CloudTrail events contained in another prefix, choose **Enter S3 URI**, and then choose **Browse S3** to browse to the prefix. If the source S3 bucket for the trail uses a KMS key for data encryption, ensure that the KMS key policy allows CloudTrail to decrypt the data. If your source S3 bucket uses multiple KMS keys, you must update each key's policy to allow CloudTrail to decrypt the data in the bucket. For more information about updating the KMS key policy, see KMS key policy for decrypting data in the source S3 bucket.

b. Choose the time range for copying the events. CloudTrail checks the prefix and log file name to verify the name contains a date between the chosen start and end date before attempting to copy trail events. You can choose a **Relative range** or an **Absolute range**. To avoid duplicating events between the source trail and destination event data store, choose a time range that is earlier than the creation of the event data store.

> **Note**
>
> CloudTrail only copies trail events that have an `eventTime` within the event data store’s retention period. For example, if an event data store’s retention period is 90 days, then CloudTrail will not copy any trail events with an `eventTime` older than 90 days.

- If you choose **Relative range**, you can choose to copy events logged in the last 6 months, 1 year, 2 years, 7 years, or a custom range. CloudTrail copies the events logged within the chosen time period.
- If you choose **Absolute range**, you can choose a specific start and end date. CloudTrail copies the events that occurred between the chosen start and end dates.

c. For **Permissions**, choose from the following IAM role options. If you choose an existing IAM role, verify that the IAM role policy provides the necessary permissions. For more information about updating the IAM role permissions, see IAM permissions for copying trail events.

- Choose **Create a new role (recommended)** to create a new IAM role. For **Enter IAM role name**, enter a name for the role. CloudTrail automatically creates the necessary permissions for this new role.
- Choose **Use a custom IAM role ARN** to use a custom IAM role that is not listed. For **Enter IAM role ARN**, enter the IAM ARN.
- Choose an existing IAM role from the drop-down list.

19. Choose **Next** to review your choices.

20. On the **Review and create** page, review your choices. Choose **Edit** to make changes to a section. When you're ready to create the event data store, choose **Create event data store**.
21. The new event data store is visible in the **Event data stores** table on the **Event data stores** page.

    From this point forward, the event data store captures events that match its advanced event selectors (if you kept the **Ingest events** option selected). Events that occurred before you created the event data store are not in the event data store, unless you opted to copy existing trail events.

You can now run queries on your new event data store. The **Sample queries** tab provides example queries to get you started. For more information about creating and editing queries, see Create or edit a query with the CloudTrail console.

You can also view the CloudTrail Lake dashboard to visualize the events in your event data store. For more information about Lake dashboards, see View CloudTrail Lake dashboards with the CloudTrail console.

Create an event data store for Insights events with the console

AWS CloudTrail Insights help AWS users identify and respond to unusual activity associated with API calls and API error rates by continuously analyzing CloudTrail management events. CloudTrail Insights analyze your normal patterns of API call volume and API error rates, also called the *baseline*, and generate Insights events when the call volume or error rates are outside normal patterns. Insights events on API call volume are generated for `write` management APIs, and Insights events on API error rate are generated for both `read` and `write` management APIs.

To log Insights events in CloudTrail Lake, you need a destination event data store that logs Insights events and a source event data store that enables Insights and logs management events.

> **Note**
>
> To log Insights events on API call volume, the source event data store must log `write` management events. To log Insights events on API error rate, the source event data store must log `read` or `write` management events.

If you have CloudTrail Insights enabled on a source event data store and CloudTrail detects unusual activity, CloudTrail delivers Insights events to your destination event data store. Unlike other types of events captured in a CloudTrail event data store, Insights events are logged only when CloudTrail detects changes in your account's API usage that differ significantly from the account's typical usage patterns.

After you enable CloudTrail Insights for the first time on an event data store, it can take up to 7 days for CloudTrail to deliver the first Insights event, if unusual activity is detected.

CloudTrail Insights analyzes management events that occur in a single Region, not globally. A CloudTrail Insights event is generated in the same Region as its supporting management events are generated.

For an organization event data store, CloudTrail analyzes management events from each member's account instead of analyzing the aggregation of all management events for the organization.

Additional charges apply for ingesting Insights events in CloudTrail Lake. You will be charged separately if you enable Insights for both trails and CloudTrail Lake event data stores. For information about CloudTrail pricing, see AWS CloudTrail Pricing.

Topics

To create a destination event data store that logs Insights events

When you create an Insights event data store, you have the option to choose an existing source event data store that logs management events and then specify the Insights types you want to receive. Or, you can alternatively enable Insights on a new or existing event data store after you create your Insights event data store and then choose this event data store as the destination event data store.

This procedure shows you how to create a destination event data store that logs Insights events.

1. Sign in to the AWS Management Console and open the CloudTrail console at https://console.aws.amazon.com/cloudtrail/.
2. From the navigation pane, open the **Lake** submenu, then choose **Event data stores**.
3. Choose **Create event data store**.
4. On the **Configure event data store** page, in **General details**, enter a name for the event data store. A name is required.

5. Choose the **Pricing option** that you want to use for your event data store. The pricing option determines the cost for ingesting and storing events, and the default and maximum retention periods for your event data store. For more information, see AWS CloudTrail Pricing and Managing CloudTrail Lake costs.

   The following are the available options:

   - **One-year extendable retention pricing** - Generally recommended if you expect to ingest less than 25 TB of event data per month and want a flexible retention period of up to 10 years. For the first 366 days (the default retention period), storage is included at no additional charge with ingestion pricing. After 366 days, extended retention is available at pay-as-you-go pricing. This is the default option.
     - **Default retention period:** 366 days
     - **Maximum retention period:** 3,653 days
   - **Seven-year retention pricing** - Recommended if you expect to ingest more than 25 TB of event data per month and need a retention period of up to 7 years. Retention is included with ingestion pricing at no additional charge.
     - **Default retention period:** 2,557 days
     - **Maximum retention period:** 2,557 days
6. Specify a retention period for the event data store in days. Retention periods can be between 7 days and 3,653 days (about 10 years) for the **One-year extendable retention pricing** option, or between 7 days and 2,557 days (about seven years) for the **Seven-year retention pricing** option. The event data store retains event data for the specified number of days.
7. (Optional) To enable encryption using AWS Key Management Service, choose **Use my own AWS KMS key**. Choose **New** to have an AWS KMS key created for you, or choose **Existing** to use an existing KMS key. In **Enter KMS alias**, specify an alias, in the format `alias/MyAliasName`. Using your own KMS key requires that you edit your KMS key policy to allow CloudTrail logs to be encrypted and decrypted. For more information, see Configure AWS KMS key policies for CloudTrail. CloudTrail also supports AWS KMS multi-Region keys. For more information about multi-Region keys, see Using multi-Region keys in the *AWS Key Management Service Developer Guide*.

   Using your own KMS key incurs AWS KMS costs for encryption and decryption. After you associate an event data store with a KMS key, the KMS key cannot be removed or changed.

> **Note**
>
> To enable AWS Key Management Service encryption for an organization event data store, you must use an existing KMS key for the management account.

8. (Optional) If you want to query against your event data using Amazon Athena, choose **Enable** in **Lake query federation**. Federation lets you view the metadata associated with the event data store in the AWS Glue Data Catalog and run SQL queries against the event data in Athena. The table metadata stored in the AWS Glue Data Catalog lets the Athena query engine know how to find, read, and process the data that you want to query. For more information, see Federate an event data store.

   To enable Lake query federation, choose **Enable** and then do the following:

   a. Choose whether you want to create a new role or use an existing IAM role. AWS Lake Formation uses this role to manage permissions for the federated event data store. When you create a new role using the CloudTrail console, CloudTrail automatically creates a role with the required permissions. If you choose an existing role, be sure the policy for the role provides the required minimum permissions.
   b. If you are creating a new role, enter a name to identify the role.
   c. If you are using an existing role, choose the role you want to use. The role must exist in your account.
9. (Optional) In the **Tags** section, you can add up to 50 tag key pairs to help you identify, sort, and control access to your event data store. For more information about how to use IAM policies to authorize access to an event data store based on tags, see Examples: Denying access to create or delete event data stores based on tags. For more information about how you can use tags in AWS, see Tagging your AWS resources in the *Tagging AWS Resources User Guide*.
10. Choose **Next** to configure the event data store.
11. On the **Choose events** page, choose **AWS events**, and then choose **CloudTrail Insights events**.
12. In **CloudTrail Insights events**, do the following.

    a. Choose **Allow delegated administrator access** if you want to give your organization's delegated administrator access to this event data store. This option is only available if you are signed in with the management account for an AWS Organizations organization.
    b. (Optional) Choose an existing source event data store that logs management events and specify the Insights types you want to receive.

To add a source event data store, do the following.

i. Choose **Add source event data store**.

ii. Choose the source event data store.

iii. Choose the **Insights type** that you want to receive.

- `ApiCallRateInsight` – The `ApiCallRateInsight` Insights type analyzes write-only management API calls that are aggregated per minute against a baseline API call volume. To receives Insights on `ApiCallRateInsight`, the source event data store must log **Write** management events.
- `ApiErrorRateInsight` – The `ApiErrorRateInsight` Insights type analyzes management API calls that result in error codes. The error is shown if the API call is unsuccessful. To receive Insights on `ApiErrorRateInsight`, the source event data store must log **Write** or **Read** management events.

iv. Repeat the previous two steps (ii and iii) to add any additional Insights types you want to receive.

13. Choose **Next** to review your choices.
14. On the **Review and create** page, review your choices. Choose **Edit** to make changes to a section. When you're ready to create the event data store, choose **Create event data store**.
15. The new event data store is visible in the **Event data stores** table on the **Event data stores** page.
16. If you did not choose a source event data store in step 10, follow the steps in To create a source event data store that enables Insights events to create a source event data store.

To create a source event data store that enables Insights events

This procedure shows you how to create a source event data store that enables Insights events and logs management events.

1. Sign in to the AWS Management Console and open the CloudTrail console at https://console.aws.amazon.com/cloudtrail/.
2. From the navigation pane, open the **Lake** submenu, then choose **Event data stores**.
3. Choose **Create event data store**.
4. On the **Configure event data store** page, in **General details**, enter a name for the event data store. A name is required.

5. Choose the **Pricing option** that you want to use for your event data store. The pricing option determines the cost for ingesting and storing events, and the default and maximum retention periods for your event data store. For more information, see AWS CloudTrail Pricing and Managing CloudTrail Lake costs.

   The following are the available options:

   - **One-year extendable retention pricing** - Generally recommended if you expect to ingest less than 25 TB of event data per month and want a flexible retention period of up to 10 years. For the first 366 days (the default retention period), storage is included at no additional charge with ingestion pricing. After 366 days, extended retention is available at pay-as-you-go pricing. This is the default option.
     - **Default retention period:** 366 days
     - **Maximum retention period:** 3,653 days
   - **Seven-year retention pricing** - Recommended if you expect to ingest more than 25 TB of event data per month and need a retention period of up to 7 years. Retention is included with ingestion pricing at no additional charge.
     - **Default retention period:** 2,557 days
     - **Maximum retention period:** 2,557 days

6. Specify a retention period for the event data store. Retention periods can be between 7 days and 3,653 days (about 10 years) for the **One-year extendable retention pricing** option, or between 7 days and 2,557 days (about seven years) for the **Seven-year retention pricing** option.

   CloudTrail Lake determines whether to retain an event by checking if the `eventTime` of the event is within the specified retention period. For example, if you specify a retention period of 90 days, CloudTrail will remove events when their `eventTime` is older than 90 days.

7. (Optional) To enable encryption using AWS Key Management Service, choose **Use my own AWS KMS key**. Choose **New** to have an AWS KMS key created for you, or choose **Existing** to use an existing KMS key. In **Enter KMS alias**, specify an alias, in the format `alias/`*`MyAliasName`*. Using your own KMS key requires that you edit your KMS key policy to allow CloudTrail logs to be encrypted and decrypted. For more information, see Configure AWS KMS key policies for CloudTrail. CloudTrail also supports AWS KMS multi-Region keys. For more information about multi-Region keys, see Using multi-Region keys in the *AWS Key Management Service Developer Guide*.

Using your own KMS key incurs AWS KMS costs for encryption and decryption. After you associate an event data store with a KMS key, the KMS key cannot be removed or changed.

> **Note**
>
> To enable AWS Key Management Service encryption for an organization event data store, you must use an existing KMS key for the management account.

8. (Optional) If you want to query against your event data using Amazon Athena, choose **Enable** in **Lake query federation**. Federation lets you view the metadata associated with the event data store in the AWS Glue Data Catalog and run SQL queries against the event data in Athena. The table metadata stored in the AWS Glue Data Catalog lets the Athena query engine know how to find, read, and process the data that you want to query. For more information, see Federate an event data store.

   To enable Lake query federation, choose **Enable** and then do the following:

   a. Choose whether you want to create a new role or use an existing IAM role. AWS Lake Formation uses this role to manage permissions for the federated event data store. When you create a new role using the CloudTrail console, CloudTrail automatically creates a role with the required permissions. If you choose an existing role, be sure the policy for the role provides the required minimum permissions.
   b. If you are creating a new role, enter a name to identify the role.
   c. If you are using an existing role, choose the role you want to use. The role must exist in your account.
9. (Optional) In the **Tags** section, you can add up to 50 tag key pairs to help you identify, sort, and control access to your event data store. For more information about how to use IAM policies to authorize access to an event data store based on tags, see Examples: Denying access to create or delete event data stores based on tags. For more information about how you can use tags in AWS, see Tagging your AWS resources in the *Tagging AWS Resources User Guide*.
10. Choose **Next** to configure the event data store.
11. On the **Choose events** page, choose **AWS events**, and then choose **CloudTrail events**.
12. In **CloudTrail events**, leave **Management events** selected.
13. To have your event data store collect events from all accounts in an AWS Organizations organization, select **Enable for all accounts in my organization**. You must be signed in to the management account for the organization to create an event data store that enables Insights.

14. Expand **Additional settings** to choose whether you want your event data store to collect events for all AWS Regions, or only the current AWS Region, and choose whether the event data store ingests events. By default, your event data store collects events from all Regions in your account and starts ingesting events when it's created.

    a. Choose **Include only the current region in my event data store** if you want to include only events that are logged in the current Region. If you do not choose this option, your event data store includes events from all Regions.

    b. Leave **Ingest events** selected.

15. Choose the type of management events you want to include in your event data store. You can choose **Read**, **Write**, or both. At least one is required.

    > **Note**
    >
    > To log Insights events on API call volume, the event data store must log `write` management events. To log Insights events on API error rate, the event data store must log `read` or `write` management events.

16. You can choose to exclude AWS Key Management Service or Amazon RDS Data API events from your event data store. For more information about these options, see Logging management events.

17. Choose **Enable Insights**.

18. In **Enable Insights**, choose the destination event store that will log Insights events. The destination event data store will collect Insights events based upon the management event activity in this event data store. For information about how to create the destination event data store, see To create a destination event data store that logs Insights events.

19. Choose the Insights types. You can choose **API call rate**, **API error rate**, or both. You must be logging **Write** management events to log Insights events for **API call rate**. You must be logging **Read** or **Write** management events to log Insights events for **API error rate**.

20. Choose **Next** to review your choices.

21. On the **Review and create** page, review your choices. Choose **Edit** to make changes to a section. When you're ready to create the event data store, choose **Create event data store**.

22. The new event data store is visible in the **Event data stores** table on the **Event data stores** page.

From this point forward, the event data store captures events that match its advanced event selectors. After you enable CloudTrail Insights for the first time on your source event data store, it can take up to 7 days for CloudTrail to deliver the first Insights event to your destination event data store, if unusual activity is detected.

You can view the CloudTrail Lake dashboard to visualize the Insights events in your destination event data store. For more information about Lake dashboards, see View CloudTrail Lake dashboards with the CloudTrail console.

Additional charges apply for ingesting Insights events in CloudTrail Lake. You will be charged separately if you enable Insights for both trails and event data stores. For information about CloudTrail pricing, see AWS CloudTrail Pricing.

Create an event data store for configuration items with the console

You can create an event data store to include AWS Config configuration items, and use the event data store to investigate non-compliant changes to your production environments. With an event data store, you can relate non-compliant rules to the users and resources associated with the changes. A configuration item represents a point-in-time view of the attributes of a supported AWS resource that exists in your account. AWS Config creates a configuration item whenever it detects a change to a resource type that it is recording. AWS Config also creates configuration items when a configuration snapshot is captured.

You can use both AWS Config and CloudTrail Lake to run queries against your configuration items. You can use AWS Config to query the current configuration state of AWS resources based on configuration properties for a single AWS account and AWS Region, or across multiple accounts and Regions. In contrast, you can use CloudTrail Lake to query across diverse data sources such as CloudTrail events, configuration items, and rule evaluations. CloudTrail Lake queries cover all AWS Config configuration items including resource configuration and compliance history.

Creating an event data store for configuration items doesn't impact existing AWS Config advanced queries, or any configured AWS Config aggregators. You can continue to run advanced queries using AWS Config, and AWS Config continues to deliver history files to your S3 buckets.

CloudTrail Lake event data stores incur charges. When you create an event data store, you choose the pricing option you want to use for the event data store. The pricing option determines the cost for ingesting and storing events, and the default and maximum retention period for the event data store. For information about CloudTrail pricing and managing Lake costs, see AWS CloudTrail Pricing and Managing CloudTrail Lake costs.

Limitations

The following limitations apply to event data stores for configuration items.

- No support for custom configuration items
- No support for event filtering using advanced event selectors

Prerequisites

Before you create your event data store, set up AWS Config recording for all your accounts and Regions. You can use Quick Setup, a capability of AWS Systems Manager, to quickly create a configuration recorder powered by AWS Config.

> **Note**
>
> You are charged service usage fees when AWS Config starts recording configurations. For more information about pricing, see AWS Config Pricing. For information about managing the configuration recorder, see Managing the Configuration Recorder in the *AWS Config Developer Guide*.

Additionally, the following actions are recommended, but are not required to create an event data store.

- Set up an Amazon S3 bucket to receive a configuration snapshot on request and configuration history. For more information about snapshots, see Managing the Delivery Channel and Delivering Configuration Snapshot to an Amazon S3 Bucket in the *AWS Config Developer Guide*.
- Specify the rules that you want AWS Config to use to evaluate compliance information for the recorded resource types. Several of the CloudTrail Lake sample queries for AWS Config require AWS Config Rules to evaluate the compliance state of your AWS resources. For more information about AWS Config Rules, see Evaluating Resources with AWS Config Rules in the *AWS Config Developer Guide*.

To create an event data store for configuration items

1. Sign in to the AWS Management Console and open the CloudTrail console at https://console.aws.amazon.com/cloudtrail/.
2. From the navigation pane, under **Lake**, choose **Event data stores**.
3. Choose **Create event data store**.
4. On the **Configure event data store** page, in **General details**, enter a name for the event data store. A name is required.
5. Choose the **Pricing option** that you want to use for your event data store. The pricing option determines the cost for ingesting and storing events, and the default and maximum retention periods for your event data store. For more information, see AWS CloudTrail Pricing and Managing CloudTrail Lake costs.

   The following are the available options:

   - **One-year extendable retention pricing** - Generally recommended if you expect to ingest less than 25 TB of event data per month and want a flexible retention period of up to 10 years. For the first 366 days (the default retention period), storage is included at no additional charge with ingestion pricing. After 366 days, extended retention is available at pay-as-you-go pricing. This is the default option.
     - **Default retention period:** 366 days
     - **Maximum retention period:** 3,653 days
   - **Seven-year retention pricing** - Recommended if you expect to ingest more than 25 TB of event data per month and need a retention period of up to 7 years. Retention is included with ingestion pricing at no additional charge.
     - **Default retention period:** 2,557 days
     - **Maximum retention period:** 2,557 days
6. Specify a retention period for the event data store. Retention periods can be between 7 days and 3,653 days (about 10 years) for the **One-year extendable retention pricing** option, or between 7 days and 2,557 days (about seven years) for the **Seven-year retention pricing** option.

   CloudTrail Lake determines whether to retain an event by checking if the `eventTime` of the event is within the specified retention period. For example, if you specify a retention period of 90 days, CloudTrail will remove events when their `eventTime` is older than 90 days.

7. (Optional) To enable encryption using AWS Key Management Service, choose **Use my own AWS KMS key**. Choose **New** to have an AWS KMS key created for you, or choose **Existing** to use an existing KMS key. In **Enter KMS alias**, specify an alias, in the format `alias/MyAliasName`. Using your own KMS key requires that you edit your KMS key policy to allow CloudTrail logs to be encrypted and decrypted. For more information, see Configure AWS KMS key policies for CloudTrail. CloudTrail also supports AWS KMS multi-Region keys. For more information about multi-Region keys, see Using multi-Region keys in the *AWS Key Management Service Developer Guide*.

   Using your own KMS key incurs AWS KMS costs for encryption and decryption. After you associate an event data store with a KMS key, the KMS key cannot be removed or changed.

   > **Note**
   >
   > To enable AWS Key Management Service encryption for an organization event data store, you must use an existing KMS key for the management account.

8. (Optional) If you want to query against your event data using Amazon Athena, choose **Enable** in **Lake query federation**. Federation lets you view the metadata associated with the event data store in the AWS Glue Data Catalog and run SQL queries against the event data in Athena. The table metadata stored in the AWS Glue Data Catalog lets the Athena query engine know how to find, read, and process the data that you want to query. For more information, see Federate an event data store.

   To enable Lake query federation, choose **Enable** and then do the following:

   a. Choose whether you want to create a new role or use an existing IAM role. AWS Lake Formation uses this role to manage permissions for the federated event data store. When you create a new role using the CloudTrail console, CloudTrail automatically creates a role with the required permissions. If you choose an existing role, be sure the policy for the role provides the required minimum permissions.

   b. If you are creating a new role, enter a name to identify the role.

   c. If you are using an existing role, choose the role you want to use. The role must exist in your account.

9. (Optional) In the **Tags** section, you can add up to 50 tag key pairs to help you identify, sort, and control access to your event data store. For more information about how to use IAM policies to authorize access to an event data store based on tags, see Examples: Denying access

to create or delete event data stores based on tags. For more information about how you can use tags in AWS, see Tagging your AWS resources in the *Tagging AWS Resources User Guide*.

10. Choose **Next**.
11. On the **Choose events** page, choose **AWS events**, and then choose **Configuration items**.
12. CloudTrail stores the event data store resource in the Region in which you create it, but by default, the configuration items collected in the data store are from all Regions in your account that have recording enabled. Optionally, you can select **Include only the current region in my event data store** to include only configuration items that are captured in the current Region. If you do not choose this option, your event data store includes configuration items from all Regions that have recording enabled.
13. To have your event data store collect configuration items from all accounts in an AWS Organizations organization, select **Enable for all accounts in my organization**. You must be signed in to the management account or delegated administrator account for the organization to create an event data store that collects configuration items for an organization.
14. Choose **Next** to review your choices.
15. On the **Review and create** page, review your choices. Choose **Edit** to make changes to a section. When you're ready to create the event data store, choose **Create event data store**.
16. The new event data store is visible in the **Event data stores** table on the **Event data stores** page.

    From this point forward, the event data store captures configuration items. Configuration items that occurred before you created the event data store are not in the event data store.

Sample queries

You can now run queries on your new event data store. The **Sample queries** tab on the CloudTrail console provides example queries to get you started. The following are a few of the sample queries that you can run against your configuration item event data store.

Description	Query

Find which user performed an action that resulted in a non-compliant status by joining a configuration item event data store with a CloudTrail event data store.	`SELECT`
	`    element_at(config1.eventData.configuration, 'targetResourceId') as targetResourceId,`

Description	Query
	element_at(config1.eventDat a.configuration, 'complianceType') as complianceType, config2.eventData.resourceType, cloudtrail.userIdentity FROM *config_event_data_store_ID* as config1 JOIN *config_event_data_store_ID* as config2 on element_at(config1 .eventData.configuration, 'targetRe sourceId') = config2.eventData. resourceId JOIN *cloudtrail_event_data_store_ID* as cloudtrail on config2.eventData. arn = element_at(cloudtrail.resou rces, 1).arn WHERE element_at(config1.eventDat a.configuration, 'configRuleList') is not null AND element_at(config1.eventDat a.configuration, 'complianceType') = 'NON_COMPLIANT' AND cloudtrail.eventTime > '2022-11- 14 00:00:00' AND config2.eventData.resourceType = 'AWS::DynamoDB::Table'

Description	Query

Find all AWS Config rules and return the compliance state from configuration items generated within the past day.	`SELECT`
	`    eventData.configuration, eventData.accountId, eventData.awsRegion,`
	`    eventData.resourceName, eventData.resourceCreationTime,`
	`    element_at(eventData.configuration,'complianceType') AS complianceType,`
	`    element_at(eventData.configuration, 'configRuleList') AS configRuleList,`
	`    element_at(eventData.configuration, 'resourceId') AS resourceId,`
	`    element_at(eventData.configuration, 'resourceType') AS resourceType`
	`FROM`
	`    config_event_data_store_ID`
	`WHERE`
	`    eventData.resourceType = 'AWS::Config::ResourceCompliance'`
	`AND`
	`    eventTime > '2022-11-22 00:00:00'`
	`ORDER BY`
	`    eventData.resourceCreationTime`
	`DESC`
	`    limit 10`

Description	Query

Find the total count of AWS Config resources grouped by resource type, account ID, and Region.	SELECT
	    eventData.resourceType, eventData
	.awsRegion, eventData.accountId,
	COUNT (*) AS resourceCount
	FROM
	    config_event_data_store_ID
	WHERE
	    eventTime > '2022-11-22 00:00:00'
	GROUP BY
	    eventData.resourceType, eventData
	.awsRegion, eventData.accountId

Find the resource creation time for all AWS Config configuration items generated on a specific date.	SELECT
	    eventData.configuration,
	 eventData.accountId,
	    eventData.awsRegion, eventData
	.resourceId,
	    eventData.resourceName, eventData
	.resourceType,
	    eventData.availabilityZone,
	 eventData.resourceCreationTime
	FROM
	    config_event_data_store_ID
	WHERE
	    eventTime > '2022-11-16 00:00:00'
	AND
	    eventTime < '2022-11-17 00:00:00'
	
	ORDER BY
	    eventData.resourceCreationTime
	DESC
	    limit 10;

For more information about creating and editing queries, see Create or edit a query with the CloudTrail console.

Configuration item schema

The following table describes the required and optional schema elements that match those in configuration item records. The contents of `eventData` are provided by your configuration items; other fields are provided by CloudTrail after ingestion.

CloudTrail event record contents are described in more detail in CloudTrail record contents.

- Fields that are provided by CloudTrail after ingestion
- Fields that are provided by your events

Fields that are provided by CloudTrail after ingestion

Field name	Input type	Requirement	Description
eventVersion	string	Required	The version of the AWS event format.
eventCategory	string	Required	The event category. For configuration items, the valid value is `ConfigurationItem` .
eventType	string	Required	The event type. For configuration items, the valid value is `AwsConfigurationItem` .
eventID	string	Required	A unique ID for an event.
eventTime	string	Required	The event timestamp, in `yyyy-MM-DDTHH:mm:ss` format, in Universal

Field name	Input type	Requirement	Description
			Coordinated Time (UTC).
awsRegion	string	Required	The AWS Region to which to assign an event.
recipientAccountId	string	Required	Represents the AWS account ID that received this event.
addendum	addendum	Optional	Shows information about why an event was delayed. If information was missing from an existing event, the addendum block includes the missing information and a reason for why it was missing.

Fields in `eventData` are provided by your configuration items

Field name	Input type	Requirement	Description
eventData	-	Required	Fields in eventData are provided by your configuration items.
• configurationItemVersion	string	Optional	The version of the configuration item from its source.

Field name	Input type	Requirement	Description
- configurationItemCaptureTime	string	Optional	The time when the configuration recording was initiated.
- configurationItemStatus	string	Optional	The configuration item status. Valid values are `OK`, `ResourceDiscovered` , `ResourceNotRecorded` , `ResourceDeleted` , and `ResourceDeletedNotRecorded` .
- accountId	string	Optional	The 12-digit AWS account ID associated with the resource.
- resourceType	string	Optional	The type of AWS resource. For more information about valid resource types, see ConfigurationItem in the *AWS Config API Reference*.
- resourceId	string	Optional	The ID of the resource (for example., sg-*xxxxxx*).

Field name	Input type	Requirement	Description
• resourceName	string	Optional	The custom name of the resource, if available.
• arn	string	Optional	Amazon Resource Name (ARN) associated with the resource.
• awsRegion	string	Optional	The AWS Region where the resource resides.
• availabilityZone	string	Optional	The Availability Zone associated with the resource.
• resourceCreationTime	string	Optional	The time stamp when the resource was created.
• configuration	JSON	Optional	The description of the resource configuration.
• supplementaryConfiguration	JSON	Optional	Configuration attributes that AWS Config returns for certain resource types to supplement the information returned for the configuration parameter.
• relatedEvents	string	Optional	A list of CloudTrail event IDs.

Field name	Input type	Requirement	Description
• relationships	-	Optional	A list of related AWS resources.
• • name	string	Optional	The type of relationship with the related resource.
• • resourceType	string	Optional	The resource type of the related resource.
• • resourceId	string	Optional	The ID of the related resource (for example, sg-*xxxxxx*).
• • resourceName	string	Optional	The custom name of the related resource, if available.
• tags	JSON	Optional	A mapping of key value tags associated with the resource.

The following example shows the hierarchy of schema elements that match those in configuration item records.

```
{
  "eventVersion": String,
  "eventCategory: String,
  "eventType": String,
  "eventID": String,
  "eventTime": String,
  "awsRegion": String,
  "recipientAccountId": String,
  "addendum": Addendum,
  "eventData": {
      "configurationItemVersion": String,
      "configurationItemCaptureTime": String,
      "configurationItemStatus": String,
      "configurationStateId": String,
      "accountId": String,
      "resourceType": String,
      "resourceId": String,
      "resourceName": String,
      "arn": String,
      "awsRegion": String,
      "availabilityZone": String,
      "resourceCreationTime": String,
      "configuration": {
        JSON,
      },
      "supplementaryConfiguration": {
        JSON,
      },
      "relatedEvents": [
        String
      ],
      "relationships": [
        struct{
          "name" : String,
          "resourceType": String,
          "resourceId": String,
          "resourceName": String
        }
      ],
     "tags": {
       JSON
     }
    }
  }
}
```

Create an event data store for events outside of AWS with the console

You can create an event data store to include events outside of AWS, and then use CloudTrail Lake to search, query, and analyze the data that is logged from your applications.

You can use CloudTrail Lake *integrations* to log and store user activity data from outside of AWS; from any source in your hybrid environments, such as in-house or SaaS applications hosted on-premises or in the cloud, virtual machines, or containers.

When you create an event data store for an integration, you also create a channel, and attach a resource policy to the channel.

CloudTrail Lake event data stores incur charges. When you create an event data store, you choose the pricing option you want to use for the event data store. The pricing option determines the cost for ingesting and storing events, and the default and maximum retention period for the event data store. For information about CloudTrail pricing and managing Lake costs, see AWS CloudTrail Pricing and Managing CloudTrail Lake costs.

To create an event data store for events outside of AWS

1. Sign in to the AWS Management Console and open the CloudTrail console at https://console.aws.amazon.com/cloudtrail/.
2. From the navigation pane, under **Lake**, choose **Event data stores**.
3. Choose **Create event data store**.
4. On the **Configure event data store** page, in **General details**, enter a name for the event data store. A name is required.
5. Choose the **Pricing option** that you want to use for your event data store. The pricing option determines the cost for ingesting and storing events, and the default and maximum retention periods for your event data store. For more information, see AWS CloudTrail Pricing and Managing CloudTrail Lake costs.

   The following are the available options:

   - **One-year extendable retention pricing** - Generally recommended if you expect to ingest less than 25 TB of event data per month and want a flexible retention period of up to 10 years. For the first 366 days (the default retention period), storage is included at no additional charge with ingestion pricing. After 366 days, extended retention is available at pay-as-you-go pricing. This is the default option.
     - **Default retention period:** 366 days
     - **Maximum retention period:** 3,653 days
   - **Seven-year retention pricing** - Recommended if you expect to ingest more than 25 TB of event data per month and need a retention period of up to 7 years. Retention is included with ingestion pricing at no additional charge.
     - **Default retention period:** 2,557 days
     - **Maximum retention period:** 2,557 days

6. Specify a retention period for the event data store. Retention periods can be between 7 days and 3,653 days (about 10 years) for the **One-year extendable retention pricing** option, or between 7 days and 2,557 days (about seven years) for the **Seven-year retention pricing** option.

   CloudTrail Lake determines whether to retain an event by checking if the `eventTime` of the event is within the specified retention period. For example, if you specify a retention period of 90 days, CloudTrail will remove events when their `eventTime` is older than 90 days.

7. (Optional) To enable encryption using AWS Key Management Service, choose **Use my own AWS KMS key**. Choose **New** to have an AWS KMS key created for you, or choose **Existing** to use an existing KMS key. In **Enter KMS alias**, specify an alias, in the format `alias/`*`MyAliasName`*. Using your own KMS key requires that you edit your KMS key policy to allow CloudTrail logs to be encrypted and decrypted. For more information, see Configure AWS KMS key policies for CloudTrail. CloudTrail also supports AWS KMS multi-Region keys. For more information about multi-Region keys, see Using multi-Region keys in the *AWS Key Management Service Developer Guide*.

   Using your own KMS key incurs AWS KMS costs for encryption and decryption. After you associate an event data store with a KMS key, the KMS key cannot be removed or changed.

   > **Note**
   >
   > To enable AWS Key Management Service encryption for an organization event data store, you must use an existing KMS key for the management account.

8. (Optional) If you want to query against your event data using Amazon Athena, choose **Enable** in **Lake query federation**. Federation lets you view the metadata associated with the event data store in the AWS Glue Data Catalog and run SQL queries against the event data in Athena. The table metadata stored in the AWS Glue Data Catalog lets the Athena query engine know how to find, read, and process the data that you want to query. For more information, see Federate an event data store.

   To enable Lake query federation, choose **Enable** and then do the following:

   a. Choose whether you want to create a new role or use an existing IAM role. AWS Lake Formation uses this role to manage permissions for the federated event data store. When you create a new role using the CloudTrail console, CloudTrail automatically creates a role with the required permissions. If you choose an existing role, be sure the policy for the role provides the required minimum permissions.

   b. If you are creating a new role, enter a name to identify the role.

   c. If you are using an existing role, choose the role you want to use. The role must exist in your account.

9. (Optional) In the **Tags** section, you can add up to 50 tag key pairs to help you identify, sort, and control access to your event data store. For more information about how to use IAM policies to authorize access to an event data store based on tags, see Examples: Denying access to create or delete event data stores based on tags. For more information about how you can use tags in AWS, see Tagging your AWS resources in the *Tagging AWS Resources User Guide*.

10. Choose **Next** to configure the event data store.

11. On the **Choose events** page, choose **Events from integrations**.

12. From **Events from integration**, choose the source to deliver events to the event data store.

13. Provide a name to identify the integration's channel. The name can be 3-128 characters. Only letters, numbers, periods, underscores, and dashes are allowed.

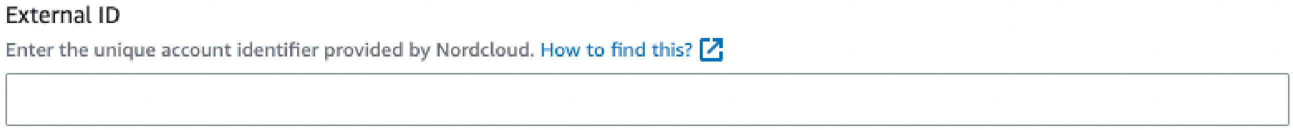
14. In **Resource policy**, configure the resource policy for the integration's channel. Resource policies are JSON policy documents that specify what actions a specified principal can perform on the resource and under what conditions. The accounts defined as principals in the resource policy can call the `PutAuditEvents` API to deliver events to your channel. The resource owner has implicit access to the resource if their IAM policy allows the `cloudtrail-data:PutAuditEvents` action.

    The information required for the policy is determined by the integration type. For a direction integration, CloudTrail automatically adds the partner's AWS account IDs, and requires you to enter the unique external ID provided by the partner. For a solution integration, you must specify at least one AWS account ID as principal, and can optionally enter an external ID to prevent against confused deputy.

    > **Note**
    >
    > If you do not create a resource policy for the channel, only the channel owner can call the `PutAuditEvents` API on the channel.

a. For a direct integration, enter the external ID provided by your partner. The integration partner provides a unique external ID, such as an account ID or a randomly generated string, to use for the integration to prevent against confused deputy. The partner is responsible for creating and providing a unique external ID.

   You can choose **How to find this?** to view the partner's documentation that describes how to find the external ID.

   External ID

   Enter the unique account identifier provided by Nordcloud. How to find this?

   > **Note**
   >
   > If the resource policy includes an external ID, all calls to the `PutAuditEvents` API must include the external ID. However, if the policy does not define an external ID, the partner can still call the `PutAuditEvents` API and specify an `externalId` parameter.

b. For a solution integration, choose **Add AWS account** to specify each AWS account ID to add as a principal in the policy.

15. Choose **Next** to review your choices.
16. On the **Review and create** page, review your choices. Choose **Edit** to make changes to a section. When you're ready to create the event data store, choose **Create event data store**.
17. The new event data store is visible in the **Event data stores** table on the **Event data stores** page.
18. Provide the channel Amazon Resource Name (ARN) to the partner application. Instructions for providing the channel ARN to the partner application are found on the partner documentation website. For more information, choose the **Learn more** link for the partner on the **Available sources** tab of the **Integrations** page to open the partner's page in AWS Marketplace.

The event data store starts ingesting partner events into CloudTrail through the integration's channel when you, the partner, or the partner applications calls the `PutAuditEvents` API on the channel.

Update an event data store with the console

This section describes how to update an event data store's settings using the AWS Management Console. For information about how to update an event data store using the AWS CLI, see Update an event data store with the AWS CLI.

To update an event data store

1. Sign in to the AWS Management Console and open the CloudTrail console at https://console.aws.amazon.com/cloudtrail/.
2. In the navigation pane, under **Lake**, choose **Event data stores**.
3. Choose the event data store that you want to update. This action opens the event data store's details page.
4. In **General details**, choose **Edit** to change the following settings:

   - **Event data store name** - Change the name that identifies your event data store.
   - **Pricing option**- For event data stores using the **Seven-year retention pricing** option, you can choose to use **One-year extendable retention pricing** instead. We recommend one-year extendable retention pricing for event data stores that ingest less than 25 TB of event data on a monthly basis. We also recommend one-year extendable retention pricing if you're seeking a flexible retention period of up to 10 years. For more information, see AWS CloudTrail Pricing and Managing CloudTrail Lake costs.

     > **Note**
     >
     > You can't change the pricing option for event data stores that use **One-year extendable retention pricing**. If you want to use **Seven-year retention pricing**, stop ingestion on your current event data store. Then create a new event data store with the **Seven-year retention pricing** option.

   - **Retention period** - Change the retention period for the event data store. The retention period determines how long event data is kept in the event data store. Retention periods can be between 7 days and 3,653 days (about 10 years) for the **One-year extendable retention pricing** option, or between 7 days and 2,557 days (about seven years) for the **Seven-year retention pricing** option.

> **Note**
>
> If you decrease the retention period of an event data store, CloudTrail will remove any events with an `eventTime` older than the new retention period. For example, if the previous retention period was 365 days and you decrease it to 100 days, CloudTrail will remove events with an `eventTime` older than 100 days.

- **Encryption** - To encrypt your event data store using your own KMS key, choose **Use my own AWS KMS key**. By default, all events in an event data store are encrypted by CloudTrail. Using your own KMS key incurs AWS KMS costs for encryption and decryption.

  > **Note**
  >
  > After you associate an event data store with a KMS key, the KMS key can't be removed or changed.

- To include only events that are logged in the current AWS Region, choose **Include on the current region in my event data store**. If you don't choose this option, your event data store includes events from all Regions.
- To have your event data store collect events from all accounts in an AWS Organizations organization, choose **Enable for all accounts in my organization**. This option is only available if you're signed in with the management account for your organization, and the **Event type** for the event data store is **CloudTrail events** or **Configuration items**.

Choose **Save changes** when you're finished.

5. In **Lake query federation**, choose **Edit** to enable or disable Lake query federation. Enabling Lake query federation lets you view the metadata for your event data store in the AWS Glue Data Catalog and run SQL queries on the event data using Amazon Athena. Disabling Lake query federation disables the integration with AWS Glue, AWS Lake Formation, and Amazon Athena. After disabling Lake query federation, you can no longer query your data in Athena. No CloudTrail Lake data is deleted when you disable federation and you can continue to run queries in CloudTrail Lake.

   To enable federation, do the following:

   a. Choose **Enable**.

   b. Choose whether to create a new IAM role, or use an existing role. When you create a new role, CloudTrail automatically creates a role with the required permissions. If you're using an existing role, be sure the role's policy provides the required minimum permissions.

   c. If you're creating a new IAM role, enter a name for the role.

   d. If you're choosing an existing IAM role, choose the role you want to use. The role must exist in your account.

   Choose **Save changes** when you are finished.

6. Edit any additional settings for your **Event type**.

Event type	Editable settings
CloudTrail events	You can edit the following settings for CloudTrail events: • To change which events your event data store logs, choose **Edit** in **CloudTrail events**. • In **Management events**, choose **Edit** to change the settings for managemen t events. For more information, see Logging management events with the AWS Management Console (step 3). • In **Data events**, choose **Edit** to change the settings for data events. You can choose which data event types you want to log and choose the log selector template you want to use. For more information, see Updating an existing event data store to log data events in the AWS Management Console. Choose **Save changes** when you're finished.

Event type	Editable settings
Events from integration	In **Integrations**, choose your integration. Then choose **Edit** to change the following settings: • In **Integration details**, change the name that identifies your integration's channel. • In **Event delivery location**, choose the destination for your events. • In **Resource policy**, configure the resource policy for the integration's channel. Choose **Save changes** when you're finished. For more information about these settings, see Create an integration with an event source outside of AWS.

7. To add, change, or remove tags, choose **Edit** in **Tags**. You can add up to 50 tag key pairs to help you identify, sort, and control access to your event data store. Choose **Save changes** when you're finished.

Stop and start event ingestion with the console

By default, event data stores are configured to ingest events. You can stop an event data store from ingesting events by using the console, AWS CLI, or APIs.

The options to **Start ingestion** and **Stop ingestion** are only available on event data stores containing either CloudTrail events (management and data events), or AWS Config configuration items.

When you stop ingestion on an event data store, the event data store's state changes to `STOPPED_INGESTION`. You can still run queries on any events already in the event data store. You can also copy trail events to the event data store (if it contains only CloudTrail management or data events).

To stop an event data store from ingesting events

1. Sign in to the AWS Management Console and open the CloudTrail console at https://console.aws.amazon.com/cloudtrail/.
2. In the navigation pane, under **Lake**, choose **Event data stores**.
3. Choose the event data store.
4. From **Actions**, choose **Stop ingestion**.
5. When you are prompted to confirm, choose **Stop ingestion**. The event data store will stop ingesting live events.
6. To resume ingestion, choose **Start ingestion**.

To restart event ingestion

1. Sign in to the AWS Management Console and open the CloudTrail console at https://console.aws.amazon.com/cloudtrail/.
2. In the navigation pane, under **Lake**, choose **Event data stores**.
3. Choose the event data store.
4. From **Actions**, choose **Start ingestion**.

Change termination protection with the console

By default, event data stores in AWS CloudTrail Lake are configured with termination protection enabled. Termination protection prevents an event data store from accidental deletion. If you want to delete the event data store, you must disable termination protection. You can disable termination protection by using the AWS Management Console, AWS CLI, or API operations.

To turn off termination protection

1. Sign in to the AWS Management Console and open the CloudTrail console at https://console.aws.amazon.com/cloudtrail/.
2. In the navigation pane, under **Lake**, choose **Event data stores**.
3. Choose the event data store.
4. From **Actions**, choose **Change termination protection**.
5. Choose **Disabled**.
6. Choose **Save**. You can now delete the event data store.

To turn on termination protection

1. Sign in to the AWS Management Console and open the CloudTrail console at https://console.aws.amazon.com/cloudtrail/.
2. In the navigation pane, under **Lake**, choose **Event data stores**.
3. Choose the event data store.
4. From **Actions**, choose **Change termination protection**.
5. To turn on termination protection, choose **Enabled**.
6. Choose **Save**.

Delete an event data store with the console

This section describes how to delete an event data store using the AWS CloudTrail console. For information about how to delete an event data store using the AWS CLI, see Delete an event data store with the AWS CLI.

> **Note**
>
> You can't delete an event data store if either termination protection or Lake query federation is enabled. By default, CloudTrail enables termination protection to protect an event data store from being accidentally deleted.
> To delete an event data store with an event type of **Events from integration**, you must first delete the integration's channel. You can delete the channel from the integration's details page or by using the **aws cloudtrail delete-channel** command. For more information, see Delete a channel to delete an integration with the AWS CLI

To delete an event data store

1. Sign in to the AWS Management Console and open the CloudTrail console at https://console.aws.amazon.com/cloudtrail/.
2. In the navigation pane, under **Lake**, choose **Event data stores**.
3. Choose the event data store.
4. From **Actions**, choose **Delete**.
5. Type the name of the event data store to confirm that you want to delete it.
6. Choose **Delete**.

After you delete an event data store, the event data store's status changes to `PENDING_DELETION` and remains in that state for 7 days. You can restore an event data store during the 7-day wait period. While in the `PENDING_DELETION` state, an event data store isn't available for queries, and no other operations can be performed on the event data store except restore operations. An event data store that is pending deletion does not ingest events and does not incur costs. Event data stores that are pending deletion count toward the quota of event data stores that can exist in one AWS Region.

Restore an event data store with the console

After you delete an event data store in AWS CloudTrail Lake, its status changes to `PENDING_DELETION` and remains in that state for 7 days. During this time, you can restore the event data store by using the AWS Management Console, AWS CLI, or the RestoreEventDataStore API operation.

This section describes how to restore an event data store using the console. For information about how to restore an event data store using the AWS CLI, see Restore an event data store with the AWS CLI.

To restore an event data store

1. Sign in to the AWS Management Console and open the CloudTrail console at https://console.aws.amazon.com/cloudtrail/.
2. In the navigation pane, under **Lake**, choose **Event data stores**.
3. Choose the event data store.
4. From **Actions**, choose **Restore**.

Create, update, and manage event data stores with the AWS CLI

This section describes the AWS CLI commands you can use to create, update, and manage your CloudTrail Lake event data stores.

When using the AWS CLI, remember that your commands run in the AWS Region configured for your profile. If you want to run the commands in a different Region, either change the default Region for your profile, or use the **--region** parameter with the command.

Available commands for event data stores

Commands for creating and updating event data stores in CloudTrail Lake include:

- `create-event-data-store` to create an event data store.
- `get-event-data-store` to return information about the event data store including the advanced event selectors configured for the event data store.
- `update-event-data-store` to change the configuration of an existing event data store.
- `list-event-data-stores` to list the event data stores.
- `delete-event-data-store` to delete an event data store.
- `restore-event-data-store` to restore an event data store that is pending deletion.
- `start-import` to start an import of trail events to an event data store, or retry a failed import.
- `get-import` to return information about a specific import.
- `stop-import` to stop an import of trail events to an event data store.
- `list-imports` to return information on all imports, or a select set of imports by `ImportStatus` or `Destination`.
- `list-import-failures` to list import failures for the specified import.
- `stop-event-data-store-ingestion` to stop event ingestion on an event data store.
- `start-event-data-store-ingestion` to restart event ingestion on an event data store.
- `enable-federation` to enable federation on an event data store to query the event data store in Amazon Athena.
- `disable-federation` to disable federation on an event data store. After you disable federation, you can no longer query against the event data store's data in Amazon Athena. You can continue to query in CloudTrail Lake.
- `put-insight-selectors` to add or modify Insights event selectors for an existing event data store, and enable or disable Insights events.
- `get-insight-selectors` to return information about Insights event selectors configured for an event data store.
- `add-tags` to add one or more tags (key-value pairs) to an existing event data store.
- `remove-tags` to remove one or more tags from a event data store.
- `list-tags` to return a list of tags associated with a event data store.

For a list of available commands for CloudTrail Lake queries, see Available commands for CloudTrail Lake queries.

For a list of available commands for CloudTrail Lake integrations, see Available commands for CloudTrail Lake integrations.

Create an event data store with the AWS CLI

This section describes how to use the `create-event-data-store` command to create an event data store and provides examples of different types of event data stores that you can create.

When you create an event data store, the only required parameter is `--name`, which is used to identify the event data store. You can configure additional optional parameters, including:

- `--advanced-event-selectors` - Specifies the type of events to include in the event data store. By default, event data stores log all management events. For more information about advanced event selectors, see AdvancedEventSelector in the CloudTrail API Reference.
- `--kms-key-id` - Specifies the KMS key ID to use to encrypt the events delivered by CloudTrail. The value can be an alias name prefixed by `alias/`, a fully specified ARN to an alias, a fully specified ARN to a key, or a globally unique identifier.
- `--multi-region-enabled` - Creates a multi-Region event data store that logs events for all AWS Regions in your account. By default, `--multi-region-enabled` is set, even if the parameter is not added.
- `--organization-enabled` - Enables an event data store to collect events for all accounts in an organization. By default, the event data store is not enabled for all accounts in an organization.
- `--billing-mode` - Determines the cost for ingesting and storing events, and the default and maximum retention period for the event data store.

  The following are the possible values:

  - `EXTENDABLE_RETENTION_PRICING` - This billing mode is generally recommended if you ingest less than 25 TB of event data a month and want a flexible retention period of up to 3653 days (about 10 years). The default retention period for this billing mode is 366 days.
  - `FIXED_RETENTION_PRICING` - This billing mode is recommended if you expect to ingest more than 25 TB of event data per month and need a retention period of up to 2557 days (about 7 years). The default retention period for this billing mode is 2557 days.

  The default value is `EXTENDABLE_RETENTION_PRICING`.
- `--retention-period` - The number of days to keep events in the event data store. Valid values are integers between 7 and 3653 if the `--billing-mode` is `EXTENDABLE_RETENTION_PRICING`, or between 7 and 2557 if the `--billing-mode` is set to `FIXED_RETENTION_PRICING`. If you do not specify `--retention-period`, CloudTrail uses the default retention period for the `--billing-mode`.

- `--start-ingestion` - The `--start-ingestion` parameter starts event ingestion on the event data store when it's created. This parameter is set even if the parameter is not added.

  Specify the `--no-start-ingestion` if you do not want the event data store to ingest live events. For example, you may want to set this parameter if you are copying events to the event data store and only plan to use the event data to analyze past events. The `--no-start-ingestion` parameter is only valid if the `eventCategory` is `Management`, `Data`, or `ConfigurationItem`.

The following examples show how to create different types of event data stores.

Examples:

- Create an event data store for S3 data events with the AWS CLI
- Create an event data store for AWS Config configuration items with the AWS CLI
- Create an organization event data store for management events with the AWS CLI
- Create event data stores for Insights events with the AWS CLI

Create an event data store for S3 data events with the AWS CLI

The following example AWS Command Line Interface (AWS CLI) **create-event-data-store** command creates an event data store named `my-event-data-store` that selects all Amazon S3 data events and is encrypted using a KMS key.

```
aws cloudtrail create-event-data-store \
--name my-event-data-store \
--kms-key-id "arn:aws:kms:us-east-1:123456789012:alias/KMS_key_alias" \
--advanced-event-selectors '[
        {
            "Name": "Select all S3 data events",
            "FieldSelectors": [
                { "Field": "eventCategory", "Equals": ["Data"] },
                { "Field": "resources.type", "Equals": ["AWS::S3::Object"] },
                { "Field": "resources.ARN", "StartsWith": ["arn:aws:s3"] }
            ]
        }
    ]'
```

The following is an example response.

```
{
    "EventDataStoreArn": "arn:aws:cloudtrail:us-east-1:123456789012:eventdatastore/
EXAMPLE-ee54-4813-92d5-999aeEXAMPLE",
    "Name": "my-event-data-store",
    "Status": "CREATED",
    "AdvancedEventSelectors": [
        {
            "Name": "Select all S3 data events",
            "FieldSelectors": [
                {
                    "Field": "eventCategory",
                    "Equals": [
                        "Data"
                    ]
                },
                {
                    "Field": "resources.type",
                    "Equals": [
                        "AWS::S3::Object"
                    ]
                },
                {
                    "Field": "resources.ARN",
                    "StartsWith": [
                        "arn:aws:s3"
                    ]
                }
            ]
        }
    ],
    "MultiRegionEnabled": true,
    "OrganizationEnabled": false,
    "BillingMode": "EXTENDABLE_RETENTION_PRICING",
    "RetentionPeriod": 366,
    "KmsKeyId": "arn:aws:kms:us-east-1:123456789012:alias/KMS_key_alias",
    "TerminationProtectionEnabled": true,
    "CreatedTimestamp": "2023-11-09T22:19:39.417000-05:00",
    "UpdatedTimestamp": "2023-11-09T22:19:39.603000-05:00"
}
```

Create an event data store for AWS Config configuration items with the AWS CLI

The following example AWS CLI **create-event-data-store** command creates an event data store named `config-items-eds` that selects AWS Config configuration items. To collect configuration items, specify that the `eventCategory` field Equals `ConfigurationItem` in the advanced event selectors.

```
aws cloudtrail create-event-data-store \
--name config-items-eds \
--advanced-event-selectors '[
    {
        "Name": "Select AWS Config configuration items",
        "FieldSelectors": [
            { "Field": "eventCategory", "Equals": ["ConfigurationItem"] }
        ]
    }
]'
```

The following is an example response.

```
{
    "EventDataStoreArn": "arn:aws:cloudtrail:us-east-1:123456789012:eventdatastore/
EXAMPLE-ee54-4813-92d5-999aeEXAMPLE",
    "Name": "config-items-eds",
    "Status": "CREATED",
    "AdvancedEventSelectors": [
        {
            "Name": "Select AWS Config configuration items",
            "FieldSelectors": [
                {
                    "Field": "eventCategory",
                    "Equals": [
                        "ConfigurationItem"
                    ]
                }
            ]
        }
    ],
    "MultiRegionEnabled": true,
    "OrganizationEnabled": false,
    "BillingMode": "EXTENDABLE_RETENTION_PRICING",
    "RetentionPeriod": 366,
    "TerminationProtectionEnabled": true,
    "CreatedTimestamp": "2023-11-07T19:03:24.277000+00:00",
    "UpdatedTimestamp": "2023-11-07T19:03:24.468000+00:00"
}
```

Create an organization event data store for management events with the AWS CLI

The following example AWS CLI **create-event-data-store** command creates an organization event data store that collects all management events and sets the `--billing-mode` parameter to `FIXED_RETENTION_PRICING`.

```
aws cloudtrail create-event-data-store --name org-management-eds --organization-enabled
 --billing-mode FIXED_RETENTION_PRICING
```

The following is an example response.

```
{
    "EventDataStoreArn": "arn:aws:cloudtrail:us-east-1:123456789012:eventdatastore/
EXAMPLE6-d493-4914-9182-e52a7934b207",
    "Name": "org-management-eds",
    "Status": "CREATED",
    "AdvancedEventSelectors": [
        {
            "Name": "Default management events",
            "FieldSelectors": [
                {
                    "Field": "eventCategory",
                    "Equals": [
                        "Management"
                    ]
                }
            ]
        }
    ],
    "MultiRegionEnabled": true,
    "OrganizationEnabled": true,
    "BillingMode": "FIXED_RETENTION_PRICING",
    "RetentionPeriod": 2557,
    "TerminationProtectionEnabled": true,
    "CreatedTimestamp": "2023-11-16T15:30:50.689000+00:00",
    "UpdatedTimestamp": "2023-11-16T15:30:50.851000+00:00"
}
```

Create event data stores for Insights events with the AWS CLI

To log Insights events in CloudTrail Lake, you need a destination event data store that collects Insights events and a source event data store that enables Insights and logs management events.

This procedure shows you how to create the destination and source event data stores and then enable Insights events.

1. Run the **aws cloudtrail create-event-data-store** command to create a destination event data store that collects Insights events. The value for `eventCategory` must be `Insight`. Replace *`retention-period-days`* with the number of days you would like to retain events in your event data store. Valid values are integers between 7 and 3653 if the `--billing-mode` is `EXTENDABLE_RETENTION_PRICING`, or between 7 and 2557 if the `--billing-mode` is set to `FIXED_RETENTION_PRICING`. If you do not specify `--retention-period`, CloudTrail uses the default retention period for the `--billing-mode`.

   If you are signed in with the management account for an AWS Organizations organization, include the `--organization-enabled` parameter if you want to give your delegated administrator access to the event data store.

   ```
   aws cloudtrail create-event-data-store \
   --name insights-event-data-store \
   --no-multi-region-enabled \
   --retention-period retention-period-days \
   --advanced-event-selectors '[
       {
         "Name": "Select Insights events",
         "FieldSelectors": [
             { "Field": "eventCategory", "Equals": ["Insight"] }
           ]
       }
     ]'
   ```

   The following is an example response.

   ```
   {
       "Name": "insights-event-data-store",
       "ARN": "arn:aws:cloudtrail:us-east-1:111122223333:eventdatastore/
   EXAMPLEf852-4e8f-8bd1-bcf6cEXAMPLE",
       "AdvancedEventSelectors": [
   ```

```
        {
           "Name": "Select Insights events",
           "FieldSelectors": [
              {
                  "Field": "eventCategory",
                  "Equals": [
                      "Insight"
                    ]
                }
            ]
        }
    ],
    "MultiRegionEnabled": false,
    "OrganizationEnabled": false,
    "BillingMode": "EXTENDABLE_RETENTION_PRICING",
    "RetentionPeriod": "90",
    "TerminationProtectionEnabled": true,
    "CreatedTimestamp": "2023-05-08T15:22:33.578000+00:00",
    "UpdatedTimestamp": "2023-05-08T15:22:33.714000+00:00"
}
```

You will use the ARN (or ID suffix of the ARN) from the response as the value for the `--insights-destination` parameter in step 3.

2. Run the **aws cloudtrail create-event-data-store** command to create a source event data store that logs management events. By default, event data stores log all management events. You don't need to specify the advanced event selectors if you want to log all management events. Replace *`retention-period-days`* with the number of days you would like to retain events in your event data store. Valid values are integers between 7 and 3653 if the `--billing-mode` is `EXTENDABLE_RETENTION_PRICING`, or between 7 and 2557 if the `--billing-mode` is set to `FIXED_RETENTION_PRICING`. If you do not specify `--retention-period`, CloudTrail uses the default retention period for the `--billing-mode`. If you are creating an organization event data store, include the `--organization-enabled` parameter.

```
aws cloudtrail create-event-data-store --name source-event-data-store --retention-period retention-period-days
```

The following is an example response.

```
{
    "EventDataStoreArn": "arn:aws:cloudtrail:us-east-1:111122223333:eventdatastore/
EXAMPLE9952-4ab9-49c0-b788-f4f3EXAMPLE",
    "Name": "source-event-data-store",
    "Status": "CREATED",
    "AdvancedEventSelectors": [
        {
            "Name": "Default management events",
            "FieldSelectors": [
                {
                    "Field": "eventCategory",
                    "Equals": [
                        "Management"
                    ]
                }
            ]
        }
    ],
    "MultiRegionEnabled": true,
    "OrganizationEnabled": false,
    "BillingMode": "EXTENDABLE_RETENTION_PRICING",
    "RetentionPeriod": 90,
    "TerminationProtectionEnabled": true,
    "CreatedTimestamp": "2023-05-08T15:25:35.578000+00:00",
    "UpdatedTimestamp": "2023-05-08T15:25:35.714000+00:00"
}
```

You will use the ARN (or ID suffix of the ARN) from the response as the value for the `--event-data-store` parameter in step 3.

3. Run the **put-insight-selectors** command to enable Insights events. Insights selector values can be `ApiCallRateInsight`, `ApiErrorRateInsight`, or both. For the `--event-data-store` parameter, specify the ARN (or ID suffix of the ARN) of the source event data store that logs management events and will enable Insights. For the `--insights-destination` parameter, specify the ARN (or ID suffix of the ARN) of the destination event data store that will log Insights events.

```
aws cloudtrail put-insight-selectors --event-data-store arn:aws:cloudtrail:us-
east-1:111122223333:eventdatastore/EXAMPLE9952-4ab9-49c0-b788-f4f3EXAMPLE --
insights-destination arn:aws:cloudtrail:us-east-1:111122223333:eventdatastore/
EXAMPLEf852-4e8f-8bd1-bcf6cEXAMPLE --insight-selectors '[{"InsightType":
 "ApiCallRateInsight"},{"InsightType": "ApiErrorRateInsight"}]'
```

The following result shows the Insights event selector that is configured for the event data store.

```
{
  "EventDataStoreARN": "arn:aws:cloudtrail:us-east-1:111122223333:eventdatastore/
EXAMPLE9952-4ab9-49c0-b788-f4f3EXAMPLE",
  "InsightsDestination": "arn:aws:cloudtrail:us-east-1:111122223333:eventdatastore/
EXAMPLEf852-4e8f-8bd1-bcf6cEXAMPLE",
  "InsightSelectors":
      [
         {
            "InsightType": "ApiErrorRateInsight"
         },
         {
            "InsightType": "ApiCallRateInsight"
         }
      ]
}
```

After you enable CloudTrail Insights for the first time on an event data store, it can take up to 7 days for CloudTrail to deliver the first Insights event, if unusual activity is detected.

CloudTrail Insights analyzes management events that occur in a single Region, not globally. A CloudTrail Insights event is generated in the same Region as its supporting management events are generated.

For an organization event data store, CloudTrail analyzes management events from each member's account instead of analyzing the aggregation of all management events for the organization.

Additional charges apply for ingesting Insights events in CloudTrail Lake. You will be charged separately if you enable Insights for both trails and event data stores. For information about CloudTrail pricing, see AWS CloudTrail Pricing.

Import trail events to an event data store with the AWS CLI

This section shows how to create and configure an event data store by running the create-event-data-store command and then how to import the events to that event data store by using the start-import command. For more information about importing trail events, see Copy trail events to an event data store.

Preparing to import trail events

Before you import trail events, make the following preparations.

- Be sure you have a role with the required permissions to import trail events to an event data store.
- Determine the --billing-mode value you want to specify for the event data store. The `--billing-mode` determines the cost of ingesting and storing events, and the default and maximum retention period for the event data store.

  When you import trail events to CloudTrail Lake, CloudTrail unzips the logs that are stored in gzip (compressed) format. Then CloudTrail copies the events contained in the logs to your event data store. The size of the uncompressed data could be greater than the actual Amazon S3 storage size. To get a general estimate of the size of the uncompressed data, multiply the size of the logs in the S3 bucket by 10. You can use this estimate to choose the `--billing-mode` value for your use case.
- Determine the value you want to specify for the `--retention-period`. CloudTrail will not copy an event if its `eventTime` is older than the specified retention period.

  To determine the appropriate retention period, take the sum of the oldest event you want to copy in days and the number of days you want to retain the events in the event data store as demonstrated in this equation:

  Retention period = *`oldest-event-in-days`* + *`number-days-to-retain`*

  For example, if the oldest event you're copying is 45 days old and you want to keep the events in the event data store for a further 45 days, you would set the retention period to 90 days.
- Decide whether you want to use the event data store to analyze any future events. If you don't want to ingest any future events, include the `--no-start-ingestion` parameter when you create the event data store. By default, event data store's begin ingesting events when they're created.

To create an event data store and import trail events to that event data store

1. Run the **create-event-data-store** command to create the new event data store. In this example, the `--retention-period` is set to `120` because the oldest event being copied is 90 days old and we want to retain the events for 30 days. The `--no-start-ingestion` parameter is set because we don't want to ingest any future events. In this example, `--billing-mode` wasn't set, because we are using the default value `EXTENDABLE_RETENTION_PRICING` as we expect to ingest less than 25 TB of event data.

   > **Note**
   >
   > If you're creating the event data store to replace your trail, we recommend configuring the `--advanced-event-selectors` to match the event selectors of your trail to ensure you have the same event coverage. By default, event data stores log all management events.

   ```
   aws cloudtrail create-event-data-store  --name import-trail-eds  --retention-period
    120 --no-start-ingestion
   ```

   The following is the example response:

   ```
   {
       "EventDataStoreArn": "arn:aws:cloudtrail:us-east-1:123456789012:eventdatastore/
   EXAMPLEa-4357-45cd-bce5-17ec652719d9",
       "Name": "import-trail-eds",
       "Status": "CREATED",
       "AdvancedEventSelectors": [
           {
               "Name": "Default management events",
               "FieldSelectors": [
                   {
                       "Field": "eventCategory",
                       "Equals": [
                           "Management"
                       ]
                   }
               ]
           }
       ],
   ```

```
    "MultiRegionEnabled": true,
    "OrganizationEnabled": false,
    "BillingMode": "EXTENDABLE_RETENTION_PRICING",
    "RetentionPeriod": 120,
    "TerminationProtectionEnabled": true,
    "CreatedTimestamp": "2023-11-09T16:52:25.444000+00:00",
    "UpdatedTimestamp": "2023-11-09T16:52:25.569000+00:00"
}
```

The initial `Status` is `CREATED` so we'll run the **get-event-data-store** command to verify ingestion is stopped.

```
aws cloudtrail get-event-data-store --event-data-store eds-id
```

The response shows the `Status` is now `STOPPED_INGESTION`, which indicates the event data store is not ingesting live events.

```
{
    "EventDataStoreArn": "arn:aws:cloudtrail:us-east-1:123456789012:eventdatastore/
EXAMPLEa-4357-45cd-bce5-17ec652719d9",
    "Name": "import-trail-eds",
    "Status": "STOPPED_INGESTION",
    "AdvancedEventSelectors": [
        {
            "Name": "Default management events",
            "FieldSelectors": [
                {
                    "Field": "eventCategory",
                    "Equals": [
                        "Management"
                    ]
                }
            ]
        }
    ],
    "MultiRegionEnabled": true,
    "OrganizationEnabled": false,
    "BillingMode": "EXTENDABLE_RETENTION_PRICING",
    "RetentionPeriod": 120,
    "TerminationProtectionEnabled": true,
    "CreatedTimestamp": "2023-11-09T16:52:25.444000+00:00",
    "UpdatedTimestamp": "2023-11-09T16:52:25.569000+00:00"
}
```

2. Run the **start-import** command to import the trail events to the event data store created in step 1. Specify the ARN (or ID suffix of the ARN) of the event data store as the value for the `--destinations` parameter. For `--start-event-time` specify the `eventTime` for the oldest event you want to copy and for `--end-event-time` specify the `eventTime` of the newest event you want to copy. For `--import-source` specify the S3 URI for the S3 bucket containing your trail logs, the AWS Region for the S3 bucket, and the ARN of the role used for importing trail events.

```
aws cloudtrail start-import \
--destinations ["arn:aws:cloudtrail:us-east-1:123456789012:eventdatastore/
EXAMPLEa-4357-45cd-bce5-17ec652719d9"] \
--start-event-time 2023-08-11T16:08:12.934000+00:00 \
--end-event-time 2023-11-09T17:08:20.705000+00:00 \
--import-source {"S3": {"S3LocationUri": "s3://aws-cloudtrail-
logs-123456789012-612ff1f6/AWSLogs/123456789012/CloudTrail/","S3BucketRegion":"us-
east-1","S3BucketAccessRoleArn": "arn:aws:iam::123456789012:role/service-role/
CloudTrailLake-us-east-1-copy-events-eds"}}
```

The following is an example response.

```
{
   "CreatedTimestamp": "2023-11-09T17:08:20.705000+00:00",
   "Destinations": [
        "arn:aws:cloudtrail:us-east-1:123456789012:eventdatastore/
EXAMPLEa-4357-45cd-bce5-17ec652719d9"
    ],
   "EndEventTime": "2023-11-09T17:08:20.705000+00:00",
   "ImportId": "EXAMPLEe-7be2-4658-9204-b38c3257fcd1",
   "ImportSource": {
      "S3": {
         "S3BucketAccessRoleArn": "arn:aws:iam::123456789012:role/service-role/
CloudTrailLake-us-east-1-copy-events-eds",
         "S3BucketRegion":"us-east-1",
         "S3LocationUri": "s3://aws-cloudtrail-logs-123456789012-111ff1f6/
AWSLogs/123456789012/CloudTrail/"
      }
   },
   "ImportStatus": "INITIALIZING",
   "StartEventTime": "2023-08-11T16:08:12.934000+00:00",
   "UpdatedTimestamp": "2023-11-09T17:08:20.806000+00:00"
}
```

3. Run the get-import command to get information about the import.

   ```
   aws cloudtrail get-import --import-id import-id
   ```

   The following is an example response.

   ```
   {
       "ImportId": "EXAMPLEe-7be2-4658-9204-b38c3EXAMPLE",
       "Destinations": [
           "arn:aws:cloudtrail:us-east-1:123456789012:eventdatastore/
   EXAMPLEa-4357-45cd-bce5-17ec652719d9"
       ],
       "ImportSource": {
           "S3": {
               "S3LocationUri": "s3://aws-cloudtrail-logs-123456789012-111ff1f6/
   AWSLogs/123456789012/CloudTrail/",
               "S3BucketRegion":"us-east-1",
               "S3BucketAccessRoleArn": "arn:aws:iam::123456789012:role/service-role/
   CloudTrailLake-us-east-1-copy-events-eds"
           }
       },
       "StartEventTime": "2023-08-11T16:08:12.934000+00:00",
       "EndEventTime": "2023-11-09T17:08:20.705000+00:00",
       "ImportStatus": "COMPLETED",
       "CreatedTimestamp": "2023-11-09T17:08:20.705000+00:00",
       "ImportStatistics": {
           "PrefixesFound": 1548,
           "PrefixesCompleted": 1548,
           "FilesCompleted": 92845,
           "EventsCompleted": 577249,
           "FailedEntries": 0
       }
   }
   ```

   An import finishes with an `ImportStatus` of `COMPLETED` if there were no failures, or `FAILED` if there were failures.

   If the import had `FailedEntries`, you can run the list-import-failures command to return a list of failures.

```
aws cloudtrail list-import-failures --import-id import-id
```

To retry an import that had failures, run the **start-import** command with only the `--import-id` parameter. When you retry an import, CloudTrail resumes the import at the location where the failure occurred.

```
aws cloudtrail start-import --import-id import-id
```

Update an event data store with the AWS CLI

This section provides examples that show how to update an event data store's settings by running the AWS CLI `update-event-data-store` command.

Examples:

- Update the billing mode with the AWS CLI
- Update the retention mode, enable termination protection, and specify a AWS KMS key with the AWS CLI
- Disable termination protection with the AWS CLI

Update the billing mode with the AWS CLI

The `--billing-mode` for the event data store determines the cost for ingesting and storing events, and the default and maximum retention period for the event data store. If an event data store's `--billing-mode` is set to `FIXED_RETENTION_PRICING`, you can change the value to `EXTENDABLE_RETENTION_PRICING`. `EXTENDABLE_RETENTION_PRICING` is generally recommended if your event data store ingests less than 25 TB of event data per month and you want a flexible retention period of up to 3653 days. For information about pricing, see AWS CloudTrail Pricing and Managing CloudTrail Lake costs.

Note

You cannot change the `--billing-mode` value from `EXTENDABLE_RETENTION_PRICING` to `FIXED_RETENTION_PRICING`. If the event data store's billing mode is set to `EXTENDABLE_RETENTION_PRICING` and you want to use `FIXED_RETENTION_PRICING` instead, you can stop ingestion on the event data store and create a new event data store that uses FIXED_RETENTION_PRICING.

The following example AWS CLI **update-event-data-store** command changes the --billing-mode for the event data store from FIXED_RETENTION_PRICING to EXTENDABLE_RETENTION_PRICING. The required --event-data-store parameter value is an ARN (or the ID suffix of the ARN) and is required; other parameters are optional.

```
aws cloudtrail update-event-data-store \
--region us-east-1 \
--event-data-store arn:aws:cloudtrail:us-east-1:123456789012:eventdatastore/EXAMPLE-
f852-4e8f-8bd1-bcf6cEXAMPLE \
--billing-mode EXTENDABLE_RETENTION_PRICING
```

The following is an example response.

```
{
    "EventDataStoreArn": "event-data-store arn:aws:cloudtrail:us-
east-1:123456789012:eventdatastore/EXAMPLE-f852-4e8f-8bd1-bcf6cEXAMPLE",
    "Name": "management-events-eds",
    "Status": "ENABLED",
    "AdvancedEventSelectors": [
        {
            "Name": "Default management events",
            "FieldSelectors": [
                {
                    "Field": "eventCategory",
                    "Equals": [
                        "Management"
                    ]
                }
            ]
        }
    ],
    "MultiRegionEnabled": true,
    "OrganizationEnabled": false,
    "BillingMode": "EXTENDABLE_RETENTION_PRICING",
    "RetentionPeriod": 2557,
    "TerminationProtectionEnabled": true,
    "CreatedTimestamp": "2023-10-27T10:55:55.384000-04:00",
    "UpdatedTimestamp": "2023-10-27T10:57:05.549000-04:00"
}
```

Update the retention mode, enable termination protection, and specify a AWS KMS key with the AWS CLI

The following example AWS CLI **update-event-data-store** command updates an event data store to change its retention period to 100 days, and enable termination protection. The required `--event-data-store` parameter value is an ARN (or the ID suffix of the ARN) and is required; other parameters are optional. In this example, the `--retention-period` parameter is added to change the retention period to 100 days. Optionally, you can choose to enable AWS Key Management Service encryption and specify an AWS KMS key by adding `--kms-key-id` to the command, and specifying a KMS key ARN as the value. `--termination-protection-enabled` is added to enable termination protection on an event data store that did not have termination protection enabled.

An event data store that logs events from outside AWS cannot be updated to log AWS events. Similarly, an event data store that logs AWS events cannot be updated to log events from outside AWS.

> **Note**
>
> If you decrease the retention period of an event data store, CloudTrail will remove any events with an `eventTime` older than the new retention period. For example, if the previous retention period was 365 days and you decrease it to 100 days, CloudTrail will remove events with an `eventTime` older than 100 days.

```
aws cloudtrail update-event-data-store \
--event-data-store arn:aws:cloudtrail:us-east-1:123456789012:eventdatastore/EXAMPLE-
f852-4e8f-8bd1-bcf6cEXAMPLE \
--retention-period 100 \
--kms-key-id "arn:aws:kms:us-east-1:0123456789:alias/KMS_key_alias" \
--termination-protection-enabled
```

The following is an example response.

```
{
    "EventDataStoreArn": "arn:aws:cloudtrail:us-east-1:123456789012:eventdatastore/
EXAMPLE-ee54-4813-92d5-999aeEXAMPLE",
    "Name": "my-event-data-store",
    "Status": "ENABLED",
    "AdvancedEventSelectors": [
        {
            "Name": "Select all S3 data events",
            "FieldSelectors": [
                {
                    "Field": "eventCategory",
                    "Equals": [
                        "Data"
                    ]
                },
                {
                    "Field": "resources.type",
                    "Equals": [
                        "AWS::S3::Object"
                    ]
                },
                {
                    "Field": "resources.ARN",
                    "StartsWith": [
                        "arn:aws:s3"
                    ]
                }
            ]
        }
    ],
    "MultiRegionEnabled": true,
    "OrganizationEnabled": false,
    "BillingMode": "EXTENDABLE_RETENTION_PRICING",
    "RetentionPeriod": 100,
    "KmsKeyId": "arn:aws:kms:us-east-1:0123456789:alias/KMS_key_alias",
    "TerminationProtectionEnabled": true,
    "CreatedTimestamp": "2023-10-27T10:55:55.384000-04:00",
    "UpdatedTimestamp": "2023-10-27T10:57:05.549000-04:00"
}
```

Disable termination protection with the AWS CLI

By default, termination protection is enabled on an event data store to protect the event data store from accidental deletion. You cannot delete an event data store when termination protection is enabled. If you want to delete the event data store, you must first disable termination protection.

The following example AWS CLI **update-event-data-store** command disables termination protection by passing the `--no-termination-protection-enabled` parameter.

```
aws cloudtrail update-event-data-store \
--region us-east-1 \
--no-termination-protection-enabled \
--event-data-store arn:aws:cloudtrail:us-east-1:123456789012:eventdatastore/EXAMPLE-
f852-4e8f-8bd1-bcf6cEXAMPLE
```

The following is an example response.

```
{
    "EventDataStoreArn": "arn:aws:cloudtrail:us-east-1:123456789012:eventdatastore/
EXAMPLE-f852-4e8f-8bd1-bcf6cEXAMPLE",
    "Name": "management-events-eds",
    "Status": "ENABLED",
    "AdvancedEventSelectors": [
        {
            "Name": "Default management events",
            "FieldSelectors": [
                {
                    "Field": "eventCategory",
                    "Equals": [
                        "Management"
                    ]
                }
            ]
        }
    ],
    "MultiRegionEnabled": true,
    "OrganizationEnabled": false,
    "BillingMode": "EXTENDABLE_RETENTION_PRICING",
    "RetentionPeriod": 366,
    "TerminationProtectionEnabled": false,
    "CreatedTimestamp": "2023-10-27T10:55:55.384000-04:00",
    "UpdatedTimestamp": "2023-10-27T10:57:05.549000-04:00"
}
```

Managing event data stores with the AWS CLI

This section describes several other commands that you can run to get information about your event data stores, start and stop ingestion on an event data store, and enable and disable federation on an event data store.

Topics

- Get an event data store with the AWS CLI
- List all event data stores in an account with the AWS CLI
- Stop ingestion on an event data store with the AWS CLI
- Start ingestion on an event data store with the AWS CLI
- Enable federation on an event data store
- Disable federation on an event data store
- Restore an event data store with the AWS CLI

Get an event data store with the AWS CLI

The following example AWS CLI **get-event-data-store** command returns information about the event data store specified by the required `--event-data-store` parameter, which accepts an ARN or the ID suffix of the ARN.

```
aws cloudtrail get-event-data-store
--event-data-store arn:aws:cloudtrail:us-east-1:123456789012:eventdatastore/EXAMPLE-
f852-4e8f-8bd1-bcf6cEXAMPLE
```

The following is an example response. Creation and last updated times are in `timestamp` format.

```
{
    "EventDataStoreARN": "arn:aws:cloudtrail:us-east-1:123456789012:eventdatastore/
EXAMPLE-f852-4e8f-8bd1-bcf6cEXAMPLE",
    "Name": "s3-data-events-eds",
    "Status": "ENABLED",
    "AdvancedEventSelectors": [
        {
            "Name": "Log DeleteObject API calls for a specific S3 bucket",
            "FieldSelectors": [
                {
                    "Field": "eventCategory",
                    "Equals": [
                        "Data"
                    ]
                },
                {
                    "Field": "eventName",
                    "Equals": [
                        "DeleteObject"
                    ]
                },
                {
                    "Field": "resources.ARN",
                    "StartsWith": [
                        "arn:aws:s3:::DOC-EXAMPLE-BUCKET"
                    ]
                },
                {
                    "Field": "readOnly",
                    "Equals": [
                        "false"
                    ]
                },
                {
                    "Field": "resources.type",
                    "Equals": [
                        "AWS::S3::Object"
                    ]
                }
            ]
        }
    ],
    "MultiRegionEnabled": true,
    "OrganizationEnabled": false,
    "BillingMode": "FIXED_RETENTION_PRICING",
    "RetentionPeriod": 2557,
    "TerminationProtectionEnabled": true,
    "CreatedTimestamp": "2023-11-09T22:20:36.344000+00:00",
    "UpdatedTimestamp": "2023-11-09T22:20:36.476000+00:00"
}
```

List all event data stores in an account with the AWS CLI

The following example AWS CLI **list-event-data-stores** command returns information about all event data stores in an account, in the current Region. Optional parameters include `--max-`

results, to specify a maximum number of results that you want the command to return on a single page. If there are more results than your specified --max-results value, run the command again adding the returned NextToken value to get the next page of results.

```
aws cloudtrail list-event-data-stores
```

The following is an example response.

```
{
    "EventDataStores": [
        {
            "EventDataStoreArn": "arn:aws:cloudtrail:us-
east-1:123456789012:eventdatastore/EXAMPLE7-cad6-4357-a84b-318f9868e969",
            "Name": "management-events-eds"
        },
        {
            "EventDataStoreArn": "arn:aws:cloudtrail:us-
east-1:123456789012:eventdatastore/EXAMPLE6-88e1-43b7-b066-9c046b4fd47a",
            "Name": "config-items-eds"
        },
        {
            "EventDataStoreArn": "arn:aws:cloudtrail:us-
east-1:123456789012:eventdatastore/EXAMPLEf-b314-4c85-964e-3e43b1e8c3b4",
            "Name": "s3-data-events"
        }
    ]
}
```

Stop ingestion on an event data store with the AWS CLI

The following example AWS CLI **stop-event-data-store-ingestion** command stops an event data store from ingesting events. To stop ingestion, the event data store Status must be ENABLED and the eventCategory must be Management, Data, or ConfigurationItem. The event data store is specified by --event-data-store, which accepts an event data store ARN, or the ID suffix of the ARN. After you run **stop-event-data-store-ingestion**, the state of the event data store changes to STOPPED_INGESTION.

The event data store does count towards your account maximum of ten event data stores when its state is STOPPED_INGESTION.

```
aws cloudtrail stop-event-data-store-ingestion
```

```
--event-data-store arn:aws:cloudtrail:us-east-1:123456789012:eventdatastore/EXAMPLE-
f852-4e8f-8bd1-bcf6cEXAMPLE
```

There is no response if the operation is successful.

Start ingestion on an event data store with the AWS CLI

The following example AWS CLI **start-event-data-store-ingestion** command starts event ingestion on an event data store. To start ingestion, the event data store `Status` must be `STOPPED_INGESTION` and the `eventCategory` must be `Management`, `Data`, or `ConfigurationItem`. The event data store is specified by `--event-data-store`, which accepts an event data store ARN, or the ID suffix of the ARN. After you run **start-event-data-store-ingestion**, the state of the event data store changes to `ENABLED`.

```
aws cloudtrail start-event-data-store-ingestion --event-data-store
 arn:aws:cloudtrail:us-east-1:123456789012:eventdatastore/EXAMPLE-f852-4e8f-8bd1-
bcf6cEXAMPLE
```

There is no response if the operation is successful.

Enable federation on an event data store

To enable federation, run the **aws cloudtrail enable-federation** command, providing the required `--event-data-store` and `--role` parameters. For `--event-data-store`, provide the event data store ARN (or the ID suffix of the ARN). For `--role`, provide the ARN for your federation role. The role must exist in your account and provide the required minimum permissions.

```
aws cloudtrail enable-federation
--event-data-store arn:aws:cloudtrail:region:account-id:eventdatastore/eds-id
--role arn:aws:iam::account-id:role/federation-role-name
```

This example shows how a delegated administrator can enable federation on an organization event data store by specifying the ARN of the event data store in the management account and the ARN of the federation role in the delegated administrator account.

```
aws cloudtrail enable-federation
--event-data-store arn:aws:cloudtrail:region:management-account-id:eventdatastore/eds-
id
--role arn:aws:iam::delegated-administrator-account-id:role/federation-role-name
```

Disable federation on an event data store

To disable federation on the event data store, run the **aws cloudtrail disable-federation** command. The event data store is specified by `--event-data-store`, which accepts an event data store ARN or the ID suffix of the ARN.

```
aws cloudtrail disable-federation
--event-data-store arn:aws:cloudtrail:region:account-id:eventdatastore/eds-id
```

> **Note**
>
> If this is an organization event data store, use the account ID for the management account.

Restore an event data store with the AWS CLI

The following example AWS CLI **restore-event-data-store** command restores an event data store that is pending deletion. The event data store is specified by `--event-data-store`, which accepts an event data store ARN or the ID suffix of the ARN. You can only restore a deleted event data store within the seven-day wait period after deletion.

```
aws cloudtrail restore-event-data-store
--event-data-store EXAMPLE-f852-4e8f-8bd1-bcf6cEXAMPLE
```

The response includes information about the event data store, including its ARN, advanced event selectors, and the status of restoration.

Delete an event data store with the AWS CLI

This section demonstrates how to delete an event data store by running the AWS CLI `delete-event-data-store` command

To delete an event data store, specify the `--event-data-store` by providing the event data store ARN, or the ID suffix of the ARN. After you run **delete-event-data-store**, the final state of the event data store is `PENDING_DELETION`, and the event data store is automatically deleted after a wait period of 7 days.

After you run **delete-event-data-store** on an event data store, you cannot run **list-queries**, **describe-query**, or **get-query-results** on queries that are using the disabled data store. The event data store does count towards your account maximum of ten event data stores when it is pending deletion.

> **Note**
>
> You can't delete an event data store if `--termination-protection-enabled` is set or its `FederationStatus` is `ENABLED`.
> To delete an event data store with an `eventCategory` of `ActivityAuditLog`, you must first delete the integration's channel. You can delete the channel by using the `aws cloudtrail delete-channel` command. For more information, see Delete a channel to delete an integration with the AWS CLI.

```
aws cloudtrail delete-event-data-store
--event-data-store arn:aws:cloudtrail:us-east-1:123456789012:eventdatastore/EXAMPLE-
f852-4e8f-8bd1-bcf6cEXAMPLE
```

There is no response if the operation is successful.

Manage event data store lifecycles

The following are the lifecycle stages of an event data store:

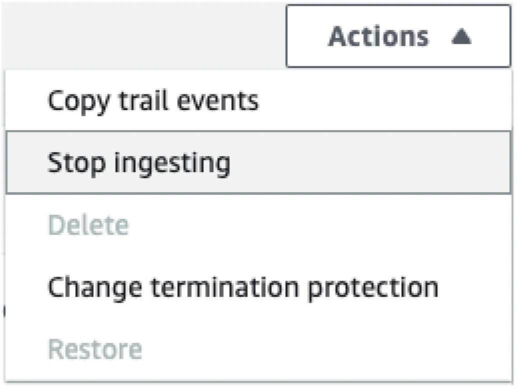
- `CREATED` – A short-term state indicating that the event data store has been created.
- `ENABLED` – The event data store is active and ingesting events. You can run queries and copy trail events to the event data store.
- `STARTING_INGESTION` – A short-term state indicating that the event data store will start ingesting live events.
- `STOPPING_INGESTION` – A short-term state indicating that the event data store will stop ingesting live events.
- `STOPPED_INGESTION` – The event data store is not ingesting live events. You can still run queries on any events already in the event data store and copy trail events to the event data store.
- `PENDING_DELETION` – The event data store was in an `ENABLED` or `STOPPED_INGESTION` state and has been deleted but is within the 7-day wait period before permanent deletion. You cannot run queries on the event data store, and no operations can be performed on the event data store except restoration.

You can only delete an event data store if both federation and termination protection are disabled. *Termination protection* prevents an event data store from getting accidentally deleted. By default, termination protection is enabled on an event data store. Federation lets you query your event data store data in Athena and is disabled by default.

After you delete an event data store, it remains in the `PENDING_DELETION` state for 7 days before it is permanently deleted. You can restore an event data store during the 7-day wait period. While in the `PENDING_DELETION` state, an event data store is not available for queries, and no other operations can be performed on the event data store except restore operations. An event data store that is pending deletion does not ingest events and does not incur costs. However, event data stores that are pending deletion count toward the quota of event data stores that can exist in one AWS Region.

Actions available on event data stores

To delete or restore an event data store, copy trail events, start or stop ingesting events, or turn on or turn off an event data store's termination protection, use commands on the **Actions** menu of the event data store's details page.



The option to **Copy trail events** is only available on event data stores that contain CloudTrail management and data events. The options to **Start ingestion** and **Stop ingestion** are only available on event data stores containing either CloudTrail events (management and data events), or AWS Config configuration items.

Copy trail events to an event data store

You can copy trail events to a CloudTrail Lake event data store to create a point-in-time snapshot of events logged to the trail. Copying a trail's events does not interfere with the trail's ability to log events and does not modify the trail in any way.

You can copy trail events to an existing event data store configured for CloudTrail events, or you can create a new CloudTrail event data store and choose the **Copy trail events** option as part of event data store creation. For more information about copying trail events to an existing event data store, see Copy trail events to an existing event data store with the console. For more information about creating a new event data store, see Create an event data store for CloudTrail events with the console.

If you are copying trail events to an organization event data store, you must use the management account for the organization. You cannot copy trail events using the delegated administrator account for an organization.

CloudTrail Lake event data stores incur charges. When you create an event data store, you choose the pricing option you want to use for the event data store. The pricing option determines the cost for ingesting and storing events, and the default and maximum retention period for the event data store. For information about CloudTrail pricing and managing Lake costs, see AWS CloudTrail Pricing and Managing CloudTrail Lake costs.

When you copy trail events to a CloudTrail Lake event data store, you incur charges based on the amount of uncompressed data the event data store ingests.

When you copy trail events to CloudTrail Lake, CloudTrail unzips the logs that are stored in gzip (compressed) format and then copies the events contained in the logs to your event data store. The size of the uncompressed data could be greater than the actual S3 storage size. To get a general estimate of the size of the uncompressed data, you can multiply the size of the logs in the S3 bucket by 10.

You can reduce costs by specifying a narrower time range for the copied events. If you are planning to only use the event data store to query your copied events, you can turn off event ingestion to avoid incurring charges on future events. For more information, see AWS CloudTrail Pricing and Managing CloudTrail Lake costs.

Scenarios

The following table describes some common scenarios for copying trail events and how you accomplish each scenario using the console.

Scenario	How do I accomplish this in the console?
Analyze and query historical trail events in CloudTrail Lake without ingesting new events	Create a new event data store and choose the **Copy trail events** option as part of event data store creation. When creating the event data store, deselect **Ingest events** (step 15 of the procedure) to ensure the event data store contains only the historical events for your trail and no future events.
Replace your existing trail with a CloudTrail Lake event data store	Create an event data store with the same event selectors as your trail to ensure that the event data store has the same coverage as your trail. To avoid duplicating events between the source trail and destination event data store, choose a date range for the copied events that is earlier than the creation of the event data store. After your event data store is created, you can turn off logging for the trail to avoid additional charges.

Topics

Considerations for copying trail events

Consider the following factors when copying trail events.

- When copying trail events, CloudTrail uses the S3 `GetObject` API operation to retrieve the trail events in the source S3 bucket. There are some S3 archived storage classes, such as S3 Glacier Flexible Retrieval, S3 Glacier Deep Archive, S3 Outposts, and S3 Intelligent-Tiering Deep Archive tiers that are not accessible by using `GetObject`. To copy trail events stored in these archived storage classes, you must first restore a copy using the S3 `RestoreObject` operation. For information about restoring archived objects, see Restoring Archived Objects in the *Amazon S3 User Guide*.

- When you copy trail events to an event data store, CloudTrail copies all trail events regardless of the configuration of the destination event data store's event types, advanced event selectors, or AWS Region.
- Before copying trail events to an existing event data store, be sure the event data store's pricing option and retention period are configured appropriately for your use case.
  - **Pricing option:** The pricing option determines the cost for ingesting and storing events. For more information about pricing options, see AWS CloudTrail Pricing and Event data store pricing options.
  - **Retention period:** The retention period determines how long event data is kept in the event data store. CloudTrail only copies trail events that have an `eventTime` within the event data store’s retention period. To determine the appropriate retention period, take the sum of the oldest event you want to copy in days and the number of days you want to retain the events in the event data store (**retention period** = *`oldest-event-in-days`* + *`number-days-to-retain`*). For example, if the oldest event you're copying is 45 days old and you want to keep the events in the event data store for a further 45 days, you would set the retention period to 90 days.
- If you are copying trail events to an event data store for investigation and do not want to ingest any future events, you can stop ingestion on the event data store. When creating the event data store, deselect the **Ingest events** option (step 15 of the procedure) to ensure the event data store contains only the historical events for your trail and no future events.
- Before copying trail events, disable any access control lists (ACLs) attached to the source S3 bucket, and update the S3 bucket policy for the destination event data store. For more information about updating the S3 bucket policy, see Amazon S3 bucket policy for copying trail events. For more information about disabling ACLs, see Controlling ownership of objects and disabling ACLs for your bucket.
- CloudTrail only copies trail events from Gzip compressed log files that are in the source S3 bucket. CloudTrail does not copy trail events from uncompressed log files, or log files that were compressed using a format other than Gzip.
- To avoid duplicating events between the source trail and destination event data store, choose a time range for the copied events that is earlier than the creation of the event data store.
- By default, CloudTrail only copies CloudTrail events contained in the S3 bucket's `CloudTrail` prefix and the prefixes inside the `CloudTrail` prefix, and does not check prefixes for other AWS

services. If you want to copy CloudTrail events contained in another prefix, you must choose the prefix when you copy trail events.

- To copy trail events to an organization event data store, you must use the management account for the organization. You cannot use the delegated administrator account to copy trail events to an organization event data store.

Required permissions for copying trail events

Before copying trail events, ensure you have all the required permissions for your IAM role. You only need to update the IAM role permissions if you choose an existing IAM role to copy trail events. If you choose to create a new IAM role, CloudTrail provides all necessary permissions for the role.

If the source S3 bucket uses a KMS key for data encryption, ensure that the KMS key policy allows CloudTrail to decrypt data in the bucket. If the source S3 bucket uses multiple KMS keys, you must update each key's policy to allow CloudTrail to decrypt data in the bucket.

Topics

IAM permissions for copying trail events

When copying trail events, you have the option to create a new IAM role, or use an existing IAM role. When you choose a new IAM role, CloudTrail creates an IAM role with the required permissions and no further action is required on your part.

If you choose an existing role, ensure the IAM role's policies allow CloudTrail to copy trail events from the source S3 bucket. This section provides examples of the required IAM role permission and trust policies.

The following example provides the permissions policy, which allows CloudTrail to copy trail events from the source S3 bucket. Replace *`DOC-EXAMPLE-BUCKET`*, *`myAccountID`*, *`region`*, *`prefix`*, and *`eventDataStoreId`* with the appropriate values for your configuration. The *`myAccountID`* is the AWS account ID used for CloudTrail Lake, which may not be the same as the AWS account ID for the S3 bucket.

Replace *`key-region`*, *`keyAccountID`*, and *`keyID`* with the values for the KMS key used to encrypt the source S3 bucket. You can omit the `AWSCloudTrailImportKeyAccess` statement if the source S3 bucket does not use a KMS key for encryption.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AWSCloudTrailImportBucketAccess",
      "Effect": "Allow",
      "Action": ["s3:ListBucket", "s3:GetBucketAcl"],
      "Resource": [
        "arn:aws:s3:::DOC-EXAMPLE-BUCKET"
      ],
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "myAccountID",
          "aws:SourceArn":
 "arn:aws:cloudtrail:region:myAccountID:eventdataStore/eventDataStoreId"
         }
       }
    },
    {
      "Sid": "AWSCloudTrailImportObjectAccess",
      "Effect": "Allow",
      "Action": ["s3:GetObject"],
      "Resource": [
        "arn:aws:s3:::DOC-EXAMPLE-BUCKET/prefix",
        "arn:aws:s3:::DOC-EXAMPLE-BUCKET/prefix/*"
      ],
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "myAccountID",
          "aws:SourceArn":
 "arn:aws:cloudtrail:region:myAccountID:eventdataStore/eventDataStoreId"
         }
       }
    },
    {
      "Sid": "AWSCloudTrailImportKeyAccess",
      "Effect": "Allow",
      "Action": ["kms:GenerateDataKey","kms:Decrypt"],
      "Resource": [
        "arn:aws:kms:key-region:keyAccountID:key/keyID"
      ]
    }
  ]
}
```

The following example provides the IAM trust policy, which allows CloudTrail to assume an IAM role to copy trail events from the source S3 bucket. Replace *myAccountID*, *region*, and *eventDataStoreArn* with the appropriate values for your configuration. The *myAccountID* is the AWS account ID used for CloudTrail Lake, which may not be the same as the AWS account ID for the S3 bucket.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "cloudtrail.amazonaws.com"
      },
      "Action": "sts:AssumeRole",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "myAccountID",
          "aws:SourceArn":
 "arn:aws:cloudtrail:region:myAccountID:eventdataStore/eventDataStoreId"
        }
      }
    }
  ]
}
```

Amazon S3 bucket policy for copying trail events

By default, Amazon S3 buckets and objects are private. Only the resource owner (the AWS account that created the bucket) can access the bucket and objects it contains. The resource owner can grant access permissions to other resources and users by writing an access policy.

Before you copy trail events, you must update the S3 bucket policy to allow CloudTrail to copy trail events from the source S3 bucket.

You can add the following statement to the S3 bucket policy to grant these permissions. Replace *`roleArn`* and *`DOC-EXAMPLE-BUCKET`* with the appropriate values for your configuration.

```
{
  "Sid": "AWSCloudTrailImportBucketAccess",
  "Effect": "Allow",
  "Action": [
    "s3:ListBucket",
    "s3:GetBucketAcl",
    "s3:GetObject"
  ],
  "Principal": {
    "AWS": "roleArn"
  },
  "Resource": [
    "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
    "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
  ]
},
```

KMS key policy for decrypting data in the source S3 bucket

If the source S3 bucket uses a KMS key for data encryption, ensure the KMS key policy provides CloudTrail with the `kms:Decrypt` and `kms:GenerateDataKey` permissions required to copy trail events from an S3 bucket with SSE-KMS encryption enabled. If your source S3 bucket uses multiple KMS keys, you must update each key's policy. Updating the KMS key policy allows CloudTrail to decrypt data in the source S3 bucket, run validation checks to ensure that events conform to CloudTrail standards, and copy events into the CloudTrail Lake event data store.

The following example provides the KMS key policy, which allows CloudTrail to decrypt the data in the source S3 bucket. Replace *`roleArn`*, *`DOC-EXAMPLE-BUCKET`*, *`myAccountID`*, *`region`*, and *`eventDataStoreId`* with the appropriate values for your configuration. The *`myAccountID`* is the AWS account ID used for CloudTrail Lake, which may not be the same as the AWS account ID for the S3 bucket.

```
{
  "Sid": "AWSCloudTrailImportDecrypt",
  "Effect": "Allow",
  "Action": [
          "kms:Decrypt",
          "kms:GenerateDataKey"
  ],
  "Principal": {
    "AWS": "roleArn"
  },
  "Resource": "*",
  "Condition": {
    "StringLike": {
      "kms:EncryptionContext:aws:s3:arn": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
    },
    "StringEquals": {
      "aws:SourceAccount": "myAccountID",
      "aws:SourceArn":
 "arn:aws:cloudtrail:region:myAccountID:eventdataStore/eventDataStoreId"
    }
  }
}
```

Copy trail events to an existing event data store with the console

Use the following procedure to copy trail events to an existing event data store. For information about how to create a new event data store, see Create an event data store for CloudTrail events with the console.

Note

Before copying trail events to an existing event data store, be sure the event data store's pricing option and retention period are configured appropriately for your use case.

- **Pricing option:** The pricing option determines the cost for ingesting and storing events. For more information about pricing options, see AWS CloudTrail Pricing and Event data store pricing options.
- **Retention period:** The retention period determines how long event data is kept in the event data store. CloudTrail only copies trail events that have an `eventTime` within the event data store's retention period. To determine the appropriate retention period, take the sum of the oldest event you want to copy in days and the number of days you want to retain the events in the event data store (**retention period** = *oldest-event-in-days* + *number-days-to-retain*). For example, if the oldest event you're copying is 45

days old and you want to keep the events in the event data store for a further 45 days, you would set the retention period to 90 days.

To copy trail events to an event data store

1. Sign in to the AWS Management Console and open the CloudTrail console at https://console.aws.amazon.com/cloudtrail/.
2. From the navigation pane, under **Lake**, choose **Event data stores**.
3. Choose **Copy trail events**.
4. On the **Copy trail events** page, for **Event source**, choose the trail that you want to copy. By default, CloudTrail only copies CloudTrail events contained in the S3 bucket's `CloudTrail` prefix and the prefixes inside the `CloudTrail` prefix, and does not check prefixes for other AWS services. If you want to copy CloudTrail events contained in another prefix, choose **Enter S3 URI**, and then choose **Browse S3** to browse to the prefix. If the source S3 bucket for the trail uses a KMS key for data encryption, ensure that the KMS key policy allows CloudTrail to decrypt the data. If your source S3 bucket uses multiple KMS keys, you must update each key's policy to allow CloudTrail to decrypt the data in the bucket. For more information about updating the KMS key policy, see KMS key policy for decrypting data in the source S3 bucket.

   The S3 bucket policy must grant CloudTrail access to copy trail events from your S3 bucket. For more information about updating the S3 bucket policy, see Amazon S3 bucket policy for copying trail events.
5. For **Specify a time range of events**, choose the time range for copying the events. CloudTrail checks the prefix and log file name to verify the name contains a date between the chosen start and end date before attempting to copy trail events. You can choose a **Relative range** or an **Absolute range**. To avoid duplicating events between the source trail and destination event data store, choose a time range that is earlier than the creation of the event data store.

   > **Note**
   >
   > CloudTrail only copies trail events that have an `eventTime` within the event data store’s retention period. For example, if an event data store’s retention period is 90 days, then CloudTrail will not copy any trail events with an `eventTime` older than 90 days.

- If you choose **Relative range**, you can choose to copy events logged in the last 6 months, 1 year, 2 years, 7 years, or a custom range. CloudTrail copies the events logged within the chosen time period.
- If you choose **Absolute range**, you can choose a specific start and end date. CloudTrail copies the events that occurred between the chosen start and end dates.

6. For **Delivery location**, choose the destination event data store from the drop-down list.
7. For **Permissions**, choose from the following IAM role options. If you choose an existing IAM role, verify that the IAM role policy provides the necessary permissions. For more information about updating the IAM role permissions, see IAM permissions for copying trail events.

   - Choose **Create a new role (recommended)** to create a new IAM role. For **Enter IAM role name**, enter a name for the role. CloudTrail automatically creates the necessary permissions for this new role.
   - Choose **Use a custom IAM role ARN** to use a custom IAM role that is not listed. For **Enter IAM role ARN**, enter the IAM ARN.
   - Choose an existing IAM role from the drop-down list.
8. Choose **Copy events**.
9. You are prompted to confirm. When you are ready to confirm, choose **Copy trail events to Lake**, and then choose **Copy events**.
10. On the **Copy details** page, you can see the copy status and review any failures. When a trail event copy completes, its **Copy status** is set to either **Completed** if there were no errors, or **Failed** if errors occurred.

    Note

    Details shown on the event copy details page are not in real-time. The actual values for details such as **Prefixes copied** may be higher than what is shown on the page. CloudTrail updates the details incrementally over the course of the event copy.
11. If the **Copy status** is **Failed**, fix any errors shown in **Copy failures**, and then choose **Retry copy**. When you retry a copy, CloudTrail resumes the copy at the location where the failure occurred.

For more information about viewing the details of a trail event copy, see View event copy details with the CloudTrail console.

Copy trail events to a new event data store with the console

This walkthrough shows you how to copy trail events to a new CloudTrail Lake event data store for historical analysis. For more information about copying trail events, see Copy trail events to an event data store.

To copy trail events to a new event data store

1. Sign in to the AWS Management Console and open the CloudTrail console at https://console.aws.amazon.com/cloudtrail/.
2. From the navigation pane, under **Lake**, choose **Event data stores**.
3. Choose **Create event data store**.
4. On the **Configure event data store** page, in **General details**, give your event data store a name, such as *my-management-events-eds*. As a best practice, use a name that quickly identifies the purpose of the event data store. For information about CloudTrail naming requirements, see Naming requirements for CloudTrail resources, S3 buckets, and KMS keys.
5. Choose the **Pricing option** that you want to use for your event data store. The pricing option determines the cost for ingesting and storing events, and the default and maximum retention periods for your event data store. For more information, see AWS CloudTrail Pricing and Managing CloudTrail Lake costs.

   The following are the available options:

   - **One-year extendable retention pricing** - Generally recommended if you expect to ingest less than 25 TB of event data per month and want a flexible retention period of up to 10 years. For the first 366 days (the default retention period), storage is included at no additional charge with ingestion pricing. After 366 days, extended retention is available at pay-as-you-go pricing. This is the default option.
     - **Default retention period:** 366 days
     - **Maximum retention period:** 3,653 days
   - **Seven-year retention pricing** - Recommended if you expect to ingest more than 25 TB of event data per month and need a retention period of up to 7 years. Retention is included with ingestion pricing at no additional charge.
     - **Default retention period:** 2,557 days
     - **Maximum retention period:** 2,557 days

6. Specify a retention period for the event data store. Retention periods can be between 7 days and 3,653 days (about 10 years) for the **One-year extendable retention pricing** option, or between 7 days and 2,557 days (about seven years) for the **Seven-year retention pricing** option.

   CloudTrail Lake determines whether to retain an event by checking if the `eventTime` of the event is within the specified retention period. For example, if you specify a retention period of 90 days, CloudTrail will remove events when their `eventTime` is older than 90 days.

   > **Note**
   >
   > CloudTrail will not copy an event if its `eventTime` is older than the specified retention period.
   > To determine the appropriate retention period, take the sum of the oldest event you want to copy in days and the number of days you want to retain the events in the event data store (**retention period** = *`oldest-event-in-days`* + *`number-days-to-retain`*). For example, if the oldest event you're copying is 45 days old and you want to keep the events in the event data store for a further 45 days, you would set the retention period to 90 days.

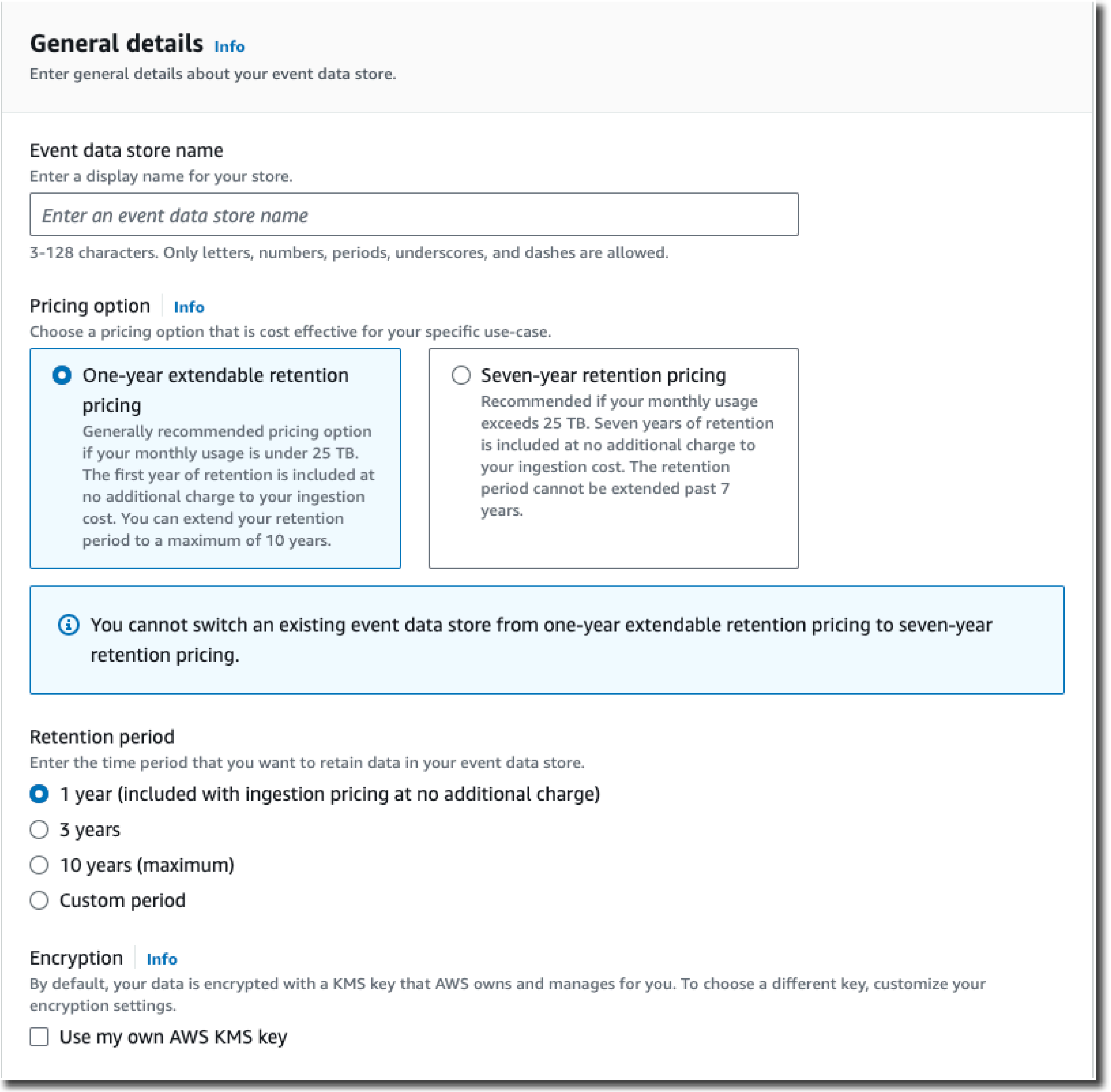
7. (Optional) In **Encryption**. choose whether you want to encrypt the event data store using your own KMS key. By default, all events in an event data store are encrypted by CloudTrail using a KMS key that AWS owns and manages for you.

   To enable encryption using your own KMS key, choose **Use my own AWS KMS key**. Choose **New** to have an AWS KMS key created for you, or choose **Existing** to use an existing KMS key. In **Enter KMS alias**, specify an alias, in the format `alias/`*`MyAliasName`*. Using your own KMS key requires that you edit your KMS key policy to allow CloudTrail logs to be encrypted and decrypted. For more information, see Configure AWS KMS key policies for CloudTrail. CloudTrail also supports AWS KMS multi-Region keys. For more information about multi-Region keys, see Using multi-Region keys in the *AWS Key Management Service Developer Guide*.

   Using your own KMS key incurs AWS KMS costs for encryption and decryption. After you associate an event data store with a KMS key, the KMS key cannot be removed or changed.

Note

To enable AWS Key Management Service encryption for an organization event data store, you must use an existing KMS key for the management account.



General details Info

Enter general details about your event data store.

Event data store name

Enter a display name for your store.

Enter an event data store name

3-128 characters. Only letters, numbers, periods, underscores, and dashes are allowed.

Pricing option | Info

Choose a pricing option that is cost effective for your specific use-case.

● One-year extendable retention pricing

Generally recommended pricing option if your monthly usage is under 25 TB. The first year of retention is included at no additional charge to your ingestion cost. You can extend your retention period to a maximum of 10 years.

○ Seven-year retention pricing

Recommended if your monthly usage exceeds 25 TB. Seven years of retention is included at no additional charge to your ingestion cost. The retention period cannot be extended past 7 years.

You cannot switch an existing event data store from one-year extendable retention pricing to seven-year retention pricing.

Retention period

Enter the time period that you want to retain data in your event data store.

● 1 year (included with ingestion pricing at no additional charge)

○ 3 years

○ 10 years (maximum)

○ Custom period

Encryption | Info

By default, your data is encrypted with a KMS key that AWS owns and manages for you. To choose a different key, customize your encryption settings.

□ Use my own AWS KMS key

8. (Optional) If you want to query against your event data using Amazon Athena, choose **Enable** in **Lake query federation**. Federation lets you view the metadata associated with the event data store in the AWS Glue Data Catalog and run SQL queries against the event data in Athena. The table metadata stored in the AWS Glue Data Catalog lets the Athena query engine know how to find, read, and process the data that you want to query. For more information, see Federate an event data store.

   To enable Lake query federation, choose **Enable** and then do the following:

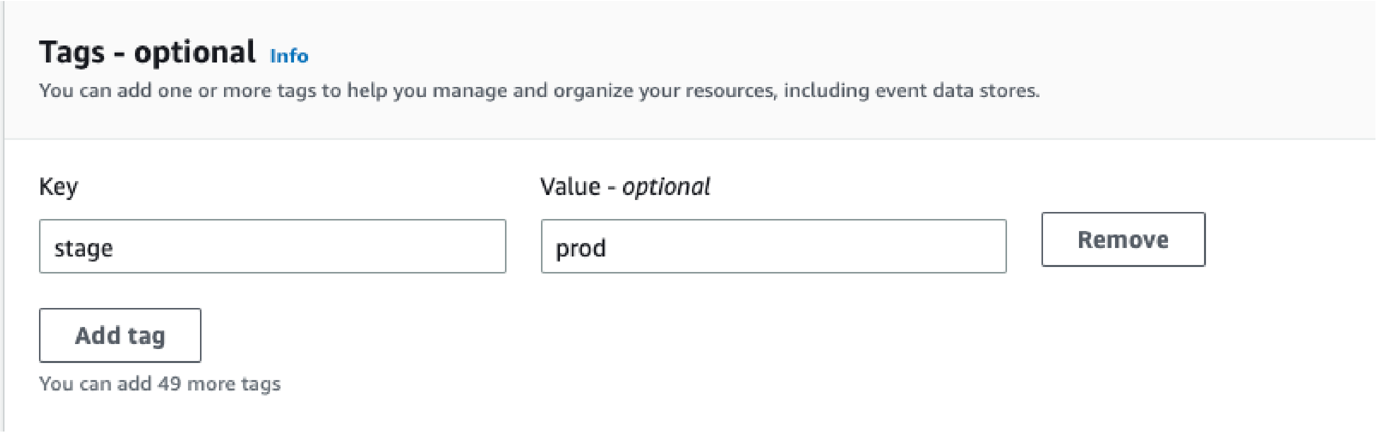
   a. Choose whether you want to create a new role or use an existing IAM role. AWS Lake Formation uses this role to manage permissions for the federated event data store. When you create a new role using the CloudTrail console, CloudTrail automatically creates a role with the required permissions. If you choose an existing role, be sure the policy for the role provides the required minimum permissions.

   b. If you are creating a new role, enter a name to identify the role.

   c. If you are using an existing role, choose the role you want to use. The role must exist in your account.

9. (Optional) In **Tags**, add one or more custom tags (key-value pairs) to your event data store. Tags can help you identify your CloudTrail event data stores. For example, you could attach a tag with the name **`stage`** and the value **`prod`**. You can use tags to limit access to your event data store. You can also use tags to track the query and ingestion costs for your event data store.

   For information about how to use tags to track costs, see Creating user-defined cost allocation tags for CloudTrail Lake event data stores. For information about how to use IAM policies to authorize access to an event data store based on tags, see Examples: Denying access to create or delete event data stores based on tags. For information about how you can use tags in AWS, see Tagging your AWS resources in the *Tagging AWS Resources User Guide*.



Tags - optional Info

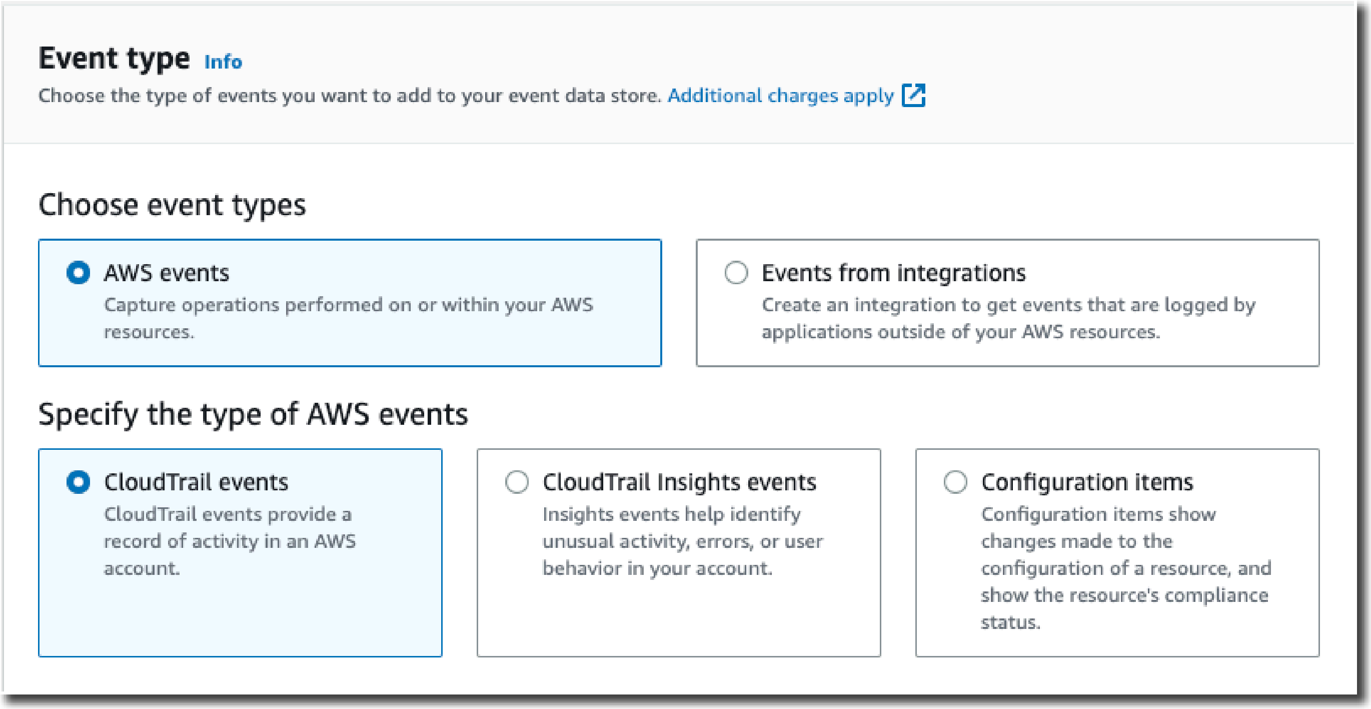
You can add one or more tags to help you manage and organize your resources, including event data stores.

Key: stage

Value - optional: prod

You can add 49 more tags

10. Choose **Next** to configure the event data store.
11. On the **Choose events** page, leave the default selections for **Event type**.



Event type Info

Choose the type of events you want to add to your event data store. Additional charges apply

Choose event types

● AWS events
Capture operations performed on or within your AWS resources.

○ Events from integrations
Create an integration to get events that are logged by applications outside of your AWS resources.

Specify the type of AWS events

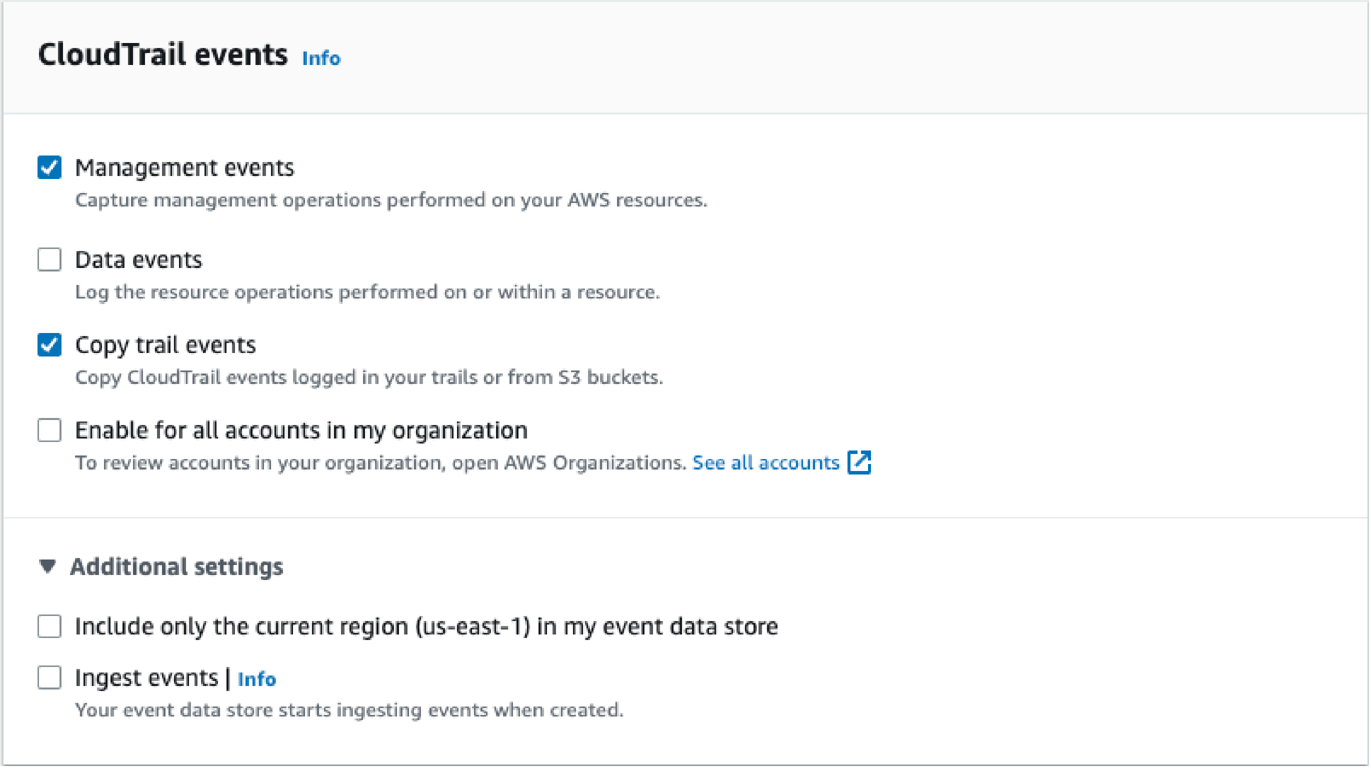
● CloudTrail events
CloudTrail events provide a record of activity in an AWS account.

○ CloudTrail Insights events
Insights events help identify unusual activity, errors, or user behavior in your account.

○ Configuration items
Configuration items show changes made to the configuration of a resource, and show the resource's compliance status.

12. For **CloudTrail events**, we'll leave **Management events** selected and choose **Copy trail events**. In this example, we're not concerned about the event types because we are only using the event data store to analyze past events and are not ingesting future events.

    If you're creating an event data store to replace an existing trail, choose the same event selectors as your trail to ensure the event data store has the same event coverage.



CloudTrail events Info

☑ Management events
Capture management operations performed on your AWS resources.

□ Data events
Log the resource operations performed on or within a resource.

☑ Copy trail events
Copy CloudTrail events logged in your trails or from S3 buckets.

□ Enable for all accounts in my organization
To review accounts in your organization, open AWS Organizations. See all accounts

▼ Additional settings

□ Include only the current region (us-east-1) in my event data store

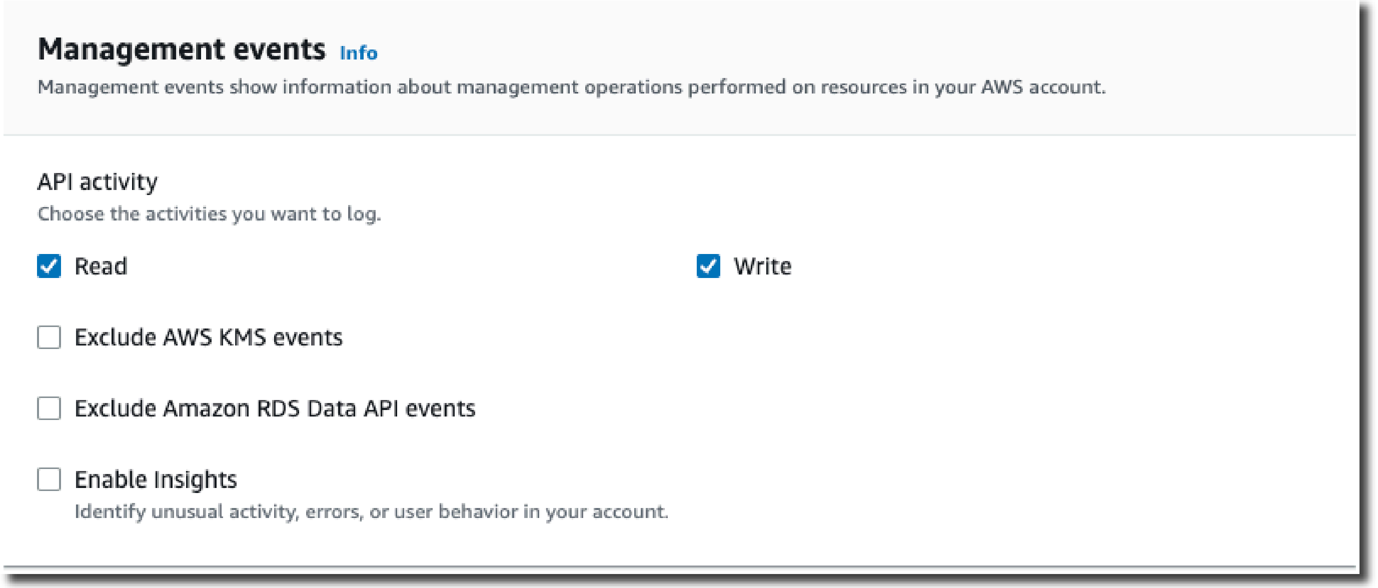
□ Ingest events | Info
Your event data store starts ingesting events when created.

13. Choose **Enable for all accounts in my organization** if this is an organization event data store. This option won't be available to change unless you have accounts configured in AWS Organizations.

> **Note**
>
> If you are creating an organization event data store, you must be signed in with the management account for the organization because only the management account can copy trail events to an organization event data store.

14. For **Additional settings**, we'll deselect **Ingest events**, because in this example we don't want the event data store to ingest any future events as we're only interested in querying the copied events. By default, an event data store collects events for all AWS Regions and starts ingesting events when it's created.

15. For **Management events**, we'll leave the default settings.



Management events Info

Management events show information about management operations performed on resources in your AWS account.

API activity

Choose the activities you want to log.

☑ Read ☑ Write

□ Exclude AWS KMS events

□ Exclude Amazon RDS Data API events

□ Enable Insights

Identify unusual activity, errors, or user behavior in your account.

16. In the **Copy trail events** area, complete the following steps.

    a. Choose the trail that you want to copy. In this example, we'll choose a trail named *management-events*.

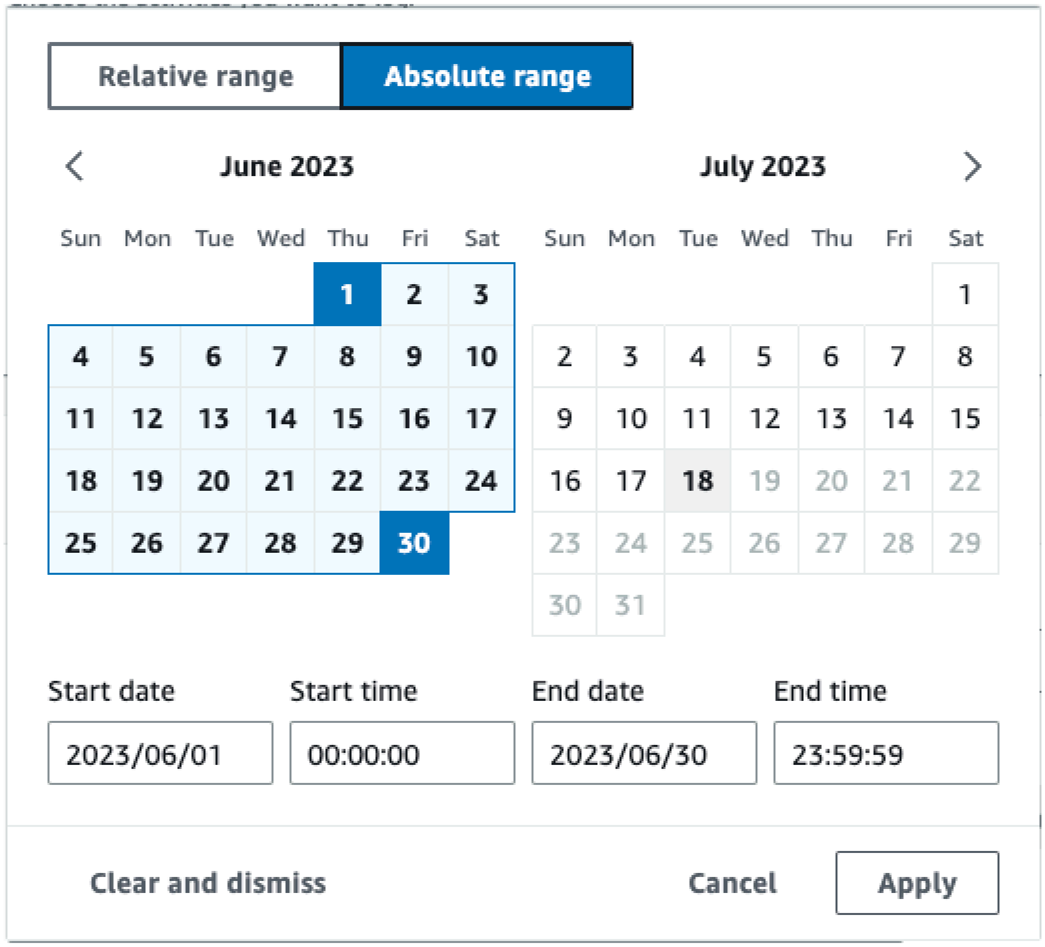
       By default, CloudTrail only copies CloudTrail events contained in the S3 bucket's `CloudTrail` prefix and the prefixes inside the `CloudTrail` prefix, and does not check prefixes for other AWS services. If you want to copy CloudTrail events contained in another prefix, choose **Enter S3 URI**, and then choose **Browse S3** to browse to the prefix. If the source S3 bucket for the trail uses a KMS key for data encryption, ensure that the KMS key policy allows CloudTrail to decrypt the data. If your source S3 bucket uses multiple KMS keys, you must update each key's policy to allow CloudTrail to decrypt the data in the bucket. For more information about updating the KMS key policy, see KMS key policy for decrypting data in the source S3 bucket.

    b. Choose a time range for copying the events. CloudTrail checks the prefix and log file name to verify the name contains a date between the chosen start and end date before attempting to copy trail events. You can choose a **Relative range** or an **Absolute range**. To avoid duplicating events between the source trail and destination event data store, choose a time range that is earlier than the creation of the event data store.

       - If you choose **Relative range**, you can choose to copy events logged in the last 6 months, 1 year, 2 years, 7 years, or a custom range. CloudTrail copies the events logged within the chosen time period.

- If you choose **Absolute range**, you can choose a specific start and end date. CloudTrail copies the events that occurred between the chosen start and end dates.

In this example, we'll choose **Absolute range** and we'll select the entire month of June.



June 2023

Sun	Mon	Tue	Wed	Thu	Fri	Sat
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	

July 2023

Sun	Mon	Tue	Wed	Thu	Fri	Sat
						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31					

Start date: 2023/06/01

Start time: 00:00:00

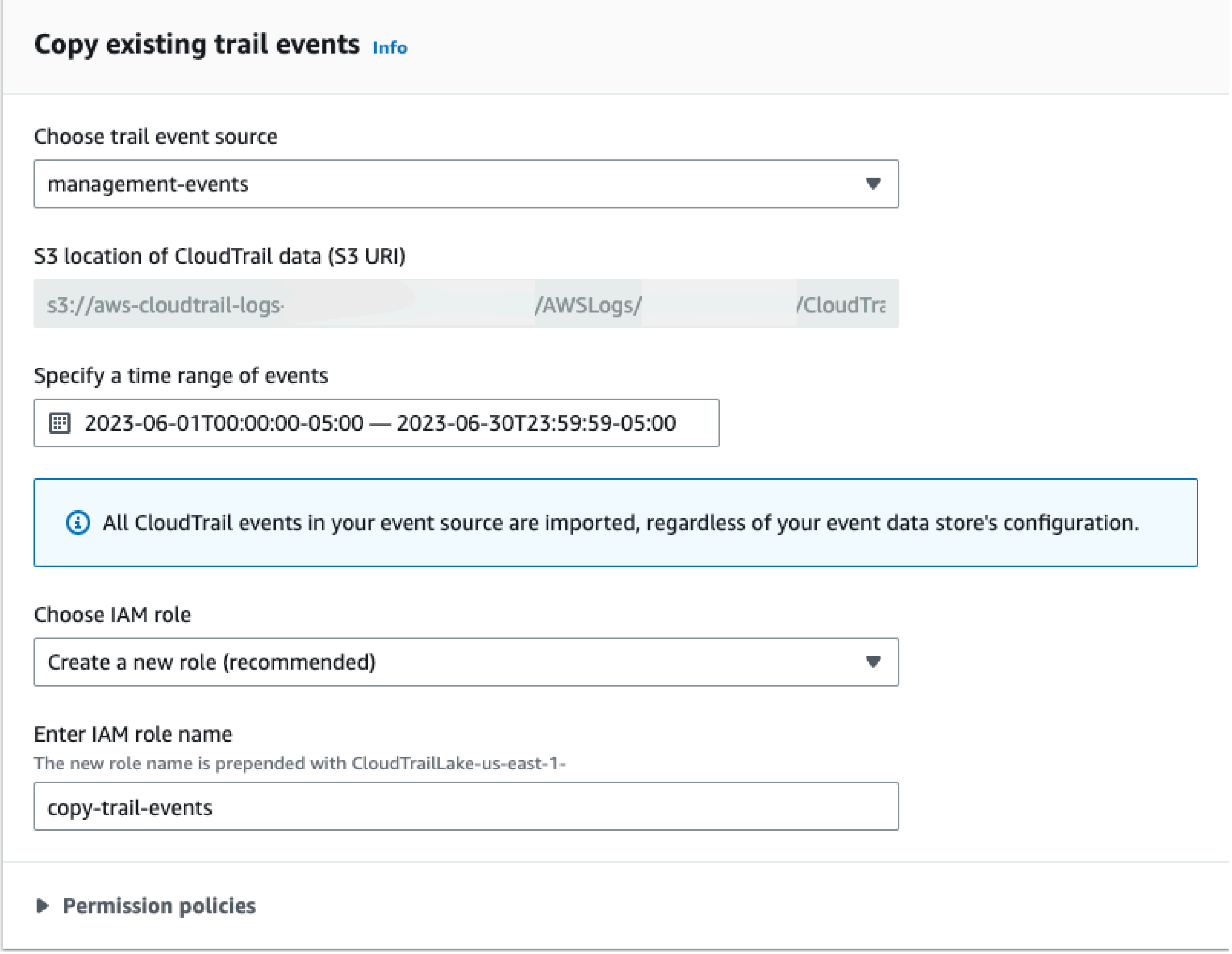
End date: 2023/06/30

End time: 23:59:59

c. For **Permissions**, choose from the following IAM role options. If you choose an existing IAM role, verify that the IAM role policy provides the necessary permissions. For more information about updating the IAM role permissions, see IAM permissions for copying trail events.

- Choose **Create a new role (recommended)** to create a new IAM role. For **Enter IAM role name**, enter a name for the role. CloudTrail automatically creates the necessary permissions for this new role.
- Choose **Use a custom IAM role ARN** to use a custom IAM role that is not listed. For **Enter IAM role ARN**, enter the IAM ARN.
- Choose an existing IAM role from the drop-down list.

In this example, we'll choose **Create a new role (recommended)** and will provide the name **`copy-trail-events`**.



Copy existing trail events

Choose trail event source

management-events ▼

S3 location of CloudTrail data (S3 URI)

s3://aws-cloudtrail-logs-[redacted]/AWSLogs/[redacted]/CloudTra

Specify a time range of events

2023-06-01T00:00:00-05:00 — 2023-06-30T23:59:59-05:00

ⓘ All CloudTrail events in your event source are imported, regardless of your event data store's configuration.

Choose IAM role

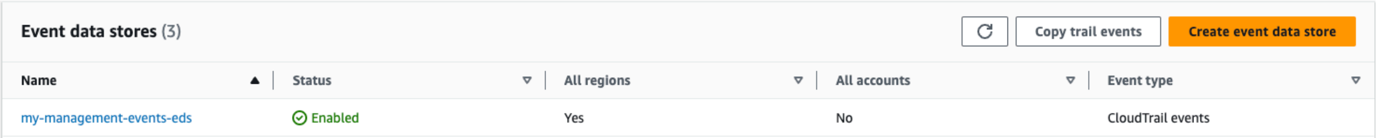
Create a new role (recommended) ▼

Enter IAM role name

The new role name is prepended with CloudTrailLake-us-east-1-

copy-trail-events

17. Choose **Next** to review your choices.
18. On the **Review and create** page, review your choices. Choose **Edit** to make changes to a section. When you're ready to create the event data store, choose **Create event data store**.
19. The new event data store is visible in the **Event data stores** table on the **Event data stores** page.

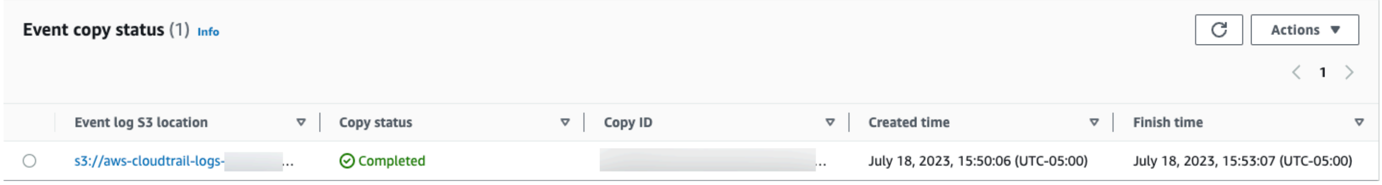


Event data stores (3)

Name	Status	All regions	All accounts	Event type
my-management-events-eds	✓ Enabled	Yes	No	CloudTrail events

20. Choose the event data store name to view its details page. The details page shows the details for your event data store and the status of the copy. The event copy status is shown in the **Event copy status** area.

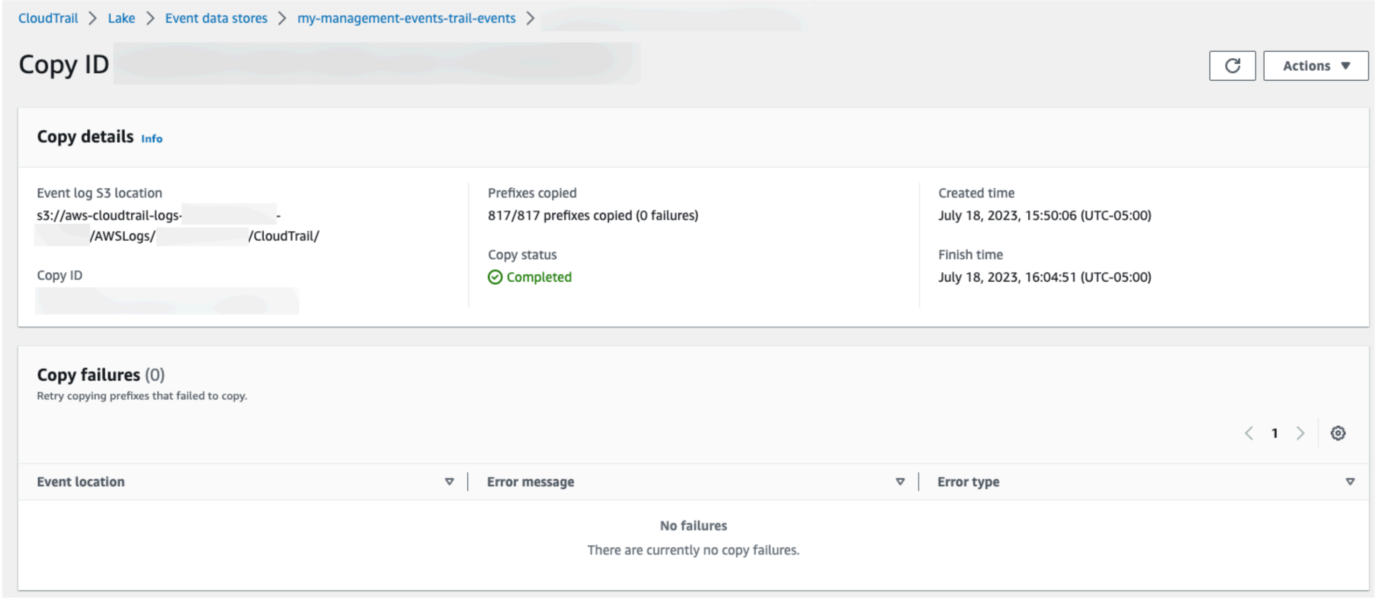
    When a trail event copy completes, its **Copy status** is set to either **Completed** if there were no errors, or **Failed** if errors occurred.



Event copy status (1) Info

	Event log S3 location	Copy status	Copy ID	Created time	Finish time
○	s3://aws-cloudtrail-logs-[redacted]...	✓ Completed	[redacted]...	July 18, 2023, 15:50:06 (UTC-05:00)	July 18, 2023, 15:53:07 (UTC-05:00)

21. To view more details about the copy, choose the copy name in the **Event log S3 location** column, or choose the **View details** option from the **Actions** menu. For more information about viewing the details of a trail event copy, see [View event copy details with the CloudTrail console](#).



CloudTrail > Lake > Event data stores > my-management-events-trail-events >
Actions ▼
1

Copy ID [redacted]

Copy details Info

Event log S3 location
s3://aws-cloudtrail-logs-[redacted]-[redacted]/AWSLogs/[redacted]/CloudTrail/

Copy ID
[redacted]

Prefixes copied
817/817 prefixes copied (0 failures)

Copy status
✓ Completed

Created time
July 18, 2023, 15:50:06 (UTC-05:00)

Finish time
July 18, 2023, 16:04:51 (UTC-05:00)

Copy failures (0)
Retry copying prefixes that failed to copy.

Event location	Error message	Error type

No failures
There are currently no copy failures.

22. The **Copy failures** area shows any errors that occurred when copying trail events. If the **Copy status** is **Failed**, fix any errors shown in **Copy failures**, and then choose **Retry copy**. When you retry a copy, CloudTrail resumes the copy at the location where the failure occurred.

View event copy details with the CloudTrail console

After a trail event copy starts, you can view the event copy details, including the status of the copy, and information on any copy failures.

> **Note**
>
> Details shown on the event copy details page are not in real-time. The actual values for details such as **Prefixes copied** may be higher than what is shown on the page. CloudTrail updates the details incrementally over the course of the event copy.

To access the event copy details page

1. Sign in to the AWS Management Console and open the CloudTrail console at https://console.aws.amazon.com/cloudtrail/.
2. From the left navigation pane, under **Lake**, choose **Event data stores**.
3. Choose the event data store.
4. Choose the event copy in the **Event copy status** section.

Copy details

From **Copy details**, you can view the following details about the trail event copy.

- **Event log S3 location** - The location of the source S3 bucket containing the trail event log files.
- **Copy ID** - The ID for the copy.
- **Prefixes copied** - Represents the number of S3 prefixes copied. During a trail event copy, CloudTrail copies the events in the trail log files that are stored in the prefixes.
- **Copy status** - The status of the copy.
  - **Initializing** - Initial status shown when the trail event copy starts.
  - **In progress** - Indicates the trail event copy is in progress.

    > **Note**
    >
    > You cannot copy trail events if another trail event copy is **In progress**. To stop a trail event copy, choose **Stop copy**.

  - **Stopped** - Indicates a **Stop copy** action occurred. To retry a trail event copy, choose **Retry copy**.

  - **Failed** - The copy completed, but some trail events failed to copy. Review the error messages in **Copy failures**. To retry a trail event copy, choose **Retry copy**. When you retry a copy, CloudTrail resumes the copy at the location where the failure occurred.
  - **Completed** - The copy completed without errors. You can query the copied trail events in the event data store.
- **Created time** - Indicates when the trail event copy started.
- **Finish time** - Indicates when the trail event copy completed or stopped.

Copy failures

From **Copy failures**, you can review the error location, error message, and error type for each copy failure. Common reasons for failure, include if an S3 prefix contained an uncompressed file, or contained a file delivered by a service other than CloudTrail. Another possible cause of failure relates to access issues. For example, if the event data store's S3 bucket did not grant CloudTrail access to import the events, you would get an `AccessDenied` error.

For each copy failure, review the following error information.

- The **Error location** - Indicates the location in the S3 bucket where the error occurred. If an error occurred because the source S3 bucket contained an uncompressed file, the **Error location** would include the prefix where you would find that file.
- The **Error message** - Provides an explanation for why the error occurred.
- The **Error type** - Provides the error type. For example, an **Error type** of `AccessDenied`, indicates that the error occurred because of a permissions issue. For more information about the required permissions for copying trail events, see Required permissions for copying trail events.

After resolving any failures, choose **Retry copy**. When you retry a copy, CloudTrail resumes the copy at the location where the failure occurred.

Federate an event data store

Federating an event data store lets you view the metadata associated with the event data store in the AWS Glue Data Catalog, registers the Data Catalog with AWS Lake Formation, and lets you run SQL queries against your event data using Amazon Athena. The table metadata stored in the AWS Glue Data Catalog lets the Athena query engine know how to find, read, and process the data that you want to query.

You can enable federation by using the CloudTrail console, AWS CLI, or EnableFederation API operation. When you enable Lake query federation, CloudTrail creates a managed database named `aws:cloudtrail` (if the database doesn't already exist) and a managed federated table in the AWS Glue Data Catalog. The event data store ID is used for the table name. CloudTrail registers the federation role ARN and event data store in AWS Lake Formation, the service responsible for allowing fine-grained access control of the federated resources in the AWS Glue Data Catalog.

To enable Lake query federation, you must create a new IAM role or choose an existing role. Lake Formation uses this role to manage permissions for the federated event data store. When you create a new role using the CloudTrail console, CloudTrail automatically creates the required permissions for the role. If you choose an existing role, be sure that the role provides the minimum permissions.

You can disable federation by using the CloudTrail console, AWS CLI, or DisableFederation API operation. When you disable federation, CloudTrail disables the integration with AWS Glue, AWS Lake Formation, and Amazon Athena. After disabling Lake query federation, you can no longer query your event data in Athena. No CloudTrail Lake data is deleted when you disable federation and you can continue to run queries in CloudTrail Lake.

There are no CloudTrail charges for federating a CloudTrail Lake event data store. There are costs for running queries in Amazon Athena. For more information about Athena pricing, see Amazon Athena Pricing.

Analyze Activity Logs with AWS CloudTrail Lake and Amazon Athena

Topics

Considerations

Consider the following factors when federating an event data store:

- There are no CloudTrail charges for federating a CloudTrail Lake event data store. There are costs for running queries in Amazon Athena. For more information about Athena pricing, see Amazon Athena Pricing.
- Lake Formation is used to manage permissions for the federated resources. If you delete the federation role, or revoke permissions to the resources from Lake Formation or AWS Glue, you can't run queries from Athena. For more information about working with Lake Formation, see Managing CloudTrail Lake federation resources with AWS Lake Formation.
- Anyone using Amazon Athena to query data registered with Lake Formation must have an IAM permissions policy that allows the `lakeformation:GetDataAccess` action. The AWS managed policy: AmazonAthenaFullAccess allows this action. If you use inline policies, be sure to update permissions policies to allow this action. For more information, see Managing Lake Formation and Athena user permissions.
- To create views on federated tables in Athena, you need a destination database other than `aws:cloudtrail`. This is because the `aws:cloudtrail` database is managed by CloudTrail.
- To create a dataset in Amazon QuickSight, you must choose the **Use custom SQL** option. For more information, see Creating a dataset using Amazon Athena data.
- If federation is enabled, you can't delete an event data store. To delete a federated event data store, you must first disable federation and termination protection if it's enabled.
- The following considerations apply to organization event data stores:
  - Only a single delegated administrator account or the management account can enable federation on an organization event data store. Other delegated administrator accounts can still query and share information using the Lake Formation data sharing feature.
  - Any delegated administrator account or the organization's management account can disable federation.

Required permissions for federation

Before federating an event data store, be sure that you have all the required permissions for the federation role and for enabling and disabling federation. You only need to update the federation role permissions if you choose an existing IAM role to enable federation. If you choose to create a new IAM role using the CloudTrail console, CloudTrail provides all necessary permissions for the role.

Topics

- Required permissions for enabling federation
- Required permissions for disabling federation

IAM permissions for federating an event data store

When you enable federation, you have the option to create a new IAM role, or use an existing IAM role. When you choose a new IAM role, CloudTrail creates an IAM role with the required permissions and no further action is required on your part.

If you choose an existing role, ensure the IAM role's policies provide the required permissions to enable federation. This section provides examples of the required IAM role permission and trust policies.

The following example provides the permissions policy for the federation role. For the first statement provide the full ARN of your event data store for the `Resource`.

The second statement in this policy allows Lake Formation to decrypt data for an event data store encrypted with a KMS key. Replace *`key-region`*, *`account-id`*, and *`key-id`* with the values for your KMS key. You can omit this statement if your event data store does not use a KMS key for encryption.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "LakeFederationEDSDataAccess",
            "Effect": "Allow",
            "Action": "cloudtrail:GetEventDataStoreData",
            "Resource": "arn:aws:cloudtrail:eds-region:account-id:eventdatastore/eds-id"
        },
        {
          "Sid": "LakeFederationKMSDecryptAccess",
          "Effect": "Allow",
          "Action": [
              "kms:Decrypt",
              "kms:GenerateDataKey"
          ],
          "Resource": "arn:aws:kms:key-region:account-id:key/key-id"
      }
    ]
}
```

The following example provides the IAM trust policy, which allows AWS Lake Formation to assume an IAM role to manage permissions for the federated event data store.

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

Required permissions for enabling federation

The following example policy provides the minimum required permissions to enable federation on an event data store. This policy allows CloudTrail to enable federation on the event data store, AWS Glue to create the federated resources in the AWS Glue Data Catalog, and AWS Lake Formation to manage resource registration.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "CloudTrailEnableFederation",
            "Effect": "Allow",
            "Action": "cloudtrail:EnableFederation",
            "Resource": "arn:aws:cloudtrail:region:account-id:eventdatastore/eds-id"
        },
        {
            "Sid": "FederationRoleAccess",
            "Effect": "Allow",
            "Action": [
                "iam:PassRole",
                "iam:GetRole"
            ],
            "Resource": "arn:aws:iam::region:role/federation-role-name"
        },
        {
            "Sid": "GlueResourceCreation",
            "Effect": "Allow",
            "Action": [
                "glue:CreateDatabase",
                "glue:CreateTable",
                "glue:PassConnection"
            ],
            "Resource": [
                "arn:aws:glue:region:account-id:catalog",
                "arn:aws:glue:region:account-id:database/aws:cloudtrail",
                "arn:aws:glue:region:account-id:table/aws:cloudtrail/eds-id",
                "arn:aws:glue:region:account-id:connection/aws:cloudtrail"
            ]
        },
        {
            "Sid": "LakeFormationRegistration",
            "Effect": "Allow",
            "Action": [
                "lakeformation:RegisterResource",
                "lakeformation:DeregisterResource"
            ],
            "Resource": "arn:aws:lakeformation:region:account-id:catalog:account-id"
        }
    ]
}
```

Required permissions for disabling federation

The following example policy provides the minimum required resources to disable federation on an event data store. This policy allows CloudTrail to disable federation on the event data store, AWS Glue to delete the managed federated table in the AWS Glue Data Catalog, and Lake Formation to deregister the federated resource.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "CloudTrailDisableFederation",
            "Effect": "Allow",
            "Action": "cloudtrail:DisableFederation",
            "Resource": "arn:aws:cloudtrail:region:account-id:eventdatastore/eds-id"
        },
        {
            "Sid": "GlueTableDeletion",
            "Effect": "Allow",
            "Action": "glue:DeleteTable",
            "Resource": [
                "arn:aws:glue:region:account-id:catalog",
                "arn:aws:glue:region:account-id:database/aws:cloudtrail",
                "arn:aws:glue:region:account-id:table/aws:cloudtrail/eds-id"
            ]
        },
        {
            "Sid": "LakeFormationDeregistration",
            "Effect": "Allow",
            "Action": "lakeformation:DeregisterResource",
            "Resource": "arn:aws:lakeformation:region:account-id:catalog:account-id"
        }
    ]
}
```

Enable Lake query federation

You can enable Lake query federation by using the CloudTrail console, AWS CLI, or EnableFederation API operation. When you enable Lake query federation, CloudTrail creates a managed database named `aws:cloudtrail` (if the database doesn't already exist) and a managed federated table in the AWS Glue Data Catalog. The event data store ID is used for the table name. CloudTrail registers the federation role ARN and event data store in AWS Lake Formation, the service responsible for allowing fine-grained access control of the federated resources in the AWS Glue Data Catalog.

This section describes how to enable federation using the CloudTrail console and AWS CLI.

CloudTrail console

The following procedure shows you how to enable Lake query federation on an existing event data store.

1. Sign in to the AWS Management Console and open the CloudTrail console at https://console.aws.amazon.com/cloudtrail/.
2. In the navigation pane, under **Lake**, choose **Event data stores**.

3. Choose the event data store that you want to update. This opens the event data store's details page.
4. In **Lake query federation**, choose **Edit** and then choose **Enable**.
5. Choose whether to create a new IAM role, or use an existing role. When you create a new role, CloudTrail automatically creates a role with the required permissions. If you're using an existing role, be sure the role's policy provides the required minimum permissions.
6. If you're creating a new IAM role, enter a name for the role.
7. If you're choosing an existing IAM role, choose the role you want to use. The role must exist in your account.
8. Choose **Save changes**. The **Federation status** changes to `Enabled`.

AWS CLI

To enable federation, run the **aws cloudtrail enable-federation** command, providing the required **--event-data-store** and **--role** parameters. For **--event-data-store**, provide the event data store ARN (or the ID suffix of the ARN). For **--role**, provide the ARN for your federation role. The role must exist in your account and provide the required minimum permissions.

```
aws cloudtrail enable-federation
--event-data-store arn:aws:cloudtrail:region:account-id:eventdatastore/eds-id
--role arn:aws:iam::account-id:role/federation-role-name
```

This example shows how a delegated administrator can enable federation on an organization event data store by specifying the ARN of the event data store in the management account and the ARN of the federation role in the delegated administrator account.

```
aws cloudtrail enable-federation
--event-data-store arn:aws:cloudtrail:region:management-account-
id:eventdatastore/eds-id
--role arn:aws:iam::delegated-administrator-account-id:role/federation-role-name
```

Disable Lake query federation

You can disable federation by using the CloudTrail console, AWS CLI, or DisableFederation API operation. When you disable federation, CloudTrail disables the integration with AWS Glue, AWS Lake Formation, and Amazon Athena. After disabling Lake query federation, you can no longer query your event data in Athena. No CloudTrail Lake data is deleted when you disable federation and you can continue to run queries in CloudTrail Lake.

This section describes how to disable federation using the CloudTrail console and AWS CLI.

CloudTrail console

The following procedure shows you how to disable Lake query federation on an existing event data store.

1. Sign in to the AWS Management Console and open the CloudTrail console at https://console.aws.amazon.com/cloudtrail/.
2. In the navigation pane, under **Lake**, choose **Event data stores**.
3. Choose the event data store that you want to update. This opens the event data store's details page.
4. In **Lake query federation**, choose **Edit** and then choose **Disable**.
5. Choose **Save changes**. The **Federation status** changes to `Disabled`.

AWS CLI

To disable federation on the event data store, run the **aws cloudtrail disable-federation** command. The event data store is specified by `--event-data-store`, which accepts an event data store ARN or the ID suffix of the ARN.

```
aws cloudtrail disable-federation
--event-data-store arn:aws:cloudtrail:region:account-id:eventdatastore/eds-id
```

> **Note**
>
> If this is an organization event data store, use the account ID for the management account.

Managing CloudTrail Lake federation resources with AWS Lake Formation

When you federate an event data store, CloudTrail registers the federation role ARN and event data store in AWS Lake Formation, the service responsible for allowing fine-grained access control of the federated resources in the AWS Glue Data Catalog. This section describes how you can use Lake Formation to manage the CloudTrail Lake federation resources.

When you enable federation, CloudTrail creates the following resources in the AWS Glue Data Catalog.

- **Managed database** – CloudTrail creates 1 database with the name `aws:cloudtrail` per account. CloudTrail manages the database. You can't delete or modify the database in AWS Glue.
- **Managed federated table** – CloudTrail creates 1 table for each federated event data store and uses the event data store ID for the table name. CloudTrail manages the tables. You can't delete or modify the tables in AWS Glue. To delete a table, you must disable federation on the event data store.

Controlling access to federated resources

You can use one of two permissions methods to control access to the managed database and tables.

- **IAM only access control** – With IAM only access control, all users in the account with the required IAM permissions are given access to all Data Catalog resources. For information about how AWS Glue works with IAM, see How AWS Glue works with IAM.

  On the Lake Formation console, this method appears as **Use only IAM access control**.

  > **Note**
  >
  > If you want to create data filters and use other Lake Formation features, you must use Lake Formation access control.

- **Lake Formation access control** – This methods provides the following advantages.
  - You can implement column-level, row-level, and cell-level security by creating data filters. For more information, see Securing data lakes with row-level access control in the *AWS Lake Formation Developer Guide*.
  - Database and tables are only visible to Lake Formation administrators and creators of the database and resources. If another user needs access to these resources, you must explicitly grant access by using Lake Formation permissions.

For more information about access control, see Methods for fine-grained access control.

Determining the permissions method for a federated resource

When you enable federation for the first time, CloudTrail creates a managed database and managed federated table using your Lake Formation data lake settings.

After CloudTrail enables federation, you can verify which permissions method you are using for the managed database and managed federated table by checking the permissions for those resources. If the `ALL` (*Super*) to `IAM_ALLOWED_PRINCIPALS` setting is present for the resource, the resource is managed exclusively by IAM permissions. If the setting is missing, the resource is managed by Lake Formation permissions. For more information about Lake Formation permissions, see [Lake Formation permissions reference].

The permissions method for the managed database and managed federated table can differ. For example, if you check the values for the database and table, you could see the following:

- For the database, the value that assigns `ALL` (*Super*) to `IAM_ALLOWED_PRINCIPALS` is present in the permissions indicating that the you're using IAM only access control for the database.
- For the table, the value that assigns `ALL` (*Super*) to `IAM_ALLOWED_PRINCIPALS` not present, which indicates access control by Lake Formation permissions.

You can switch between access methods at any time by adding or removing `ALL` (*Super*) to `IAM_ALLOWED_PRINCIPALS` permission on any federated resource in Lake Formation.

Cross-account sharing using Lake Formation

This section describes how to share a managed database and managed federated table across accounts by using Lake Formation.

You can share a managed database across accounts by taking these steps:

1. Update the cross-account data sharing version to version 4.
2. Remove `Super` to `IAM_ALLOWED_PRINCIPALS` permissions from the database if present to switch to Lake Formation access control.
3. Grant `Describe` permissions to the external account on the database.
4. If a Data Catalog resource is shared with your AWS account and your account is not in the same AWS organization as the sharing account, accept the resource share invitation from AWS Resource Access Manager (AWS RAM). For more information, see Accepting a resource share invitation from AWS RAM.

After completing these steps, the database should be visible to the external account. By default, sharing the database does not give access to any tables in the database.

You can share all or individual managed federated tables with an external account by taking these steps:

1. Update the cross-account data sharing version to version 4.
2. Remove `Super` to `IAM_ALLOWED_PRINCIPALS` permissions from the table if present to switch to Lake Formation access control.
3. (Optional) Specify any data filters to restrict columns or rows.
4. Grant `Select` permissions to the external account on the table.
5. If a Data Catalog resource is shared with your AWS account and your account is not in the same AWS organization as the sharing account, accept the resource share invitation from AWS Resource Access Manager (AWS RAM). For an organization, you can auto accept using RAM settings. For more information, see Accepting a resource share invitation from AWS RAM.
6. The table should now be visible. To enable Amazon Athena queries on this table, create a resource link in this account with the shared table.

The owning account can revoke sharing at any point by removing permissions for the external account from Lake Formation, or by disabling federation in CloudTrail.

Understanding organization event data stores

If you have created an organization in AWS Organizations, you can create an *organization event data store* that logs all events for all AWS accounts in that organization. Organization event data stores can apply to all AWS Regions, or the current Region. You can't use an organization event data store to collect events from outside of AWS.

You can create an organization event data store by using either the management account or the delegated administrator account. When a delegated administrator creates an organization event data store, the organization event data store exists in the management account for the organization. This approach is because the management account maintains ownership of all organization resources.

The management account for an organization can update an account-level event data store to apply it to an organization.

When the organization event data store is specified as applying to an organization, it's automatically applied to all member accounts in the organization. Member accounts can't see the organization event data store, nor can they modify or delete it. By default, member accounts don't have access to the organization event data store, nor can they run queries on organization event data stores.

The following table shows the capabilities of the management account and delegated administrator accounts within the AWS Organizations organization.

Capabilities	**Management account**	**Delegated administrator account**
Register or remove delegated administrator accounts.	Yes	No
Create an organization event data store for AWS CloudTrail events or AWS Config configuration items.	Yes	Yes
Enable Insights on an organization event data store.	Yes	No
Update an organization event data store.	Yes	Yes[1]
Enable Lake query federation on an organization event data store.[2]	Yes	Yes
Disable Lake query federation on an organization event data store.	Yes	Yes
Delete an organization event data store.	Yes	Yes
Copy trail events to an event data store.	Yes	No
Run queries on organization event data stores.	Yes	Yes
View the CloudTrail Lake dashboard for an organization event data store.	Yes	Yes

[1] Only the management account can convert an organization event data store to an account-level event data store, or convert an account-level event data store to an organization event data store. These actions are not allowed for the delegated administrator because organization event data stores only exist in the management account. When an organization event data store is converted to an account-level event data store, only the management account has access to the event data store. Likewise, only an account-level event data store in the management account can be converted to an organization event data store.

[2] Only a single delegated administrator account or the management account can enable federation on an organization event data store. Other delegated administrator accounts can query and share information using the Lake Formation data sharing feature. Any delegated administrator account as well as the organization's management account can disable federation.

Create an organization event data store

The management account or delegated administrator account for an organization can create an organization event data store to collect either CloudTrail events (management events, data events) or AWS Config configuration items.

> **Note**
>
> Only the organization's management account can copy trail events to an event data store.

CloudTrail console

To create an organization event data store using the console

1. Follow the steps in the create an event data store for CloudTrail events procedure to create an organization event data store for CloudTrail management or data events.

   OR

   Follow the steps in the create an event data store for AWS Config configuration items procedure to create an organization event data store for AWS Config configuration items.
2. On the **Choose events** page, choose **Enable for all accounts in my organization**.

AWS CLI

To create an organization event data store run the create-event-data-store command and include the `--organization-enabled` option.

The following example AWS CLI `create-event-data-store` command creates an organization event data store that collects all management events. Because CloudTrail logs management events by default, you don't need to specify advanced event selectors if your event data store is logging all management events and is not collecting any data events.

```
aws cloudtrail create-event-data-store --name org-management-eds --organization-
enabled
```

The following is an example response.

```
{
    "EventDataStoreArn": "arn:aws:cloudtrail:us-east-1:123456789012:eventdatastore/
EXAMPLE6-d493-4914-9182-e52a7934b207",
    "Name": "org-management-eds",
    "Status": "CREATED",
    "AdvancedEventSelectors": [
        {
            "Name": "Default management events",
            "FieldSelectors": [
                {
                    "Field": "eventCategory",
                    "Equals": [
                        "Management"
                    ]
                }
            ]
        }
    ],
    "MultiRegionEnabled": true,
    "OrganizationEnabled": true,
    "BillingMode": "EXTENDABLE_RETENTION_PRICING",
    "RetentionPeriod": 366,
    "TerminationProtectionEnabled": true,
    "CreatedTimestamp": "2023-11-16T15:30:50.689000+00:00",
    "UpdatedTimestamp": "2023-11-16T15:30:50.851000+00:00"
}
```

The next example AWS CLI `create-event-data-store` command creates an organization event data store named `config-items-org-eds` that collects AWS Config configuration items. To collect configuration items, specify that the `eventCategory` field equals `ConfigurationItem` in the advanced event selectors.

```
aws cloudtrail create-event-data-store --name config-items-org-eds \
--organization-enabled \
--advanced-event-selectors '[
    {
        "Name": "Select AWS Config configuration items",
        "FieldSelectors": [
            { "Field": "eventCategory", "Equals": ["ConfigurationItem"] }
        ]
    }
]'
```

Apply an account-level event data store to an organization

The organization's management account can convert an account-level event data store to apply it to an organization.

CloudTrail console

To update an account-level event data store using the console

1. Sign in to the AWS Management Console and open the CloudTrail console at https://console.aws.amazon.com/cloudtrail/.
2. In the navigation pane, under **Lake**, choose **Event data stores**.
3. Choose the event data store that you want to update. This action opens the event data store's details page.
4. In **General details**, choose **Edit**.
5. Choose **Enable for all accounts in my organization**.
6. Choose **Save changes**.

For additional information about updating an event data store, see Update an event data store with the console.

AWS CLI

To update an account-level event data store to apply it to an organization, run the update-event-data-store command and include the `--organization-enabled` option.

```
aws cloudtrail update-event-data-store --region us-east-1 \
--organization-enabled \
--event-data-store arn:aws:cloudtrail:us-east-1:123456789012:eventdatastore/EXAMPLE-
f852-4e8f-8bd1-bcf6cEXAMPLE
```

See also

- Organization delegated administrator
- Add a CloudTrail delegated administrator
- Remove a CloudTrail delegated administrator

Create an integration with an event source outside of AWS

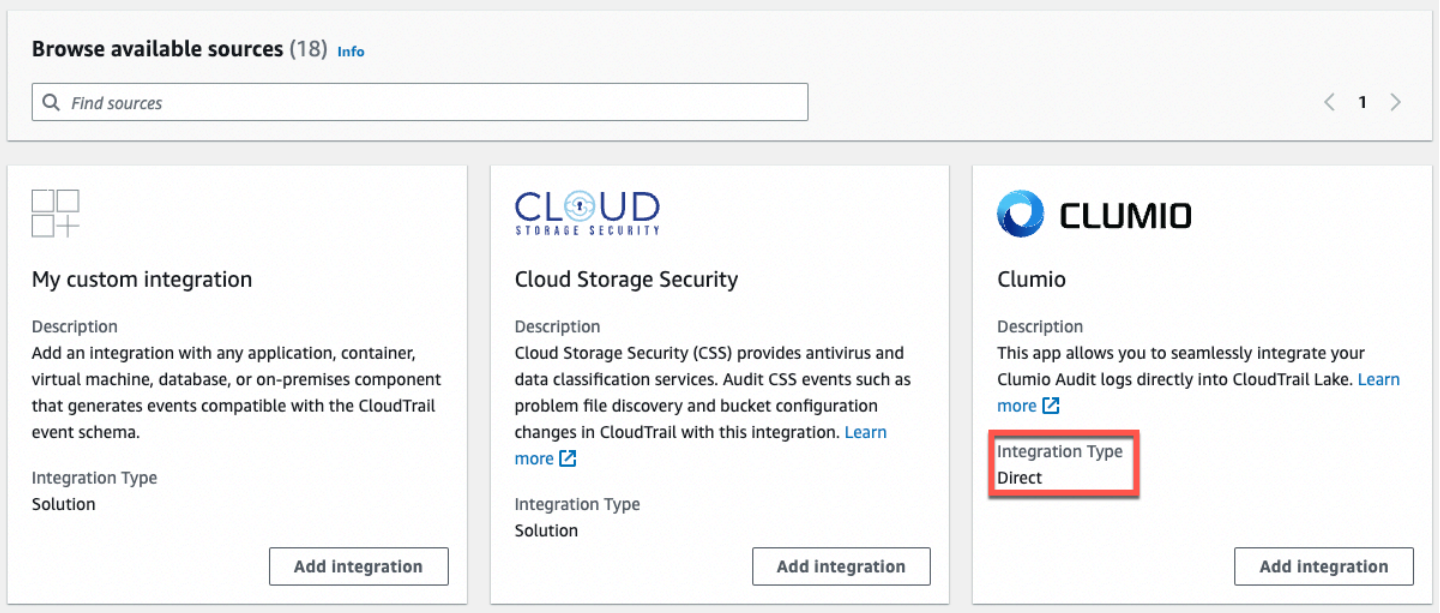
You can use CloudTrail to log and store user activity data from any source in your hybrid environments, such as in-house or SaaS applications hosted on-premises or in the cloud, virtual machines, or containers. You can store, access, analyze, troubleshoot and take action on this data without maintaining multiple log aggregators and reporting tools.

Activity events from non-AWS sources work by using *channels* to bring events into CloudTrail Lake from external partners that work with CloudTrail, or from your own sources. When you create a channel, you choose one or more event data stores to store events that arrive from the channel source. You can change the destination event data stores for a channel as needed, as long as the destination event data stores are set to log `eventCategory="ActivityAuditLog"` events. When you create a channel for events from an external partner, you provide a channel ARN to the partner or source application. The resource policy attached to the channel allows the source to transmit events through the channel. If a channel does not have a resource policy, only the channel owner can call the `PutAuditEvents` API on the channel.

CloudTrail has partnered with many event source providers, such as Okta and LaunchDarkly. When you create an integration with an event source outside AWS, you can choose one of these partners as your event source, or choose **My custom integration** to integrate events from your own sources into CloudTrail. A maximum of one channel is allowed per source.

There are two types of integrations: direct and solution. With direct integrations, the partner calls the `PutAuditEvents` API to deliver events to the event data store for your AWS account. With solution integrations, the application runs in your AWS account and the application calls the `PutAuditEvents` API to deliver events to the event data store for your AWS account.

From the **Integrations** page, you can choose the **Available sources** tab to the view the **Integration type** for partners.



Browse available sources (18) Info

Find sources

My custom integration	Cloud Storage Security	Clumio
Description Add an integration with any application, container, virtual machine, database, or on-premises component that generates events compatible with the CloudTrail event schema.	Description Cloud Storage Security (CSS) provides antivirus and data classification services. Audit CSS events such as problem file discovery and bucket configuration changes in CloudTrail with this integration. Learn more	Description This app allows you to seamlessly integrate your Clumio Audit logs directly into CloudTrail Lake. Learn more
Integration Type Solution	Integration Type Solution	Integration Type Direct
Add integration	Add integration	Add integration

To get started, create an integration to log events from partner or other application sources using the CloudTrail console.

Topics

- Create an integration with a CloudTrail partner with the console
- Create a custom integration with the console
- Create, update, and manage CloudTrail Lake integrations with the AWS CLI
- Additional information about integration partners
- CloudTrail Lake integrations event schema

Create an integration with a CloudTrail partner with the console

When you create an integration with an event source outside AWS, you can choose one of these partners as your event source. When you create an integration in CloudTrail with a partner application, the partner needs the Amazon Resource Name (ARN) of the channel that you create in this workflow to send events to CloudTrail. After you create the integration, you finish configuring the integration by following the partner's instructions to provide the required channel ARN to the partner. The integration starts ingesting partner events into CloudTrail after the partner calls `PutAuditEvents` on the integration's channel.

1. Sign in to the AWS Management Console and open the CloudTrail console at https://console.aws.amazon.com/cloudtrail/.
2. From the navigation pane, under **Lake**, choose **Integrations**.
3. On the **Add integration** page, enter a name for your channel. The name can be 3-128 characters. Only letters, numbers, periods, underscores, and dashes are allowed.
4. Choose the partner application source from which you want to get events. If you're integrating with events from your own applications hosted on-premises or in the cloud, choose **My custom integration**.
5. From **Event delivery location**, choose to log the same activity events to existing event data stores, or create a new event data store.

   If you choose to create a new event data store, enter a name for the event data store, choose the pricing option, and specify the retention period in days. The event data store retains event data for the specified number of days.

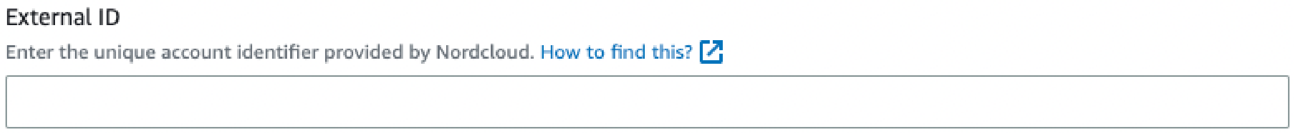
   If you choose to log activity events to one or more existing event data stores, choose the event data stores from the list. The event data stores can only include activity events. The event type in the console must be **Events from integrations**. In the API, the `eventCategory` value must be `ActivityAuditLog`.
6. In **Resource policy**, configure the resource policy for the integration's channel. Resource policies are JSON policy documents that specify what actions a specified principal can perform on the resource and under what conditions. The accounts defined as principals in the resource policy can call the `PutAuditEvents` API to deliver events to your channel. The resource owner has implicit access to the resource if their IAM policy allows the `cloudtrail-data:PutAuditEvents` action.

   The information required for the policy is determined by the integration type. For a direction integration, CloudTrail automatically adds the partner's AWS account IDs, and requires you to enter the unique external ID provided by the partner. For a solution integration, you must specify at least one AWS account ID as principal, and can optionally enter an external ID to prevent against confused deputy.

> **Note**
>
> If you do not create a resource policy for the channel, only the channel owner can call the `PutAuditEvents` API on the channel.

a. For a direct integration, enter the external ID provided by your partner. The integration partner provides a unique external ID, such as an account ID or a randomly generated string, to use for the integration to prevent against confused deputy. The partner is responsible for creating and providing a unique external ID.

   You can choose **How to find this?** to view the partner's documentation that describes how to find the external ID.

   External ID

   Enter the unique account identifier provided by Nordcloud. How to find this?

   > **Note**
   >
   > If the resource policy includes an external ID, all calls to the `PutAuditEvents` API must include the external ID. However, if the policy does not define an external ID, the partner can still call the `PutAuditEvents` API and specify an `externalId` parameter.

b. For a solution integration, choose **Add AWS account** to specify an AWS account ID to add as a principal in the policy.

7. (Optional) In the **Tags** area, you can add up to 50 tag key and value pairs to help you identify, sort, and control access to your event data store and channel. For more information about how to use IAM policies to authorize access to an event data store based on tags, see Examples: Denying access to create or delete event data stores based on tags. For more information about how you can use tags in AWS, see Tagging AWS resources in the *AWS General Reference*.

8. When you are ready to create the new integration, choose **Add integration**. There is no review page. CloudTrail creates the integration, but you must provide the channel Amazon Resource Name (ARN) to the partner application. Instructions for providing the channel ARN to the partner application are found on the partner documentation website. For more information,

choose the **Learn more** link for the partner on the **Available sources** tab of the **Integrations** page to open the partner's page in AWS Marketplace.

To finish the setup for your integration, provide the channel ARN to the partner or source application. Depending upon the integration type, either you, the partner, or the application runs the `PutAuditEvents` API to deliver activity events to the event data store for your AWS account. After your activity events are delivered, you can use CloudTrail Lake to search, query, and analyze the data that is logged from your applications. Your event data includes fields that match CloudTrail event payload, such as `eventVersion`, `eventSource`, and `userIdentity`.

Create a custom integration with the console

You can use CloudTrail to log and store user activity data from any source in your hybrid environments, such as in-house or SaaS applications hosted on-premises or in the cloud, virtual machines, or containers. Perform the first half of this procedure in the CloudTrail Lake console, then call the `PutAuditEvents` API to ingest events, providing your channel ARN and event payload. After you use the `PutAuditEvents` API to ingest your application activity into CloudTrail, you can use CloudTrail Lake to search, query, and analyze the data that is logged from your applications.

1. Sign in to the AWS Management Console and open the CloudTrail console at https://console.aws.amazon.com/cloudtrail/.
2. From the navigation pane, under **Lake**, choose **Integrations**.
3. On the **Add integration** page, enter a name for your channel. The name can be 3-128 characters. Only letters, numbers, periods, underscores, and dashes are allowed.
4. Choose **My custom integration**.
5. From **Event delivery location**, choose to log the same activity events to existing event data stores, or create a new event data store.

   If you choose to create a new event data store, enter a name for the event data store and specify the retention period in days. You can keep the event data in an event data store for up to 3,653 days (about 10 years) if you choose the **One-year extendable retention pricing** option, or up to 2,557 days (about 7 years) if you choose the **Seven-year retention pricing** option.

   If you choose to log activity events to one or more existing event data stores, choose the event data stores from the list. The event data stores can only include activity events. The event type in the console must be **Events from integrations**. In the API, the `eventCategory` value must be `ActivityAuditLog`.

6. In **Resource policy**, configure the resource policy for the integration's channel. Resource policies are JSON policy documents that specify what actions a specified principal can perform on the resource and under what conditions. The accounts defined as principals in the resource policy can call the `PutAuditEvents` API to deliver events to your channel.

   > **Note**
   >
   > If you do not create a resource policy for the channel, only the channel owner can call the `PutAuditEvents` API on the channel.

   a. (Optional) Enter a unique external ID to provide an extra layer of protection. The external ID is a unique string such as an account ID or a randomly generated string, to prevent against confused deputy.

      > **Note**
      >
      > If the resource policy includes an external ID, all calls to the `PutAuditEvents` API must include the external ID. However, if the policy does not define an external ID, you can still call the `PutAuditEvents` API and specify an `externalId` parameter.

   b. Choose **Add AWS account** to specify each AWS account ID to add as a principal in the resource policy for the channel.

7. (Optional) In the **Tags** area, you can add up to 50 tag key and value pairs to help you identify, sort, and control access to your event data store and channel. For more information about how to use IAM policies to authorize access to an event data store based on tags, see Examples: Denying access to create or delete event data stores based on tags. For more information about how you can use tags in AWS, see Tagging your AWS resources in the *AWS General Reference*.

8. When you are ready to create the new integration, choose **Add integration**. There is no review page. CloudTrail creates the integration, but to integrate your custom events, you must specify the channel ARN in a `PutAuditEvents` request.

9. Call the PutAuditEvents API to ingest your activity events into CloudTrail. You can add up to 100 activity events (or up to 1 MB) per PutAuditEvents request. You'll need the channel ARN that you created in preceding steps, the payload of events that you want CloudTrail to add, and the external ID (if specified for your resource policy). Be sure that there is no sensitive or personally-identifying information in event payload before ingesting it into CloudTrail. Events that you ingest into CloudTrail must follow the CloudTrail Lake integrations event schema.

   > **Tip**
   >
   > Use AWS CloudShell to be sure you are running the most current AWS APIs.

   The following examples show how to use the **put-audit-events** CLI command. The **--audit-events** and **--channel-arn** parameters are required. You need the ARN of the channel that you created in the preceding steps, which you can copy from the integration details page. The value of **--audit-events** is a JSON array of event objects. --audit-events includes a required ID from the event, the required payload of the event as the value of EventData, and an optional checksum to help validate the integrity of the event after ingestion into CloudTrail.

   ```
   aws cloudtrail-data put-audit-events \
   --region region \
   --channel-arn $ChannelArn \
   --audit-events \
   id="event_ID",eventData='"{event_payload}"' \
   id="event_ID",eventData='"{event_payload}"',eventDataChecksum="optional_checksum"
   ```

   The following is an example command with two event examples.

   ```
   aws cloudtrail-data put-audit-events \
   --region us-east-1 \
   --channel-arn arn:aws:cloudtrail:us-east-1:01234567890:channel/EXAMPLE8-0558-4f7e-
   a06a-43969EXAMPLE \
   --audit-events \
   id="EXAMPLE3-0f1f-4a85-9664-d50a3EXAMPLE",eventData='"{\"eventVersion\":\0.01\",
   \"eventSource\":\"custom1.domain.com\", ...
   \}"' \
   id="EXAMPLE7-a999-486d-b241-b33a1EXAMPLE",eventData='"{\"eventVersion\":\0.02\",
   \"eventSource\":\"custom2.domain.com\", ...
   \}"',eventDataChecksum="EXAMPLE6e7dd61f3ead...93a691d8EXAMPLE"
   ```

The following example command adds the `--cli-input-json` parameter to specify a JSON file (`custom-events.json`) of event payload.

```
aws cloudtrail-data put-audit-events \
--channel-arn $channelArn \
--cli-input-json file://custom-events.json \
--region us-east-1
```

The following are the sample contents of the example JSON file, `custom-events.json`.

```
{
  "auditEvents": [
   {
    "eventData": "{\"version\":\"eventData.version\",\"UID\":\"UID\",
    \"userIdentity\":{\"type\":\"CustomUserIdentity\",\"principalId\":
\"principalId\",
    \"details\":{\"key\":\"value\"}},\"eventTime\":\"2021-10-27T12:13:14Z\",
\"eventName\":\"eventName\",
    \"userAgent\":\"userAgent\",\"eventSource\":\"eventSource\",
    \"requestParameters\":{\"key\":\"value\"},\"responseElements\":{\"key\":
\"value\"},
    \"additionalEventData\":{\"key\":\"value\"},
    \"sourceIPAddress\":\"source_IP_address\",\"recipientAccountId\":
\"recipient_account_ID\"}",
    "id": "1"
   }
 ]
}
```

(Optional) Calculate a checksum value

The checksum that you specify as the value of `EventDataChecksum` in a `PutAuditEvents` request helps you verify that CloudTrail receives the event that matches with the checksum; it helps verify the integrity of events. The checksum value is a base64-SHA256 algorithm that you calculate by running the following command.

```
printf %s "{"eventData": "{\"version\":\"eventData.version\",\"UID\":\"UID\",
    \"userIdentity\":{\"type\":\"CustomUserIdentity\",\"principalId\":\"principalId
\",
        \"details\":{\"key\":\"value\"}},\"eventTime\":\"2021-10-27T12:13:14Z\",
\"eventName\":\"eventName\",
        \"userAgent\":\"userAgent\",\"eventSource\":\"eventSource\",
        \"requestParameters\":{\"key\":\"value\"},\"responseElements\":{\"key\":\"value
\"},
        \"additionalEventData\":{\"key\":\"value\"},
        \"sourceIPAddress\":\"source_IP_address\",
        \"recipientAccountId\":\"recipient_account_ID\"}",
        "id": "1"}" \
 | openssl dgst -binary -sha256 | base64
```

The command returns the checksum. The following is an example.

```
EXAMPLEHjkI8iehvCUCWTIAbNYkOgO/t0YNw+7rrQE=
```

The checksum value becomes the value of `EventDataChecksum` in your `PutAuditEvents` request. If the checksum doesn't match with the one for the provided event, CloudTrail rejects the event with an `InvalidChecksum` error.

Create, update, and manage CloudTrail Lake integrations with the AWS CLI

This section describes the commands you can use to create, update and manage your CloudTrail Lake integrations using the AWS CLI.

When using the AWS CLI, remember that your commands run in the AWS Region configured for your profile. If you want to run the commands in a different Region, either change the default Region for your profile, or use the **--region** parameter with the command.

Available commands for CloudTrail Lake integrations

Commands for creating, updating, and managing integrations in CloudTrail Lake include:

- `create-event-data-store` to create an event data store for events outside of AWS.
- `delete-channel` to delete a channel used for an integration.
- `delete-resource-policy` to delete the resource policy attached to a channel for a CloudTrail Lake integration.
- `get-channel` to return information about a CloudTrail channel.

- `get-resource-policy` to retrieve the JSON text of the resource-based policy document attached to the CloudTrail channel.
- `list-channels` to list the channels in the current account, and their source names.
- `put-audit-events` to ingest your application events into CloudTrail Lake. A required parameter, `auditEvents`, accepts the JSON records (also called payload) of events that you want CloudTrail to ingest. You can add up to 100 of these events (or up to 1 MB) per `PutAuditEvents` request.
- `put-resource-policy` to attach a resource-based permission policy to a CloudTrail channel that is used for an integration with an event source outside of AWS. For more information about resource-based policies, see AWS CloudTrail resource-based policy examples.
- `update-channel` to update a channel specified by a required channel ARN or UUID.

For a list of available commands for CloudTrail Lake event data stores, see Available commands for event data stores.

For a list of available commands for CloudTrail Lake queries, see Available commands for CloudTrail Lake queries.

Create an integration to log events from outside AWS with the AWS CLI

This section describes how you can use the AWS CLI to create a CloudTrail Lake integration to log events from outside of AWS.

In the AWS CLI, you create an integration in four commands (three if you already have an event data store that meets the criteria). Event data stores that you use as the destinations for an integration must be for a single Region and single account; they cannot be multi-region, they cannot log events for organizations in AWS Organizations, and they can only include activity events. The event type in the console must be **Events from integrations**. In the API, the `eventCategory` value must be `ActivityAuditLog`. For more information about integrations, see Create an integration with an event source outside of AWS.

1. Run **create-event-data-store** to create an event data store, if you do not already have one or more event data stores that you can use for the integration.

   The following example AWS CLI command creates an event data store that logs events from outside AWS. For activity events, the `eventCategory` field selector value is `ActivityAuditLog`. The event data store has a retention period of 90 days set. By default, the event data store collects events from all Regions, but because this is collecting non-

AWS events, set it to a single Region by adding the `--no-multi-region-enabled` option. Termination protection is enabled by default, and the event data store does not collect events for accounts in an organization.

```
aws cloudtrail create-event-data-store \
--name my-event-data-store \
--no-multi-region-enabled \
--retention-period 90 \
--advanced-event-selectors '[
    {
      "Name": "Select all external events",
      "FieldSelectors": [
          { "Field": "eventCategory", "Equals": ["ActivityAuditLog"] }
        ]
    }
  ]'
```

The following is an example response.

```
{
    "EventDataStoreArn": "arn:aws:cloudtrail:us-east-1:123456789012:eventdatastore/
EXAMPLEf852-4e8f-8bd1-bcf6cEXAMPLE",
    "Name": "my-event-data-store",
    "AdvancedEventSelectors": [
        {
           "Name": "Select all external events",
           "FieldSelectors": [
              {
                  "Field": "eventCategory",
                  "Equals": [
                      "ActivityAuditLog"
                    ]
                }
            ]
        }
    ],
    "MultiRegionEnabled": true,
    "OrganizationEnabled": false,
    "BillingMode": "EXTENDABLE_RETENTION_PRICING",
    "RetentionPeriod": 90,
    "TerminationProtectionEnabled": true,
    "CreatedTimestamp": "2023-10-27T10:55:55.384000-04:00",
    "UpdatedTimestamp": "2023-10-27T10:57:05.549000-04:00"
}
```

You'll need the event data store ID (the suffix of the ARN, or `EXAMPLEf852-4e8f-8bd1-bcf6cEXAMPLE` in the preceding response example) to go on to the next step and create your channel.

2. Run the **create-channel** command to create a channel that allows a partner or source application to send events to an event data store in CloudTrail.

   A channel has the following components:

   Source

   CloudTrail uses this information to determine the partners that are sending event data to CloudTrail on your behalf. A source is required, and can be either `Custom` for all valid non-AWS events, or the name of a partner event source. A maximum of one channel is allowed per source.

   For information about the `Source` values for available partners, see Additional information about integration partners.

   Ingestion status

   The channel status shows when the last events were received from a channel source.

   Destinations

   The destinations are the CloudTrail Lake event data stores that are receiving events from the channel. You can change destination event data stores for a channel.

   To stop receiving events from a source, delete the channel.

   You need the ID of at least one destination event data store to run this command. The valid type of destination is `EVENT_DATA_STORE`. You can send ingested events to more than one event data store. The following example command creates a channel that sends events to two event data stores, represented by their IDs in the `Location` attribute of the `--destinations` parameter. The `--destinations`, `--name`, and `--source` parameters are required. To ingest events from a CloudTrail partner, specify the name of the partner as the value of `--source`. To ingest events from your own applications outside AWS, specify `Custom` as the value of `--source`.

```
aws cloudtrail create-channel \
    --region us-east-1 \
    --destinations '[{"Type": "EVENT_DATA_STORE", "Location":
 "EXAMPLEf852-4e8f-8bd1-bcf6cEXAMPLE"}, {"Type": "EVENT_DATA_STORE", "Location":
 "EXAMPLEg922-5n2l-3vz1- apqw8EXAMPLE"}]'
    --name my-partner-channel \
    --source $partnerSourceName \
```

In the response to your **create-channel** command, copy the ARN of the new channel. You need the ARN to run the `put-resource-policy` and `put-audit-events` commands in the next steps.

3. Run the **put-resource-policy** command to attach a resource policy to the channel. Resource policies are JSON policy documents that specify what actions a specified principal can perform on the resource and under what conditions. The accounts defined as principals in the channel's resource policy can call the `PutAuditEvents` API to deliver events.

   Note

   If you do not create a resource policy for the channel, only the channel owner can call the `PutAuditEvents` API on the channel.

   The information required for the policy is determined by the integration type.

   - For a direction integration, CloudTrail requires the policy to contain the partner's AWS account IDs, and requires you to enter the unique external ID provided by the partner. CloudTrail automatically adds the partner's AWS account IDs to the resource policy when you create an integration using the CloudTrail console. Refer to the partner's documentation to learn how to get the AWS account numbers required for the policy.
   - For a solution integration, you must specify at least one AWS account ID as principal, and can optionally enter an external ID to prevent against confused deputy.

   The following are requirements for the resource policy:

   - The resource ARN defined in the policy must match the channel ARN the policy is attached to.
   - The policy contains only one action: cloudtrail-data:PutAuditEvents

- The policy contains at least one statement. The policy can have a maximum of 20 statements.
- Each statement contains at least one principal. A statement can have a maximum of 50 principals.

```
aws cloudtrail put-resource-policy \
    --resource-arn "channelARN" \
    --policy "{
    "Version": "2012-10-17",
    "Statement":
    [
        {
            "Sid": "ChannelPolicy",
            "Effect": "Allow",
            "Principal":
            {
                "AWS":
                [
                    "arn:aws:iam::111122223333:root",
                    "arn:aws:iam::444455556666:root",
                    "arn:aws:iam::123456789012:root"
                ]
            },
            "Action": "cloudtrail-data:PutAuditEvents",
            "Resource": "arn:aws:cloudtrail:us-east-1:777788889999:channel/
EXAMPLE-80b5-40a7-ae65-6e099392355b",
            "Condition":
            {
                "StringEquals":
                {
                    "cloudtrail:ExternalId": "UniqueExternalIDFromPartner"
                }
            }
        }
    ]
}"
```

For more information about resource policies, see AWS CloudTrail resource-based policy examples.

4. Run the PutAuditEvents API to ingest your activity events into CloudTrail. You'll need the payload of events that you want CloudTrail to add. Be sure that there is no sensitive or personally-identifying information in event payload before ingesting it into CloudTrail. Note that the PutAuditEvents API uses the cloudtrail-data CLI endpoint, not the cloudtrail endpoint.

   The following examples show how to use the **put-audit-events** CLI command. The **--audit-events** and **--channel-arn** parameters are required. The **--external-id** parameter is required if an external ID is defined in the resource policy. You need the ARN of the channel that you created in the preceding step. The value of **--audit-events** is a JSON array of event objects. --audit-events includes a required ID from the event, the required payload of the event as the value of EventData, and an optional checksum to help validate the integrity of the event after ingestion into CloudTrail.

   ```
   aws cloudtrail-data put-audit-events \
   --channel-arn $ChannelArn \
   --external-id $UniqueExternalIDFromPartner \
   --audit-events \
   id="event_ID",eventData='"{event_payload}"' \
   id="event_ID",eventData='"{event_payload}"',eventDataChecksum="optional_checksum"
   ```

   The following is an example command with two event examples.

   ```
   aws cloudtrail-data put-audit-events \
   --channel-arn arn:aws:cloudtrail:us-east-1:123456789012:channel/EXAMPLE8-0558-4f7e-
   a06a-43969EXAMPLE \
   --external-id UniqueExternalIDFromPartner \
   --audit-events \
   id="EXAMPLE3-0f1f-4a85-9664-d50a3EXAMPLE",eventData='"{\"eventVersion\":\0.01\",
   \"eventSource\":\"custom1.domain.com\", ...
   \}"' \
   id="EXAMPLE7-a999-486d-b241-b33a1EXAMPLE",eventData='"{\"eventVersion\":\0.02\",
   \"eventSource\":\"custom2.domain.com\", ...
   \}"',eventDataChecksum="EXAMPLE6e7dd61f3ead...93a691d8EXAMPLE"
   ```

   The following example command adds the --cli-input-json parameter to specify a JSON file (custom-events.json) of event payload.

```
aws cloudtrail-data put-audit-events --channel-arn $channelArn --external-id
 $UniqueExternalIDFromPartner --cli-input-json file://custom-events.json --region
 us-east-1
```

The following are the sample contents of the example JSON file, `custom-events.json`.

```
{
    "auditEvents": [
      {
        "eventData": "{\"version\":\"eventData.version\",\"UID\":\"UID\",
        \"userIdentity\":{\"type\":\"CustomUserIdentity\",\"principalId\":
\"principalId\",
        \"details\":{\"key\":\"value\"}},\"eventTime\":\"2021-10-27T12:13:14Z\",
\"eventName\":\"eventName\",
        \"userAgent\":\"userAgent\",\"eventSource\":\"eventSource\",
        \"requestParameters\":{\"key\":\"value\"},\"responseElements\":{\"key\":
\"value\"},
        \"additionalEventData\":{\"key\":\"value\"},
        \"sourceIPAddress\":\"12.34.56.78\",\"recipientAccountId\":
\"152089810396\"}",
        "id": "1"
      }
   ]
}
```

You can verify that the integration is working, and CloudTrail is ingesting events from the source correctly, by running the get-channel command. The output of **get-channel** shows the most recent time stamp that CloudTrail received events.

```
aws cloudtrail get-channel --channel arn:aws:cloudtrail:us-east-1:01234567890:channel/
EXAMPLE8-0558-4f7e-a06a-43969EXAMPLE
```

(Optional) Calculate a checksum value

The checksum that you specify as the value of `EventDataChecksum` in a `PutAuditEvents` request helps you verify that CloudTrail receives the event that matches with the checksum; it helps verify the integrity of events. The checksum value is a base64-SHA256 algorithm that you calculate by running the following command.

```
printf %s "{"eventData": "{\"version\":\"eventData.version\",\"UID\":\"UID\",
        \"userIdentity\":{\"type\":\"CustomUserIdentity\",\"principalId\":\"principalId
\",
        \"details\":{\"key\":\"value\"}},\"eventTime\":\"2021-10-27T12:13:14Z\",
\"eventName\":\"eventName\",
        \"userAgent\":\"userAgent\",\"eventSource\":\"eventSource\",
        \"requestParameters\":{\"key\":\"value\"},\"responseElements\":{\"key\":\"value
\"},
        \"additionalEventData\":{\"key\":\"value\"},
        \"sourceIPAddress\":\"source_IP_address\",
        \"recipientAccountId\":\"recipient_account_ID\"}",
        "id": "1"}" \
 | openssl dgst -binary -sha256 | base64
```

The command returns the checksum. The following is an example.

```
EXAMPLEDHjkI8iehvCUCWTIAbNYkOgO/t0YNw+7rrQE=
```

The checksum value becomes the value of `EventDataChecksum` in your `PutAuditEvents` request. If the checksum doesn't match with the one for the provided event, CloudTrail rejects the event with an `InvalidChecksum` error.

Update a channel with the AWS CLI

This section describes how you can use the AWS CLI to update a channel for a CloudTrail Lake integration. You can run the `update-channel` command to update the name of the channel or to specify a different destination event data store. You cannot update the source of a channel.

When you run the command, the `--channel` parameter is required.

The following is an example that demonstrates how to update the channel name and destination.

```
aws cloudtrail update-channel \
--channel aws:cloudtrail:us-east-1:123456789012:channel/EXAMPLE8-0558-4f7e-
a06a-43969EXAMPLE \
--name "new-channel-name" \
--destinations '[{"Type": "EVENT_DATA_STORE", "Location": "EXAMPLEf852-4e8f-8bd1-
bcf6cEXAMPLE"}, {"Type": "EVENT_DATA_STORE", "Location": "EXAMPLEg922-5n2l-3vz1-
 apqw8EXAMPLE"}]'
```

Delete a channel to delete an integration with the AWS CLI

This section describes how to run the `delete-channel` command to delete the channel for a CloudTrail Lake integration. You would delete a channel, if you wanted to stop ingesting partner or other activity events outside of AWS. The ARN or channel ID (the ARN suffix) of the channel that you want to delete is required.

The following example shows how to delete the channel.

```
aws cloudtrail delete-channel \
--channel EXAMPLE8-0558-4f7e-a06a-43969EXAMPLE
```

Additional information about integration partners

The table in this section provides the source name for each integration partner and identifies the integration type (direct or solution).

The information in the **Source name** column is required when calling the `CreateChannel` API. You specify the source name as the value for the `Source` parameter.

Partner name (console)	Source name (API)	Integration type
My custom integration	`Custom`	solution
Cloud Storage Security	`CloudStorageSecurityConsole`	solution
Clumio	`Clumio`	direct
CrowdStrike	`CrowdStrike`	solution
CyberArk	`CyberArk`	solution
GitHub	`GitHub`	solution
Kong Inc	`KongGatewayEnterprise`	solution
LaunchDarkly	`LaunchDarkly`	direct

Partner name (console)	Source name (API)	Integration type
Netskope	`NetskopeCloudExchange`	solution
Nordcloud, an IBM Company	`IBMMulticloud`	direct
MontyCloud	`MontyCloud`	direct
Okta	`OktaSystemLogEvents`	solution
One Identity	`OneLogin`	solution
Shoreline.io	`Shoreline`	solution
Snyk.io	`Snyk`	direct
Wiz	`WizAuditLogs`	solution

View partner documentation

You can learn more about a partner's integration with CloudTrail Lake by viewing their documentation.

To view partner documentation

1. Sign in to the AWS Management Console and open the CloudTrail console at https://console.aws.amazon.com/cloudtrail/.
2. From the navigation pane, under **Lake**, choose **Integrations**.
3. From the **Integrations** page, choose **Available sources**, then choose **Learn more** for the partner whose documentation you want to view.

CloudTrail Lake integrations event schema

The following table describes the required and optional schema elements that match those in CloudTrail event records. The contents of `eventData` are provided by your events; other fields are provided by CloudTrail after ingestion.

CloudTrail event record contents are described in more detail in CloudTrail record contents.

- Fields that are provided by CloudTrail after ingestion
- Fields that are provided by your events

The following fields are provided by CloudTrail after ingestion:

Field name	Input type	Requirement	Description
eventVersion	string	Required	The event version.
eventCategory	string	Required	The event category. For non-AWS events, the value is `ActivityAuditLog` .
eventType	string	Required	The event type. For non-AWS events, the valid value is `ActivityLog` .
eventID	string	Required	A unique ID for an event.
eventTime	string	Required	Event timestamp , in `yyyy-MM-DDTHH:mm:ss` format, in Universal Coordinated Time (UTC).
awsRegion	string	Required	The AWS Region where the `PutAuditEvents` call was made.
recipientAccountId	string	Required	Represents the account ID that

Field name	Input type	Requirement	Description
			received this event. CloudTrail populates this field by calculating it from event payload.
addendum	-	Optional	Shows information about why event processing was delayed. If information was missing from an existing event, the addendum block includes the missing information and a reason for why it was missing.
• reason	string	Optional	The reason that the event or some of its contents were missing.
• updatedFields	string	Optional	The event record fields that are updated by the addendum. This is only provided if the reason is `UPDATED_DATA` .
• originalUID	string	Optional	The original event UID from the source. This is only provided if the reason is `UPDATED_DATA` .

Field name	Input type	Requirement	Description
• originalEventID	string	Optional	The original event ID. This is only provided if the reason is `UPDATED_DATA` .
metadata	-	Required	Information about the channel that the event used.
• ingestionTime	string	Required	The timestamp when the event was processed, in `yyyy-MM-DDTHH:mm:ss` format, in Universal Coordinated Time (UTC).
• channelARN	string	Required	The ARN of the channel that the event used.

The following fields are provided by customer events:

Field name	Input type	Requirement	Description
eventData	-	Required	The audit data sent to CloudTrail in a `PutAuditEvents` call.
• version	string	Required	The version of the event from its source.

Field name	Input type	Requirement	Description
			Length constraints: Maximum length of 256.
• userIdentity	-	Required	Information about the user who made a request.
• • type	string	Required	The type of user identity. Length constraints: Maximum length of 128.
• • principalId	string	Required	A unique identifier for the actor of the event. Length constraints: Maximum length of 1024.
• • details	JSON object	Optional	Additional information about the identity.
• userAgent	string	Optional	The agent through which the request was made. Length constraints: Maximum length of 1024.

Field name	Input type	Requirement	Description
- eventSource	string	Required	This is the partner event source, or the custom application about which events are logged. Length constraints: Maximum length of 1024.
- eventName	string	Required	The requested action, one of the actions in the API for the source service or application. Length constraints: Maximum length of 1024.
- eventTime	string	Required	Event timestamp , in `yyyy-MM-DDTHH:mm:ss` format, in Universal Coordinated Time (UTC).
- UID	string	Required	The UID value that identifies the request. The service or application that is called generates this value. Length constraints: Maximum length of 1024.

Field name	Input type	Requirement	Description
• requestParameters	JSON object	Optional	The parameters, if any, that were sent with the request. This field has a maximum size of 100 kB, and content exceeding the limit is rejected.
• responseElements	JSON object	Optional	The response element for actions that make changes (create, update, or delete actions). This field has a maximum size of 100 kB, and content exceeding the limit is rejected.
• errorCode	string	Optional	A string represent ing an error for the event. Length constraints: Maximum length of 256.
• errorMessage	string	Optional	The description of the error. Length constraints: Maximum length of 256.

Field name	Input type	Requirement	Description
• sourceIPAddress	string	Optional	The IP address from which the request was made. Both IPv4 and IPv6 addresses are accepted.
• recipientAccountId	string	Required	Represents the account ID that received this event. The account ID must be the same as the AWS account ID that owns the channel.
• additionalEventData	JSON object	Optional	Additional data about the event that was not part of the request or response. This field has a maximum size of 28 kB, and content exceeding that limit is rejected.

The following example shows the hierarchy of schema elements that match those in CloudTrail event records.

```
{
    "eventVersion": String,
    "eventCategory": String,
    "eventType": String,
    "eventID": String,
    "eventTime": String,
    "awsRegion": String,
    "recipientAccountId": String,
    "addendum": {
       "reason": String,
       "updatedFields": String,
       "originalUID": String,
       "originalEventID": String
    },
    "metadata" : {
       "ingestionTime": String,
       "channelARN": String
    },
    "eventData": {
        "version": String,
        "userIdentity": {
          "type": String,
          "principalId": String,
          "details": {
             JSON
          }
        },
        "userAgent": String,
        "eventSource": String,
        "eventName": String,
        "eventTime": String,
        "UID": String,
        "requestParameters": {
           JSON
        },
        "responseElements": {
           JSON
        },
        "errorCode": String,
        "errorMessage": String,
        "sourceIPAddress": String,
        "recipientAccountId": String,
        "additionalEventData": {
           JSON
        }
    }
}
```

View CloudTrail Lake dashboards with the CloudTrail console

You can use CloudTrail Lake dashboards to visualize the events in an event data store. You can select from several different dashboard types. The dashboard types available for an event data store are dependent upon the advanced event selectors configuration of the event data store. For example, if a dashboard type displays information about CloudTrail management events, you can only select the dashboard if the currently selected event data store collects CloudTrail management events.

Each dashboard type consists of multiple widgets and each widget represents a SQL query. To view the query for a widget, choose **View and analyze in query editor** to open up the query editor. You can't modify the system-generated query that is used to populate the widget, but you can make edits to the query and run the query in the query editor for further analysis.

To populate and update a dashboard, choose **Run queries**. When you choose **Run queries**, CloudTrail runs system-generated queries on your behalf. Because running queries incur costs, CloudTrail asks you to acknowledge the costs associated with running queries. This is a one time confirmation. For more information about CloudTrail pricing, see CloudTrail Pricing.

Topics

Limitations

The following limitations apply to the current release.

- The current release doesn't support customized dashboards, widgets, or queries.
- The current release only provides dashboards for event data stores that collect CloudTrail events (data events, management events) and Insights events.
- The current release doesn't support editing the system-generated queries used to populate the dashboard. You can view and edit the underlying query for any widget on the **Query Editor** tab, however, any changes you make to the query are intended for supplemental analysis outside of the dashboard.

Prerequisites

The following prerequisites apply to Lake dashboards.

- To view and use Lake dashboards, you must create at least one CloudTrail Lake event data store. You can create event data stores using the console, AWS CLI, or SDKs. For information about creating an event data store using the console, see Create an event data store for CloudTrail events with the console. For information about creating an event data store using the AWS CLI, see Create, update, and manage event data stores with the AWS CLI.
- To populate the dashboard, CloudTrail runs queries on your behalf. The first time you view the **Dashboards** page, CloudTrail asks you to acknowledge the costs associated with running queries. Choose **I agree** to acknowledge the cost of running queries.

Choosing a dashboard

Use the following procedure to choose an event data store and dashboard type to view.

1. Sign in to the AWS Management Console and open the CloudTrail console at https://console.aws.amazon.com/cloudtrail/.
2. In the left navigation pane, under **Lake**, choose **Dashboard**.
3. Choose the event data store for which you want to visualize data.
4. Choose the dashboard type you want to view. The dashboards list is populated based upon the advanced event selectors configuration of the selected event data store.

   The following are the possible dashboard types.

   - **Overview** dashboard - Shows the most active users, AWS Regions, and AWS services by event count. You can also view information about `read` and `write` management event activity, most throttled events, and the top errors. This dashboard is available for event data stores that collect management events.
   - **Management Events** dashboard - Shows console sign-in events, access denied events, destructive actions, and top errors by user. You can also view information about TLS versions and outdated TLS calls by user. This dashboard is available for event data stores that collect management events.
   - **S3 Data Events** dashboard - Shows S3 account activity, most accessed S3 objects, top S3 users, and top S3 actions. This dashboard is available for event data stores that collect Amazon S3 data events.

- **Insights Events** dashboard - Shows the overall proportion of Insights events by Insights type, the proportion of Insights events by Insights type for the top users and services, and the number of Insights events per day. The dashboard also includes a widget that lists up to 30 days of Insights events. This dashboard is only available for event data stores that collect Insights events.

> **Note**
>
> - After you enable CloudTrail Insights for the first time on the source event data store, it can take up to 7 days for CloudTrail to deliver the first Insights event, if unusual activity is detected. For more information, see Understanding Insights events delivery.
> - The **Insights Events** dashboard only displays information about the Insights events collected by the selected event data store, which is determined by the configuration of the source event data store. For example, if you configure the source event data store to enable Insights events on `ApiCallRateInsight` but not `ApiErrorRateInsight`, you won't see information about Insights events on `ApiErrorRateInsight`.

5. Choose to filter the dashboard data by an **Absolute range** or **Relative range**. Choose **Absolute range** to select a specific date and time range. Choose **Relative range** to select a predefined time range or a custom range. By default, the dashboard displays event data for the past 24 hours.

   > **Note**
   >
   > CloudTrail Lake queries incur costs based upon the amount of data scanned. To help control costs, you can filter on a narrower time range. For more information about CloudTrail pricing, see AWS CloudTrail Pricing.

6. Choose **Run queries** to run the queries for the dashboard's widgets.

Filtering a dashboard on a date or time range

By default, the dashboard displays data for the past 24 hours. You can filter a dashboard by an **Absolute range** or **Relative range**.

Choose **Absolute range** to select a specific date and time range.

Choose **Relative range** to select a predefined time range or a custom range.

After you've chosen the time range, choose **Run queries** to refresh the dashboard.

> **Note**
>
> CloudTrail Lake queries incur costs based upon the amount of data scanned. To help control costs, you can filter on a narrower time range. For more information about CloudTrail pricing, see AWS CloudTrail Pricing.

Viewing the query for a dashboard widget

Each widget represents a SQL query. To view the query for a widget, choose **View and analyze in query editor** to open up the query editor. Using the query editor, you can further refine the query outside the dashboard and run the query to see the results of your updated query. For more information about working with queries, see Create or edit a query with the CloudTrail console.

> **Note**
>
> You cannot modify the system-generated query for a dashboard widget. Any changes made to the query on the **Query Editor** tab are intended solely for further analysis outside of the dashboard.

CloudTrail Lake queries

> **Note**
>
> Introducing a preview feature for CloudTrail Lake queries that uses generative artificial intelligence (generative AI) capabilities to produce a SQL query from an English language prompt. For more information, see Create CloudTrail Lake queries from English language prompts.

Queries in CloudTrail Lake are authored in SQL. You can build a query on the CloudTrail Lake **Editor** tab by writing the query in SQL from scratch, or by opening a saved or sample query and editing it. You cannot overwrite an included sample query with your changes, but you can save it as a new query. For more information about the SQL query language that is allowed, see CloudTrail Lake SQL constraints.

An unbounded query (such as `SELECT * FROM edsID`) scans all data in your event data store. To help control costs, we recommend that you constrain queries by adding starting and ending `eventTime` time stamps to queries. The following is an example that searches for all events in a specified event data store where the event time is after (>) January 5, 2023 at 1:51 p.m. and before (<) January 19, 2023 at 1:51 p.m. Because an event data store has a minimum retention period of seven days, the minimum time span between starting and ending `eventTime` values is also seven days.

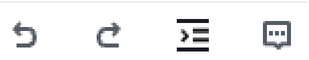
```
SELECT *
FROM eds-ID
WHERE
     eventtime >='2023-01-05 13:51:00' and eventtime < ='2023-01-19 13:51:00'
```

Topics

- Query editor tools
- Create CloudTrail Lake queries from English language prompts
- View sample queries with the CloudTrail console
- Create or edit a query with the CloudTrail console
- Run a query and save query results with the console
- View query results with the console
- Download saved query results
- Validate CloudTrail Lake saved query results
- Run and manage CloudTrail Lake queries with the AWS CLI

Query editor tools

A toolbar at the upper right of the query editor offers commands to help author and format your SQL query.



The following list describes the commands on the toolbar.

- **Undo –** Reverts the last content change made in the query editor.
- **Redo –** Repeats the last content change made in the query editor.
- **Format selected –** Arranges the query editor content according to SQL formatting and spacing conventions.
- **Comment/uncomment selected** - Comments the selected portion of the query if it is not already commented. If the selected portion is already commented, choosing this option removes the comment.

Create CloudTrail Lake queries from English language prompts

Note

The query generator is in preview release in US East (N. Virginia) for CloudTrail Lake and is subject to change.

You can use the CloudTrail Lake query generator to produce a query from an English language prompt that you provide. The query generator uses generative artificial intelligence (generative AI) to produce a ready-to-use SQL query from your prompt, which you can then choose to run in Lake's query editor, or further fine tune. You don't need to have extensive knowledge of SQL or CloudTrail event fields to use the query generator.

The prompt can be a question or a statement about the event data in your CloudTrail Lake event data store. For example, you can enter prompts like "What are my top errors in the past month?" and “Give me a list of users that used SNS.”

A prompt can have a minimum of 3 characters and a maximum of 500 characters.

There are no charges for generating queries; however, when you run queries, you incur charges based on the amount of optimized and compressed data scanned. To help control costs, we recommend that you constrain queries by adding starting and ending `eventTime` timestamps to queries.

Note

You can provide feedback about a generated query by choosing the thumbs up or thumbs down button that appears below the generated query. When you provide feedback, CloudTrail saves your prompt and the generated query.

Do not include any personally identifying, confidential, or sensitive information in your prompts.
This feature uses generative AI large language models (LLMs); we recommend double-checking the LLM response.

To use the query generator on the CloudTrail console

1. Sign in to the AWS Management Console and open the CloudTrail console at https://console.aws.amazon.com/cloudtrail/.
2. From the navigation pane, under **Lake**, choose **Query**.
3. On the **Query** page, choose the **Editor** tab.
4. Choose the event data store you want to create a query for.
5. In the **Query generator** area, enter a prompt in plain English. For examples, see Example prompts.
6. Choose **Generate query**. The query generator will attempt to generate a query from your prompt. If successful, the query generator provides the SQL query in the editor. If the prompt is unsuccessful, rephrase your prompt and try again.
7. (Optional) Choose **Run** to run the query.

   > **Note**
   >
   > When you run queries, you incur charges based on the amount of optimized and compressed data scanned. To help control costs, we recommend that you constrain queries by adding starting and ending `eventTime` timestamps to queries.

8. (Optional) You can provide your feedback about the generated query. To provide feedback, choose the thumbs up or thumbs down button that appears below the prompt. When you provide feedback, CloudTrail saves your prompt and the generated query.

Required permissions

To use this feature, you must use the `AWSCloudTrail_FullAccess` or `AdministratorAccess` policy.

You can also include the `cloudtrail:GenerateQuery` action in a new or existing customer managed or inline policy.

Region support

This feature is currently only available in the US East (N. Virginia) Region.

Limitations

The following are limitations of the query generator:

- The query generator can only accept prompts in English.
- The query generator can only generate queries for event data stores that collect CloudTrail events (management events, data events).
- The query generator cannot generate queries for prompts that do not pertain to CloudTrail Lake event data.

Example prompts

This section provides example prompts and the resulting SQL queries generated from the prompts.

If you choose to run the example queries in this section, replace *`eds-id`* with the ID of the event data store that you want to query and replace the timestamps with the appropriate timestamps for your use case. Timestamps have the following format: `YYYY-MM-DD HH:MM:SS`.

Prompt: What are my top errors in the past month?

SQL query:

```
SELECT
    errorMessage,
    COUNT(*) as eventCount
FROM
    eds-id
WHERE
    errorMessage IS NOT NULL
AND eventTime >= timestamp '2024-05-01 00:00:00'
AND eventTime <= timestamp '2024-05-31 23:59:59'
GROUP BY 1
ORDER BY 2 DESC
LIMIT 2;
```

Prompt: Give me a list of users that used SNS.

SQL query:

```
SELECT
    DISTINCT userIdentity.arn AS user
FROM
    eds-id
WHERE
    eventSource = 'sns.amazonaws.com'
```

Prompt: What are my API counts each day for read and write events in the past month?

SQL query:

```
SELECT date(eventTime) AS event_date,
    SUM(
        CASE
            WHEN readonly = true THEN 1
            ELSE 0
        END
    ) AS read_events,
    SUM(
        CASE
            WHEN readonly = false THEN 1
            ELSE 0
        END
    ) AS write_events
FROM
    eds-id
WHERE
    eventTime >= timestamp '2024-05-04 00:00:00'
AND eventTime <= timestamp '2024-06-04 23:59:59'
GROUP BY 1
ORDER BY 1 ASC;
```

Prompt: Show any events with access denied errors for the past three weeks.

SQL query:

```
SELECT *
FROM
  eds-id
WHERE
```

```
  WHERE (errorCode = 'AccessDenied' OR errorMessage = 'Access Denied')
AND eventTime >= timestamp '2024-05-16 01:00:00'
AND eventTime <= timestamp '2024-06-06 01:00:00'
```

View sample queries with the CloudTrail console

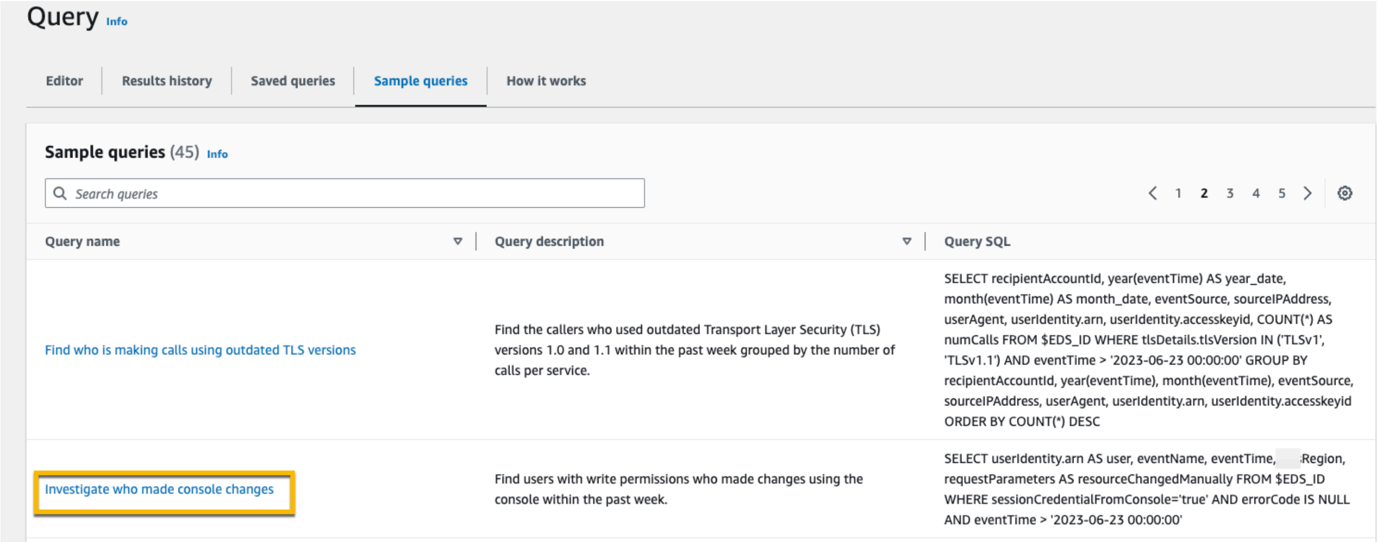
The CloudTrail console provides a number of sample queries that can help you get started writing your own queries.

CloudTrail queries incur charges based upon the amount of data scanned. To help control costs, we recommend that you constrain queries by adding starting and ending `eventTime` time stamps to queries. For more information about CloudTrail pricing, see AWS CloudTrail Pricing.

> **Note**
>
> You can also view queries created by the GitHub community. For more information, see CloudTrail Lake sample queries on the GitHub website. AWS CloudTrail has not evaluated the queries in GitHub.

To view and run a sample query

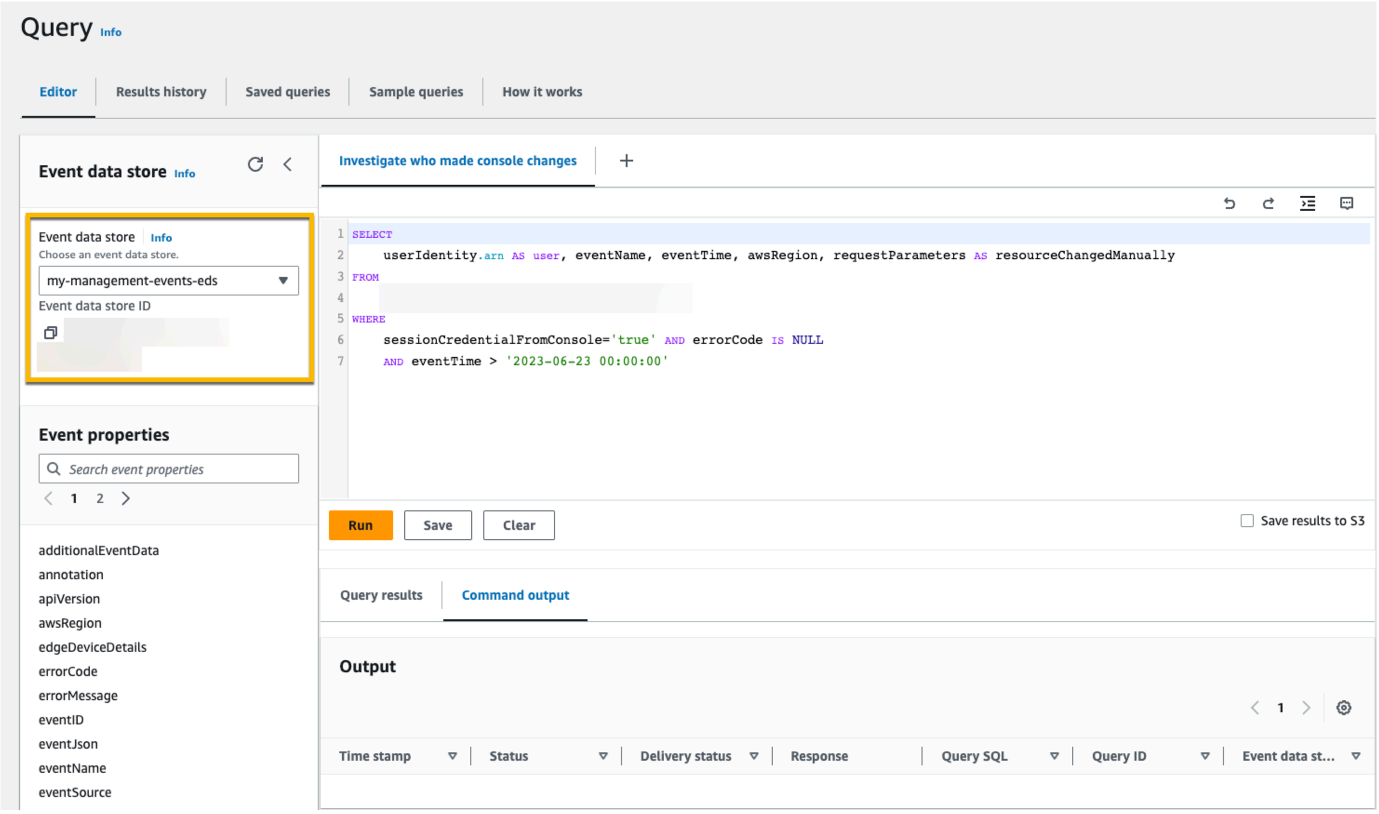
1. Sign in to the AWS Management Console and open the CloudTrail console at https://console.aws.amazon.com/cloudtrail/.
2. From the navigation pane, under **Lake**, choose **Query**.
3. On the **Query** page, choose the **Sample queries** tab.
4. Choose a sample query from the list or search for the query to filter the list. In this example, we'll open the query **Investigate who made console changes** by choosing the **Query name**. This opens the query in the **Editor** tab.



Sample queries (45) Info

Query name	Query description	Query SQL
Find who is making calls using outdated TLS versions	Find the callers who used outdated Transport Layer Security (TLS) versions 1.0 and 1.1 within the past week grouped by the number of calls per service.	SELECT recipientAccountId, year(eventTime) AS year_date, month(eventTime) AS month_date, eventSource, sourceIPAddress, userAgent, userIdentity.arn, userIdentity.accesskeyid, COUNT(*) AS numCalls FROM $EDS_ID WHERE tlsDetails.tlsVersion IN ('TLSv1', 'TLSv1.1') AND eventTime > '2023-06-23 00:00:00' GROUP BY recipientAccountId, year(eventTime), month(eventTime), eventSource, sourceIPAddress, userAgent, userIdentity.arn, userIdentity.accesskeyid ORDER BY COUNT(*) DESC
Investigate who made console changes	Find users with write permissions who made changes using the console within the past week.	SELECT userIdentity.arn AS user, eventName, eventTime, [redacted]Region, requestParameters AS resourceChangedManually FROM $EDS_ID WHERE sessionCredentialFromConsole='true' AND errorCode IS NULL AND eventTime > '2023-06-23 00:00:00'

5. On the **Editor** tab, choose the event data store for which you want to run the query. When you choose the event data store from the list, CloudTrail automatically populates the event data store ID in the FROM line of the query editor.



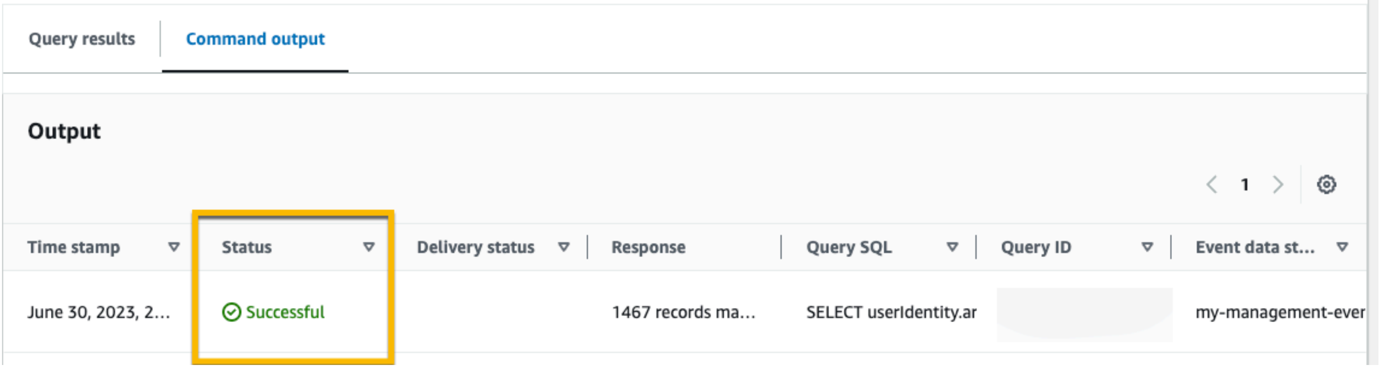
Event data store Info
Choose an event data store.
my-management-events-eds
Event data store ID
[redacted]

```
SELECT
    userIdentity.arn AS user, eventName, eventTime, awsRegion, requestParameters AS resourceChangedManually
FROM
    [redacted]
WHERE
    sessionCredentialFromConsole='true' AND errorCode IS NULL
    AND eventTime > '2023-06-23 00:00:00'
```

Time stamp	Status	Delivery status	Response	Query SQL	Query ID	Event data st...

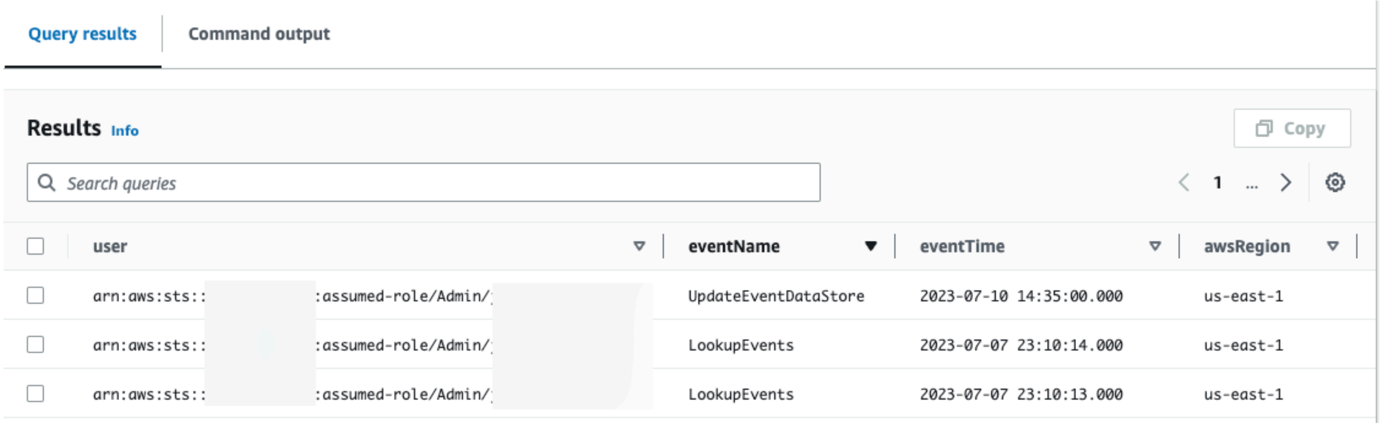
6. Choose **Run** to run the query.

   The **Command output** tab shows you metadata about your query, such as whether the query was successful, the number of records matched, and the run time of the query.



Time stamp	Status	Delivery status	Response	Query SQL	Query ID	Event data st...
June 30, 2023, 2...	Successful		1467 records ma...	SELECT userIdentity.ar	[redacted]	my-management-ever

The **Query results** tab shows you the event data in the selected event data store that matched your query.



□	user	eventName	eventTime	awsRegion
□	arn:aws:sts::[redacted]:assumed-role/Admin/[redacted]	UpdateEventDataStore	2023-07-10 14:35:00.000	us-east-1
□	arn:aws:sts::[redacted]:assumed-role/Admin/[redacted]	LookupEvents	2023-07-07 23:10:14.000	us-east-1
□	arn:aws:sts::[redacted]:assumed-role/Admin/[redacted]	LookupEvents	2023-07-07 23:10:13.000	us-east-1

For more information about editing a query, see Create or edit a query with the CloudTrail console. For more information about running a query and saving query results, see Run a query and save query results with the console.

Create or edit a query with the CloudTrail console

Note

Introducing a preview feature for CloudTrail Lake queries that uses generative artificial intelligence (generative AI) capabilities to produce a SQL query from an English language prompt. For more information, see Create CloudTrail Lake queries from English language prompts.

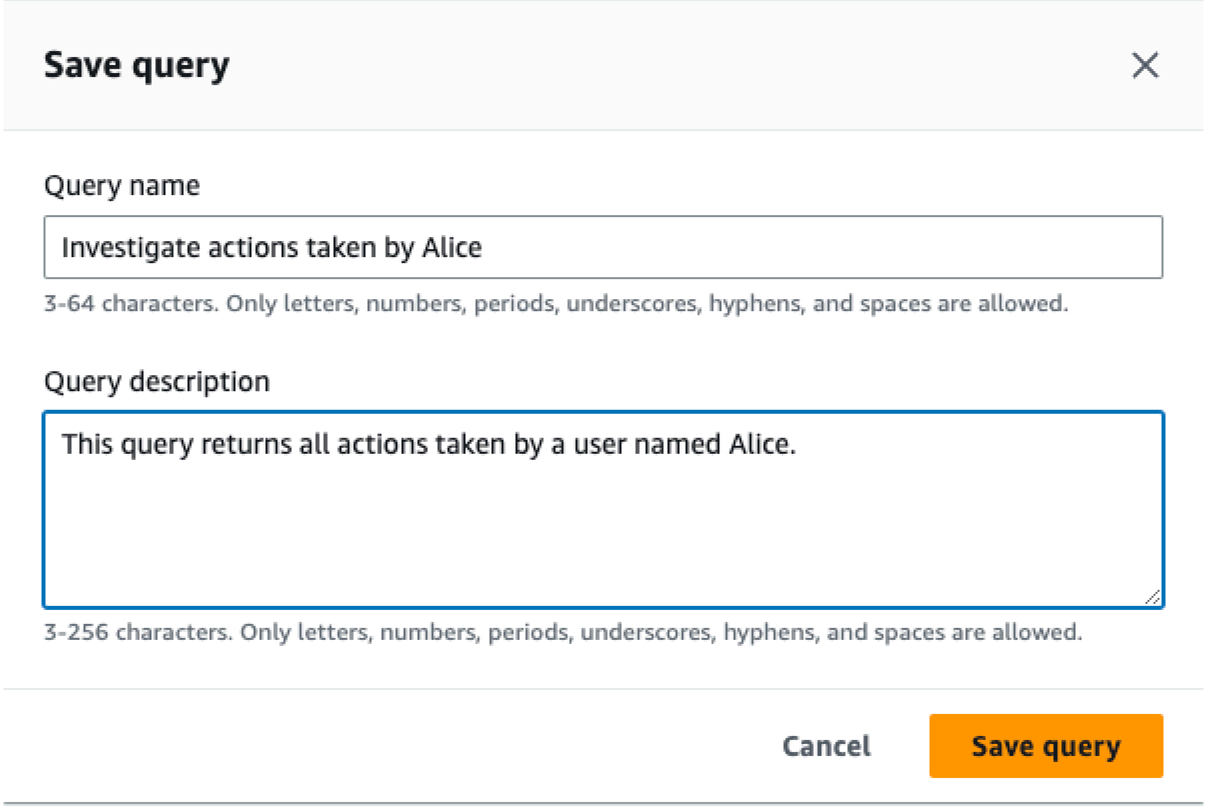
In this walkthrough, we open one of the sample queries, edit it to find actions taken by a specific user named `Alice`, and save it as a new query. You can also edit a saved query on the **Saved**

queries tab, if you have saved queries. To help control costs, we recommend that you constrain queries by adding starting and ending `eventTime` time stamps to queries.

1. Sign in to the AWS Management Console and open the CloudTrail console at https://console.aws.amazon.com/cloudtrail/.
2. From the navigation pane, under **Lake**, choose **Query**.
3. On the **Query** page, choose the **Sample queries** tab.
4. Open a sample query by choosing the **Query name**. This opens the query in the **Editor** tab. In this example, we'll select the query named **Investigate user actions** and edit the query to find the actions for a specific user named `Alice`.
5. In the **Editor** tab, edit the `WHERE` line to specify the user that you want to investigate and update the `eventTime` values as needed. The value of `FROM` is the ID portion of the event data store's ARN and is automatically populated by CloudTrail when you choose the event data store.

```
SELECT
    eventID, eventName, eventSource, eventTime, userIdentity.arn AS user
FROM
    event-data-store-id
WHERE
    userIdentity.arn LIKE '%Alice%'
    AND eventTime > '2023-06-23 00:00:00' AND eventTime < '2023-06-26 00:00:00'
```

6. You can run a query before you save it, to verify that the query works. To run a query, choose an event data store from the **Event data store** drop-down list, and then choose **Run**. View the **Status** column of the **Command output** tab for the active query to verify that a query ran successfully.
7. When you have updated the sample query, choose **Save**.
8. In **Save query**, enter a name and description for the query. Choose **Save query** to save your changes as the new query. To discard changes to a query, choose **Cancel**, or close the **Save query** window.



Query name

Investigate actions taken by Alice

3-64 characters. Only letters, numbers, periods, underscores, hyphens, and spaces are allowed.

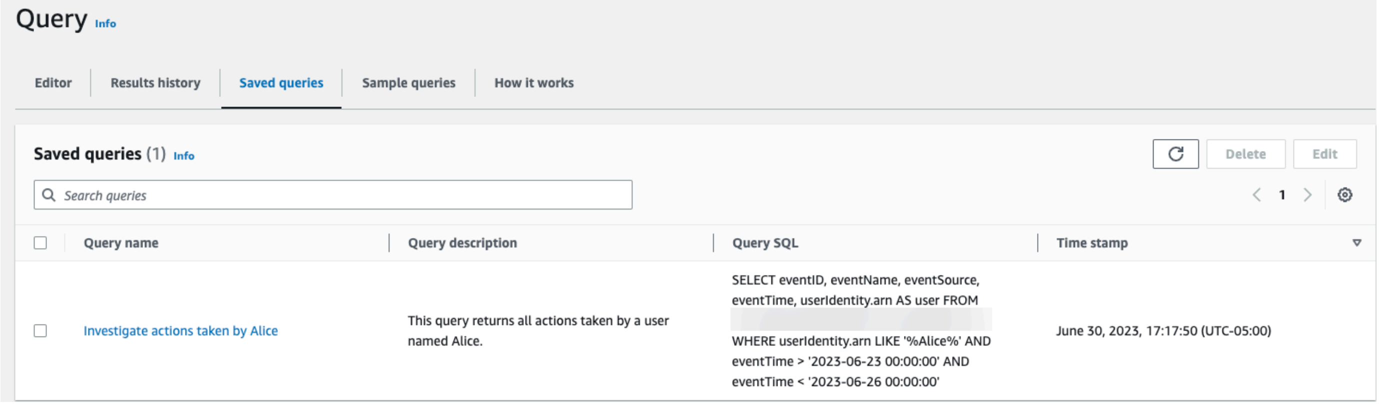
Query description

This query returns all actions taken by a user named Alice.

3-256 characters. Only letters, numbers, periods, underscores, hyphens, and spaces are allowed.

> **Note**
>
> Saved queries are tied to your browser; if you use a different browser or a different device to access the CloudTrail console, the saved queries are not available.

9. Open the **Saved queries** tab to see the new query in the table.



Saved queries (1) Info

☐	Query name	Query description	Query SQL	Time stamp
☐	Investigate actions taken by Alice	This query returns all actions taken by a user named Alice.	SELECT eventID, eventName, eventSource, eventTime, userIdentity.arn AS user FROM [redacted] WHERE userIdentity.arn LIKE '%Alice%' AND eventTime > '2023-06-23 00:00:00' AND eventTime < '2023-06-26 00:00:00'	June 30, 2023, 17:17:50 (UTC-05:00)

Run a query and save query results with the console

Note

Introducing a preview feature for CloudTrail Lake queries that uses generative artificial intelligence (generative AI) capabilities to produce a SQL query from an English language prompt. For more information, see Create CloudTrail Lake queries from English language prompts.

After you choose or save a query, you can run a query on an event data store.

When you run a query, you have the option to save the query results to an Amazon S3 bucket. When you run queries in CloudTrail Lake, you incur charges based on the amount of data scanned by the query. There are no additional CloudTrail Lake charges for saving query results to an S3 bucket, however, there are S3 storage charges. For more information about S3 pricing, see Amazon S3 pricing.

When you save query results, the query results may display in the CloudTrail console before they are viewable in the S3 bucket since CloudTrail delivers the query results after the query scan completes. While most queries complete within a few minutes, depending on the size of your event data store, it can take considerably longer for CloudTrail to deliver query results to your S3 bucket. CloudTrail delivers the query results to the S3 bucket in compressed gzip format. On average, after the query scan completes you can expect a latency of 60 to 90 seconds for every GB of data delivered to the S3 bucket.

To run a query using CloudTrail Lake

1. Sign in to the AWS Management Console and open the CloudTrail console at https://console.aws.amazon.com/cloudtrail/.
2. From the navigation pane, under **Lake**, choose **Query**.
3. On the **Saved queries** or **Sample queries** tabs, choose a query to run by choosing the **Query name**.
4. On the **Editor** tab, for **Event data store**, choose an event data store from the drop-down list.
5. (Optional) On the **Editor** tab, choose **Save results to S3** to save the query results to an S3 bucket. When you choose the default S3 bucket, CloudTrail creates and applies the required bucket policies. If you choose the default S3 bucket, your IAM policy needs to include

permission for the `s3:PutEncryptionConfiguration` action because by default server-side encryption is enabled for the bucket. For more information about saving query results, see Additional information about saved query results.

> **Note**
>
> To use a different bucket, specify a bucket name, or choose **Browse S3** to choose a bucket. The bucket policy must grant CloudTrail permission to deliver query results to the bucket. For information about manually editing the bucket policy, see Amazon S3 bucket policy for CloudTrail Lake query results.

6. On the **Editor** tab, choose **Run**.

   Depending on the size of your event data store, and the number of days of data it includes, a query can take several minutes to run. The **Command output** tab shows the status of a query, and whether a query is finished running. When a query has finished running, open the **Query results** tab to see a table of results for the active query (the query currently shown in the editor).

> **Note**
>
> Queries that run for longer than one hour might time out. You can still get partial results that were processed before the query timed out. CloudTrail does not deliver partial query results to an S3 bucket. To avoid a time out, you can refine your query to limit the amount of data scanned by specifying a narrower time range.

Additional information about saved query results

After you save query results, you can download the saved query results from the S3 bucket. For more information about finding and downloading saved query results, see Download saved query results.

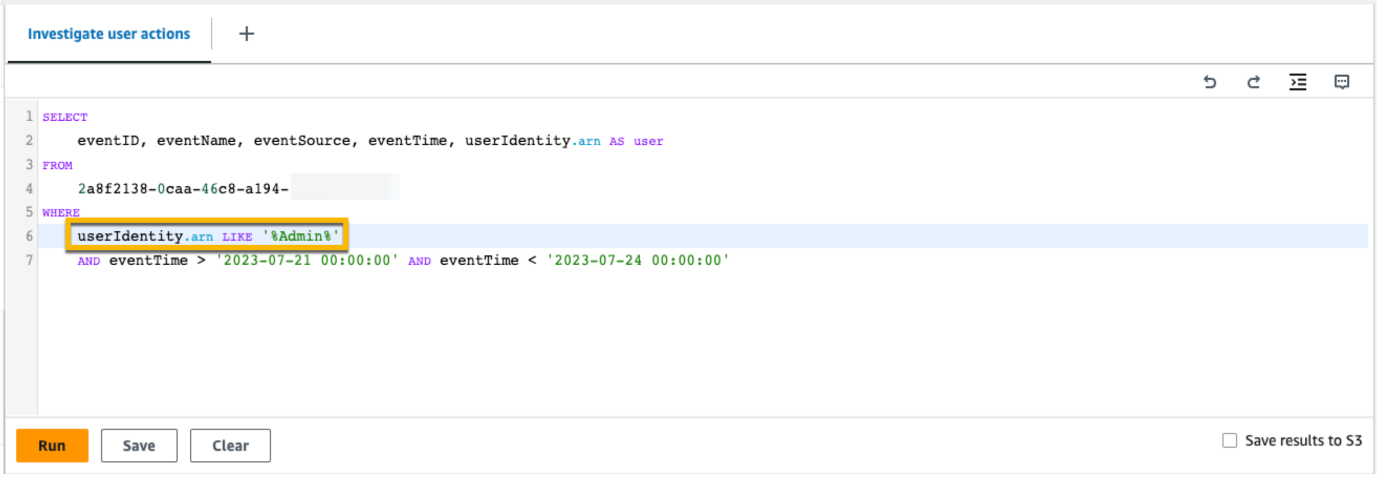
You can also validate saved query results to determine whether the query results were modified, deleted, or unchanged after CloudTrail delivered the query results. For more information about validating saved query results, see Validate CloudTrail Lake saved query results.

Example: Save query results to an Amazon S3 bucket

This walkthrough shows how you can save query results to an S3 bucket and then download those query results.

To save query results to an Amazon S3 bucket

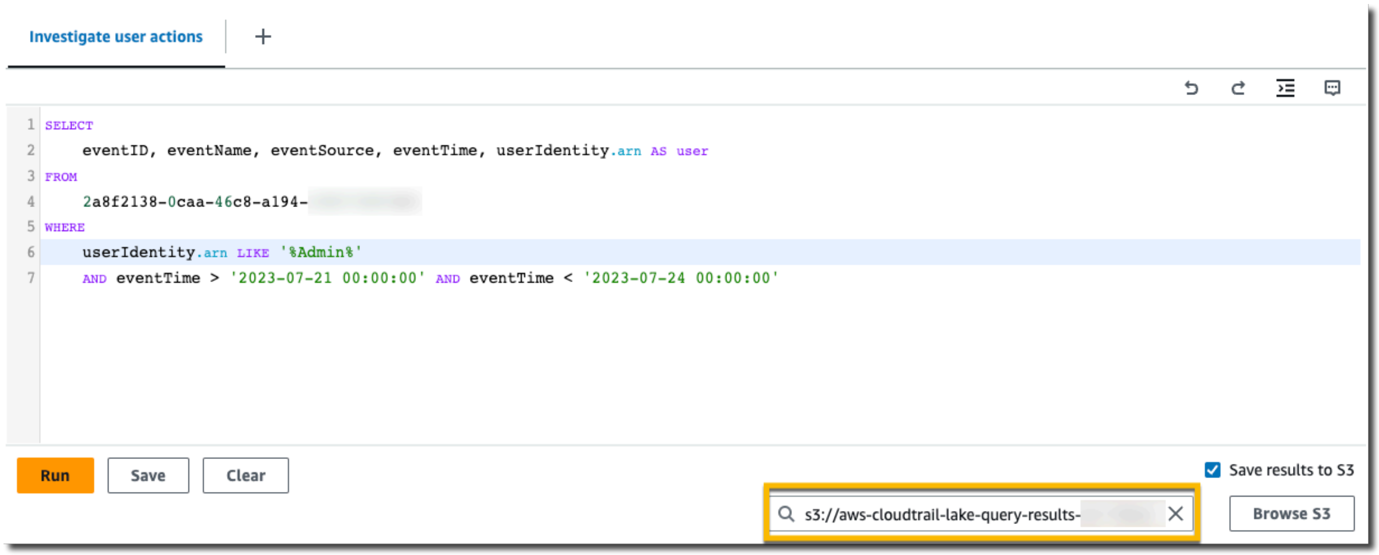
1. Sign in to the AWS Management Console and open the CloudTrail console at https://console.aws.amazon.com/cloudtrail/.
2. From the navigation pane, under **Lake**, choose **Query**.
3. On the **Sample queries** or **Saved queries** tabs, choose a query to run by choosing the **Query name**. In this example, we'll choose the sample query named **Investigate user actions**.
4. On the **Editor** tab, for **Event data store**, choose an event data store from the drop-down list. When you choose the event data store from the list, CloudTrail automatically populates the event data store ID in the `From` line.
5. In this sample query, we'll edit the `userIdentity.ARN` value to specify a user named `Admin`, and we'll leave the default values for `eventTime`. When you run a query, you're charged for the amount of data scanned. To help control costs, we recommend that you constrain queries by adding starting and ending `eventTime` time stamps to queries.



```
SELECT
    eventID, eventName, eventSource, eventTime, userIdentity.arn AS user
FROM
    2a8f2138-0caa-46c8-a194-[redacted]
WHERE
    userIdentity.arn LIKE '%Admin%'
    AND eventTime > '2023-07-21 00:00:00' AND eventTime < '2023-07-24 00:00:00'
```

6. Choose **Save results to S3** to save the query results to an S3 bucket. When you choose the default S3 bucket, CloudTrail creates and applies the required bucket policies. If you choose the default S3 bucket, your IAM policy needs to include permission for the `s3:PutEncryptionConfiguration` action because by default server-side encryption is enabled for the bucket. In this example, we'll use the default S3 bucket.

> **Note**
>
> To use a different bucket, specify a bucket name, or choose **Browse S3** to choose a bucket. The bucket policy must grant CloudTrail permission to deliver query results to the bucket. For information about manually editing the bucket policy, see [Amazon S3 bucket policy for CloudTrail Lake query results]().



```
SELECT
    eventID, eventName, eventSource, eventTime, userIdentity.arn AS user
FROM
    2a8f2138-0caa-46c8-a194-[redacted]
WHERE
    userIdentity.arn LIKE '%Admin%'
    AND eventTime > '2023-07-21 00:00:00' AND eventTime < '2023-07-24 00:00:00'
```

☑ Save results to S3

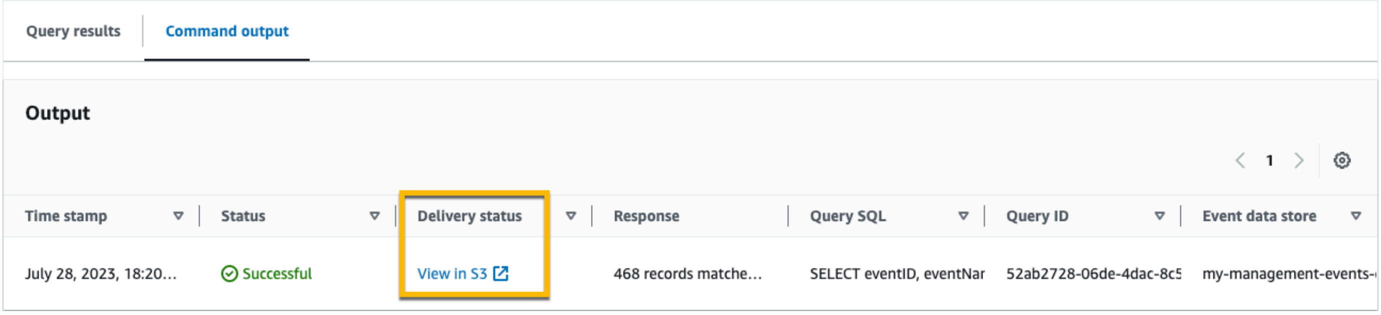
s3://aws-cloudtrail-lake-query-results-[redacted]

7. Choose **Run**. Depending on the size of your event data store, and the number of days of data it includes, a query can take several minutes to run. The **Command output** tab shows the status of a query, and whether a query is finished running. When a query has finished running, open the **Query results** tab to see a table of results for the active query (the query currently shown in the editor).

8. When CloudTrail completes delivery of the saved query results to your S3 bucket, the **Delivery status** column provides a link to the S3 bucket that contains your saved query result files as well as a [sign file]() that you can use to verify your saved query results. Choose **View in S3** to view the query result files and sign files in the S3 bucket.

> **Note**
>
> When you save query results, the query results may display in the CloudTrail console before they are viewable in the S3 bucket because CloudTrail delivers the query results after the query scan completes. While most queries complete within a few minutes, depending on the size of your event data store, it can take considerably longer for

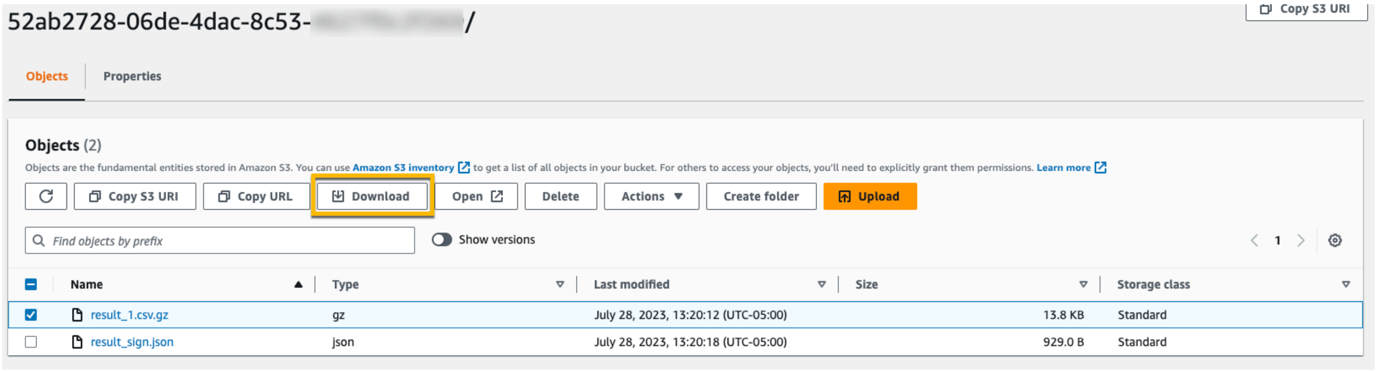
CloudTrail to deliver query results to your S3 bucket. CloudTrail delivers the query results to the S3 bucket in compressed gzip format. On average, after the query scan completes you can expect a latency of 60 to 90 seconds for every GB of data delivered to the S3 bucket.



Output

Time stamp	Status	Delivery status	Response	Query SQL	Query ID	Event data store
July 28, 2023, 18:20...	Successful	View in S3	468 records matche...	SELECT eventID, eventNar	52ab2728-06de-4dac-8c5	my-management-events-

9. To download your query results, choose the query result file (in this example, `result_1.csv.gz`) and then choose **Download**.


52ab2728-06de-4dac-8c53-[redacted]/

Objects (2)

Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

	Name	Type	Last modified	Size	Storage class
☑	result_1.csv.gz	gz	July 28, 2023, 13:20:12 (UTC-05:00)	13.8 KB	Standard
□	result_sign.json	json	July 28, 2023, 13:20:18 (UTC-05:00)	929.0 B	Standard

For information about validating saved query results, see Validate CloudTrail Lake saved query results.

View query results with the console

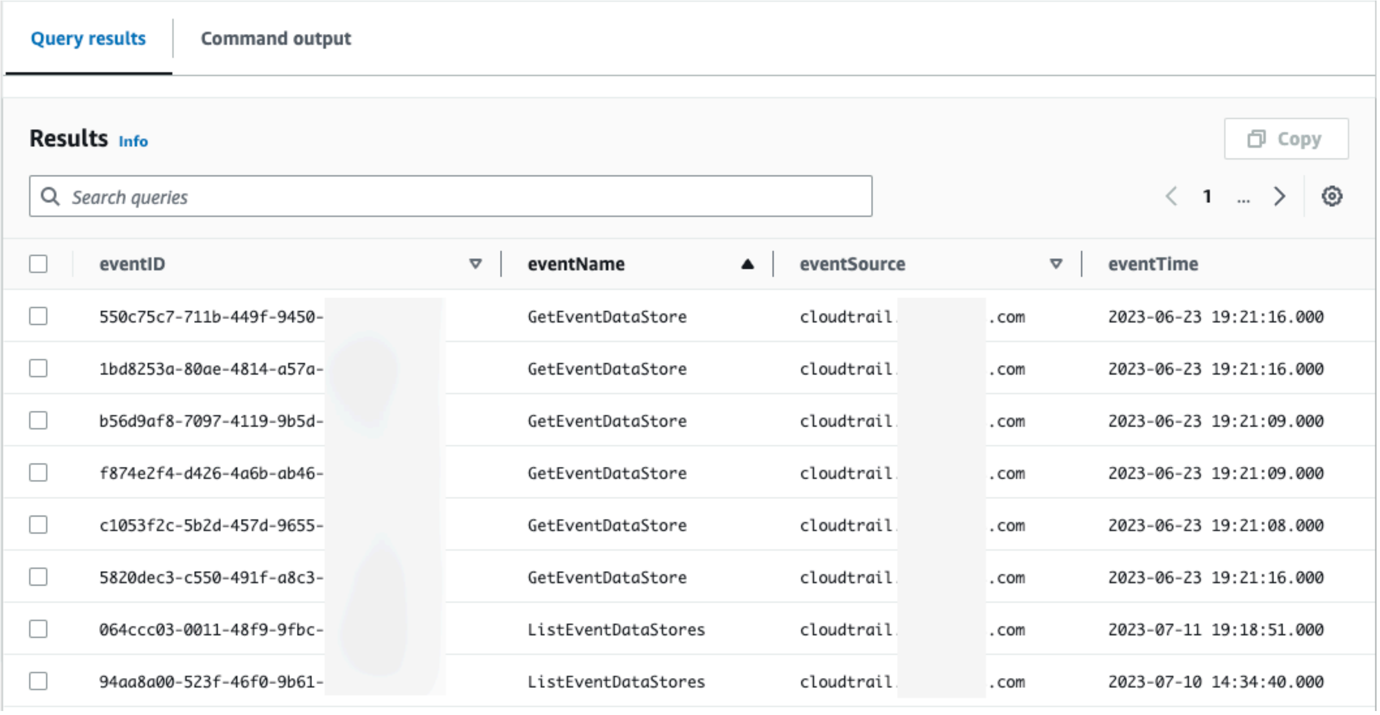
After your query finishes, you can view its results. The results of a query are available for seven days after the query finishes. You can view results for the active query on the **Query results** tab, or you can access results for all recent queries on the **Results history** tab on the **Lake** home page.

Query results can change from older runs of a query to newer ones, as later events in the query period can be logged between queries.

When you save query results, the query results may display in the CloudTrail console before they are viewable in the S3 bucket since CloudTrail delivers the query results after the query scan completes. While most queries complete within a few minutes, depending on the size of your event data store, it can take considerably longer for CloudTrail to deliver query results to your S3 bucket. CloudTrail delivers the query results to the S3 bucket in compressed gzip format.
On average, after the query scan completes you can expect a latency of 60 to 90 seconds for every GB of data delivered to the S3 bucket. For more information about finding and downloading saved query results, see Download saved query results.

> **Note**
>
> Queries that run for longer than one hour might time out. You can still get partial results that were processed before the query timed out. CloudTrail does not deliver partial query results to an S3 bucket. To avoid a time out, you can refine your query to limit the amount of data scanned by specifying a narrower time range.

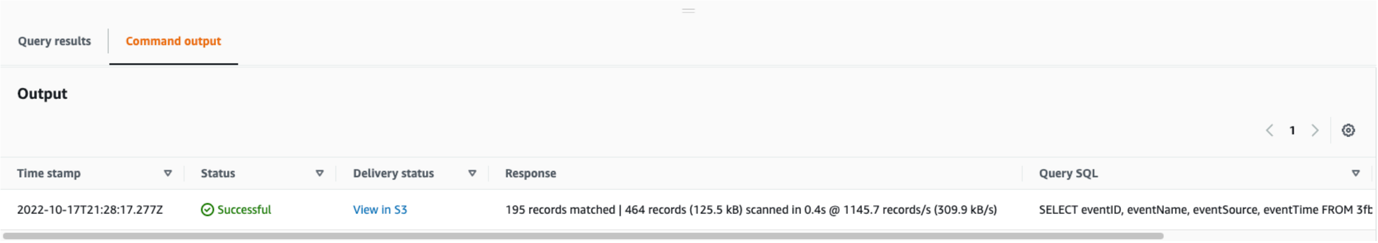
1. On the **Query results** tab for an active query, each row represents an event result that matched the query. Filter results by entering all or part of an event field value in the search bar. To copy an event, choose the event you want to copy and then choose **Copy**.



□	eventID	eventName	eventSource	eventTime
□	550c75c7-711b-449f-9450-[redacted]	GetEventDataStore	cloudtrail[redacted].com	2023-06-23 19:21:16.000
□	1bd8253a-80ae-4814-a57a-[redacted]	GetEventDataStore	cloudtrail[redacted].com	2023-06-23 19:21:16.000
□	b56d9af8-7097-4119-9b5d-[redacted]	GetEventDataStore	cloudtrail[redacted].com	2023-06-23 19:21:09.000
□	f874e2f4-d426-4a6b-ab46-[redacted]	GetEventDataStore	cloudtrail[redacted].com	2023-06-23 19:21:09.000
□	c1053f2c-5b2d-457d-9655-[redacted]	GetEventDataStore	cloudtrail[redacted].com	2023-06-23 19:21:08.000
□	5820dec3-c550-491f-a8c3-[redacted]	GetEventDataStore	cloudtrail[redacted].com	2023-06-23 19:21:16.000
□	064ccc03-0011-48f9-9fbc-[redacted]	ListEventDataStores	cloudtrail[redacted].com	2023-07-11 19:18:51.000
□	94aa8a00-523f-46f0-9b61-[redacted]	ListEventDataStores	cloudtrail[redacted].com	2023-07-10 14:34:40.000

2. On the **Command output** tab, view metadata about the query that was run, such as the event data store ID, run time, number of results scanned, and whether or not the query was

successful. If you saved the query results to an Amazon S3 bucket, the metadata also includes a link to the S3 bucket containing the saved query results.



Time stamp	Status	Delivery status	Response	Query SQL
2022-10-17T21:28:17.277Z	✓ Successful	View in S3	195 records matched | 464 records (125.5 kB) scanned in 0.4s @ 1145.7 records/s (309.9 kB/s)	SELECT eventID, eventName, eventSource, eventTime FROM 3ft

Download saved query results

After you save query results, you need to be able to locate the file containing the query results. CloudTrail delivers your query results to an Amazon S3 bucket that you specify when you save the query results.

> **Note**
>
> When you save query results, the query results may display in the console before they are viewable in the S3 bucket since CloudTrail delivers the query results after the query scan completes. While most queries complete within a few minutes, depending on the size of your event data store, it can take considerably longer for CloudTrail to deliver query results to your S3 bucket. CloudTrail delivers the query results to the S3 bucket in compressed gzip format. On average, after the query scan completes you can expect a latency of 60 to 90 seconds for every GB of data delivered to the S3 bucket.

Topics

- Find your CloudTrail Lake saved query results
- Download your CloudTrail Lake saved query results

Find your CloudTrail Lake saved query results

CloudTrail publishes query result and sign files to your S3 bucket. The query result file contains the output of the saved query and the sign file provides the signature and hash value for the query results. You can use the sign file to validate the query results. For more information about validating query results, see Validate CloudTrail Lake saved query results.

To retrieve a query result or sign file, you can use the Amazon S3 console, the Amazon S3 command line interface (CLI), or the API.

To find your query results and sign files with the Amazon S3 console

1. Open the Amazon S3 console.
2. Choose the bucket you specified.
3. Navigate through the object hierarchy until you find the query result and sign files. The query result file has a .csv.gz extension and the sign file has a .json extension.

You will navigate through an object hierarchy that is similar to the following example, but with a different bucket name, account ID, date, and query ID.

```
All Buckets
    DOC-EXAMPLE-BUCKET
        AWSLogs
            Account_ID;
                CloudTrail-Lake
                    Query
                        2022
                            06
                              20
                                Query_ID
```

Download your CloudTrail Lake saved query results

When you save query results, CloudTrail delivers two types of files to your Amazon S3 bucket.

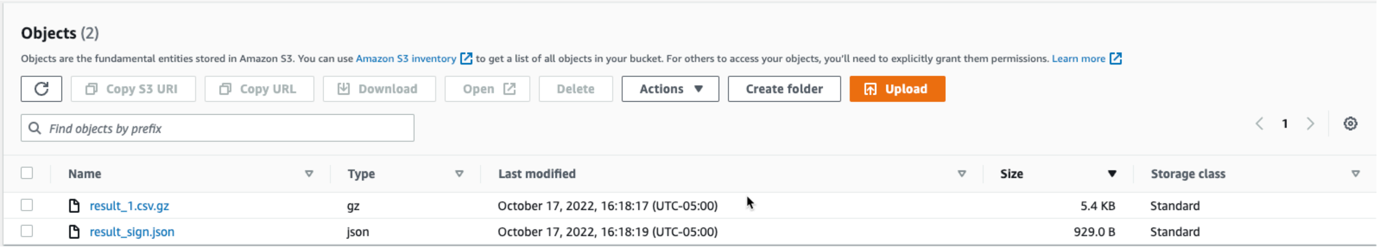
- A sign file in JSON format that you can use to validate the query result files. The sign file is named result_sign.json. For more information about the sign file, see CloudTrail sign file structure.
- One or more query result files in CSV format, which contain the results from the query. The number of query result files delivered is dependent upon the total size of the query results. The maximum file size for a query result file is 1 TB. Each query result file is named result_*number*.csv.gz. For example, if the total size of the query results was 2 TB, you would have two query result files, result_1.csv.gz and result_2.csv.gz.

CloudTrail query result and sign files are Amazon S3 objects. You can use the S3 console, the AWS Command Line Interface (CLI), or the S3 API to retrieve query result and sign files.

The following procedure describes how to download the query result and sign files with the Amazon S3 console.

To download your query result or sign file with the Amazon S3 console

1. Open the Amazon S3 console.
2. Choose the bucket and choose the file that you want to download.



Objects (2)

Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

□	Name	Type	Last modified	Size	Storage class
□	result_1.csv.gz	gz	October 17, 2022, 16:18:17 (UTC-05:00)	5.4 KB	Standard
□	result_sign.json	json	October 17, 2022, 16:18:19 (UTC-05:00)	929.0 B	Standard

3. Choose **Download** and follow any prompts to save the file.

> **Note**
>
> Some browsers, such as Chrome, automatically extract the query result file for you. If your browser does this for you, skip to step 5.

4. Use a product such as 7-Zip to extract the query result file.
5. Open the query result or sign file.

Validate CloudTrail Lake saved query results

To determine whether the query results were modified, deleted, or unchanged after CloudTrail delivered the query results, you can use CloudTrail query results integrity validation. This feature is built using industry standard algorithms: SHA-256 for hashing and SHA-256 with RSA for digital signing. This makes it computationally infeasible to modify, delete or forge CloudTrail query result files without detection. You can use the command line to validate the query result files.

Why use it?

Validated query result files are invaluable in security and forensic investigations. For example, a validated query result file enables you to assert positively that the query result file itself has not changed. The CloudTrail query result file integrity validation process also lets you know if a query result file has been deleted or changed.

Topics

Validate saved query results with the AWS CLI

You can validate the integrity of the query result files and sign file by using the **aws cloudtrail verify-query-results** command.

Prerequisites

To validate query results integrity with the command line, the following conditions must be met:

- You must have online connectivity to AWS.
- You must use AWS CLI version 2.
- To validate query result files and sign file locally, the following conditions apply:
  - You must put the query result files and sign file in the specified file path. Specify the file path as the value for the **--local-export-path** parameter.
  - You must not rename the query result files and sign file.
- To validate the query result files and sign file in the S3 bucket, the following conditions apply:
  - You must not rename the query result files and sign file.
  - You must have read access to the Amazon S3 bucket that contains the query result files and sign file.
  - The specified S3 prefix must contain the query result files and sign file. Specify the S3 prefix as the value for the **--s3-prefix** parameter.

verify-query-results

The **verify-query-results** command verifies the hash value of each query result file by comparing the value with the `fileHashValue` in the sign file, and then validating the `hashSignature` in the sign file.

When you verify query results, you can use either the **--s3-bucket** and **--s3-prefix** command line options to validate the query result files and sign file stored in an S3 bucket, or you can use the **--local-export-path** command line option to perform a local validation of the downloaded query result files and sign file.

> **Note**
>
> The **verify-query-results** command is Region specific. You must specify the **--region** global option to validate query results for a specific AWS Region.

The following are the options for the **verify-query-results** command.

--s3-bucket *<string>*

Specifies the S3 bucket name that stores the query result files and sign file. You cannot use this parameter with **--local-export-path**.

--s3-prefix *<string>*

Specifies the S3 path of the S3 folder that contains the query result files and sign file (for example, `s3/path/`). You cannot use this parameter with **--local-export-path**. You do not need to provide this parameter if the files are located in the root directory of the S3 bucket.

--local-export-path *<string>*

Specifies the local directory that contains the query result files and sign file (for example, `/local/path/to/export/file/`). You cannot use this parameter with **--s3-bucket** or **--s3-prefix**.

Examples

The following example validates query results using the **--s3-bucket** and **--s3-prefix** command line options to specify the S3 bucket name and prefix containing the query result files and sign file.

```
aws cloudtrail verify-query-results --s3-bucket DOC-EXAMPLE-BUCKET --s3-prefix prefix
 --region region
```

The following example validates downloaded query results using the **--local-export-path** command line option to specify the local path for the query result files and sign file. For more information about downloading query result files, see [Download your CloudTrail Lake saved query results](#).

```
aws cloudtrail verify-query-results --local-export-path local_file_path --region region
```

Validation results

The following table describes the possible validation messages for query result files and sign file.

File Type	Validation Message	Description
`Sign file`	`Successfully validated sign and query result files`	The sign file signature is valid. The query result files it references can be checked.
`Query result file`	`ValidationError: "File` *`file_name`* `has inconsistent hash value with hash value recorded in sign file, hash value in sign file is` *`expected_hash`* `, but get` *`computed_hash`*	Validation failed because the hash value for the query result file did not match the `fileHashValue` in the sign file.
`Sign file`	`ValidationError: Invalid signature in sign file`	Validation for the sign file failed because the signature is not valid.

CloudTrail sign file structure

The sign file contains the name of each query result file that was delivered to your Amazon S3 bucket when you saved the query results, the hash value for each query result file, and the digital signature of the file. The digital signature and hash values are used for validating the integrity of the query result files and of the sign file itself.

Sign file location

The sign file is delivered to an Amazon S3 bucket location that follows this syntax.

```
s3://DOC-EXAMPLE-BUCKET/optional-prefix/AWSLogs/aws-account-ID/CloudTrail-Lake/
Query/year/month/date/query-ID/result_sign.json
```

Sample sign file contents

The following example sign file contains information for CloudTrail Lake query results.

```
{
  "version": "1.0",
  "region": "us-east-1",
  "files": [
    {
      "fileHashValue" :
 "de85a48b8a363033c891abd723181243620a3af3b6505f0a44db77e147e9c188",
      "fileName" : "result_1.csv.gz"
    }
  ],
  "hashAlgorithm" : "SHA-256",
  "signatureAlgorithm" : "SHA256withRSA",
  "queryCompleteTime": "2022-05-10T22:06:30Z",
  "hashSignature" :
 "7664652aaf1d5a17a12ba50abe6aca77c0ec76264bdf7dce71ac6d1c7781117c2a412e5820bccf473b1361306dff6
  "publicKeyFingerprint" : "67b9fa73676d86966b449dd677850753"
}
```

Sign file field descriptions

The following are descriptions for each field in the sign file:

`version`

The version of the sign file.

`region`

The Region for the AWS account used for saving the query results.

`files.fileHashValue`

The hexadecimal encoded hash value of the compressed query result file content.

`files.fileName`

The name of the query result file.

`hashAlgorithm`

The hash algorithm used to hash the query result file.

`signatureAlgorithm`

The algorithm used to sign the file.

`queryCompleteTime`

Indicates when CloudTrail delivered the query results to the S3 bucket. You can use this value to find the public key.

`hashSignature`

The hash signature for the file.

`publicKeyFingerprint`

The hexadecimal encoded fingerprint of the public key used to sign the file.

Custom implementations of CloudTrail query result file integrity validation

Because CloudTrail uses industry standard, openly available cryptographic algorithms and hash functions, you can create your own tools to validate the integrity of the CloudTrail query result files. When you save query results to an Amazon S3 bucket, CloudTrail delivers a sign file to your S3 bucket. You can implement your own validation solution to validate the signature and query result files. For more information about the sign file, see CloudTrail sign file structure.

This topic describes how the sign file is signed, and then details the steps that you will need to take to implement a solution that validates the sign file and the query result files that the sign file references.

Understanding how CloudTrail sign files are signed

CloudTrail sign files are signed with RSA digital signatures. For each sign file, CloudTrail does the following:

1. Creates a hash list containing the hash value for each query result file.
2. Gets a private key unique to the Region.
3. Passes the SHA-256 hash of the string and the private key to the RSA signing algorithm, which produces a digital signature.
4. Encodes the byte code of the signature into hexadecimal format.
5. Puts the digital signature into the sign file.

Contents of the data signing string

The data signing string consists of the hash value for each query result file separated by a space. The sign file lists the `fileHashValue` for each query result file.

Custom validation implementation steps

When implementing a custom validation solution, you will need to validate the sign file and the query result files that it references.

Validate the sign file

To validate a sign file, you need its signature, the public key whose private key was used to sign it, and a data signing string that you compute.

1. Get the sign file.
2. Verify that the sign file has been retrieved from its original location.
3. Get the hexadecimal-encoded signature of the sign file.
4. Get the hexadecimal-encoded fingerprint of the public key whose private key was used to sign the sign file.

5. Retrieve the public key for the time range corresponding to `queryCompleteTime` in the sign file. For the time range, choose a `StartTime` earlier than the `queryCompleteTime` and an `EndTime` later than the `queryCompleteTime`.
6. From among the public keys retrieved, choose the public key whose fingerprint matches the `publicKeyFingerprint` value in the sign file.
7. Using a hash list containing the hash value for each query result file separated by a space, recreate the data signing string used to verify the sign file signature. The sign file lists the `fileHashValue` for each query result file.

   For example, if your sign file's `files` array contains the following three query result files, your hash list is "aaa bbb ccc".

   ```
   “files": [

      {

           "fileHashValue" : “aaa”,

           "fileName" : "result_1.csv.gz"

      },
      {

           "fileHashValue" : “bbb”,

           "fileName" : "result_2.csv.gz"

      },
      {

           "fileHashValue" : “ccc”,

           "fileName" : "result_3.csv.gz"

      }
   ],
   ```
8. Validate the signature by passing in the SHA-256 hash of the string, the public key, and the signature as parameters to the RSA signature verification algorithm. If the result is true, the sign file is valid.

Validate the query result files

If the sign file is valid, validate the query result files that the sign file references. To validate the integrity of a query result file, compute its SHA-256 hash value on its compressed content and compare the results with the `fileHashValue` for the query result file recorded in the sign file. If the hashes match, the query result file is valid.

The following sections describe the validation process in detail.

A. Get the sign file

The first steps are to get the sign file and get the fingerprint of the public key.

1. Get the sign file from your Amazon S3 bucket for the query results that you want to validate.
2. Next, get the `hashSignature` value from the sign file.
3. In the sign file, get the fingerprint of the public key whose private key was used to sign the file from the `publicKeyFingerprint` field.

B. Retrieve the public key for validating the sign file

To get the public key to validate the sign file, you can use either the AWS CLI or the CloudTrail API. In both cases, you specify a time range (that is, a start time and end time) for the sign file that you want to validate. Use a time range corresponding to the `queryCompleteTime` in the sign file. One or more public keys may be returned for the time range that you specify. The returned keys may have validity time ranges that overlap.

> **Note**
>
> Because CloudTrail uses different private/public key pairs per Region, each sign file is signed with a private key unique to its Region. Therefore, when you validate a sign file from a particular Region, you must retrieve its public key from the same Region.

Use the AWS CLI to retrieve public keys

To retrieve a public key for a sign file by using the AWS CLI, use the `cloudtrail list-public-keys` command. The command has the following format:

```
aws cloudtrail list-public-keys [--start-time <start-time>] [--end-time
<end-time>]
```

The start-time and end-time parameters are UTC timestamps and are optional. If not specified, the current time is used, and the currently active public key or keys are returned.

Sample Response

The response will be a list of JSON objects representing the key (or keys) returned:

Use the CloudTrail API to retrieve public keys

To retrieve a public key for a sign file by using the CloudTrail API, pass in start time and end time values to the `ListPublicKeys` API. The `ListPublicKeys` API returns the public keys whose private keys were used to sign the file within the specified time range. For each public key, the API also returns the corresponding fingerprint.

`ListPublicKeys`

This section describes the request parameters and response elements for the `ListPublicKeys` API.

> **Note**
>
> The encoding for the binary fields for `ListPublicKeys` is subject to change.

Request Parameters

Name	Description
`StartTime`	Optionally specifies, in UTC, the start of the time range to look up the public key for CloudTrail sign file. If StartTime is not specified, the current time is used, and the current public key is returned. Type: DateTime
`EndTime`	Optionally specifies, in UTC, the end of the time range to look up public keys for CloudTrail sign files. If EndTime is not specified, the current time is used. Type: DateTime

Response Elements

PublicKeyList, an array of PublicKey objects that contains:

Name	Description
Value	The DER encoded public key value in PKCS #1 format. Type: Blob
ValidityStartTime	The starting time of validity of the public key. Type: DateTime
ValidityEndTime	The ending time of validity of the public key. Type: DateTime
Fingerprint	The fingerprint of the public key. The fingerprint can be used to identify the public key that you must use to validate the sign file. Type: String

C. Choose the public key to use for validation

From among the public keys retrieved by `list-public-keys` or `ListPublicKeys`, choose the public key whose fingerprint matches the fingerprint recorded in the `publicKeyFingerprint` field of the sign file. This is the public key that you will use to validate the sign file.

D. Recreate the data signing string

Now that you have the signature of the sign file and the associated public key, you need to calculate the data signing string. After you have calculated the data signing string, you will have the inputs needed to verify the signature.

The data signing string consists of the hash value for each query result file separated by a space. After you recreate this string, you can validate the sign file.

E. Validate the sign file

Pass the recreated data signing string, digital signature, and public key to the RSA signature verification algorithm. If the output is true, the signature of the sign file is verified and the sign file is valid.

F. Validate the query result files

After you have validated the sign file, you can validate the query result files it references. The sign file contains the SHA-256 hashes of the query result files. If one of the query result files was modified after CloudTrail delivered it, the SHA-256 hashes will change, and the signature of the sign file will not match.

Use the following procedure to validate the query result files listed in the sign file's `files` array.

1. Retrieve the original hash of the file from the `files.fileHashValue` field in the sign file.
2. Hash the compressed contents of the query result file with the hashing algorithm specified in `hashAlgorithm`.
3. Compare the hash value that you generated for each query result file with the `files.fileHashValue` in the sign file. If the hashes match, the query result files are valid.

Validating signature and query result files offline

When validating sign and query result files offline, you can generally follow the procedures described in the previous sections. However, you must take into account the following information about public keys.

Public keys

In order to validate offline, the public key that you need for validating query result files in a given time range must first be obtained online (by calling `ListPublicKeys`, for example) and then stored offline. This step must be repeated whenever you want to validate additional files outside the initial time range that you specified.

Sample validation snippet

The following sample snippet provides skeleton code for validating CloudTrail sign and query result files. The skeleton code is online/offline agnostic; that is, it is up to you to decide whether to implement it with or without online connectivity to AWS. The suggested implementation uses the Java Cryptography Extension (JCE) and Bouncy Castle as a security provider.

The sample snippet shows:

- How to create the data signing string used to validate the sign file signature.
- How to verify the sign file's signature.
- How to calculate the hash value for the query result file and compare it with the `fileHashValue` listed in the sign file to verify the authenticity of the query result file.

```
import org.apache.commons.codec.binary.Hex;
import org.bouncycastle.asn1.pkcs.PKCSObjectIdentifiers;
import org.bouncycastle.asn1.pkcs.RSAPublicKey;
import org.bouncycastle.asn1.x509.AlgorithmIdentifier;
import org.bouncycastle.asn1.x509.SubjectPublicKeyInfo;
import org.bouncycastle.jce.provider.BouncyCastleProvider;
import org.json.JSONArray;
import org.json.JSONObject;

import java.security.KeyFactory;
import java.security.MessageDigest;
import java.security.PublicKey;
import java.security.Security;
import java.security.Signature;
import java.security.spec.X509EncodedKeySpec;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.List;
import java.util.stream.Collectors;


public class SignFileValidationSampleCode {

    public void validateSignFile(String s3Bucket, String s3PrefixPath) throws Exception
 {
        MessageDigest messageDigest = MessageDigest.getInstance("SHA-256");

        // Load the sign file from S3 (using Amazon S3 Client) or from your local copy
        JSONObject signFile = loadSignFileToMemory(s3Bucket, String.format("%s/%s",
 s3PrefixPath, "result_sign.json"));

        // Using the Bouncy Castle provider as a JCE security provider - http://
www.bouncycastle.org/
        Security.addProvider(new BouncyCastleProvider());
        List<String> hashList = new ArrayList<>();

        JSONArray jsonArray = signFile.getJSONArray("files");

        for (int i = 0; i < jsonArray.length(); i++) {
            JSONObject file = jsonArray.getJSONObject(i);
            String fileS3ObjectKey = String.format("%s/%s", s3PrefixPath,
file.getString("fileName"));

            // Load the export file from S3 (using Amazon S3 Client) or from your local
copy
            byte[] exportFileContent = loadCompressedExportFileInMemory(s3Bucket,
fileS3ObjectKey);
            messageDigest.update(exportFileContent);
            byte[] exportFileHash = messageDigest.digest();
            messageDigest.reset();
            byte[] expectedHash = Hex.decodeHex(file.getString("fileHashValue"));

            boolean signaturesMatch = Arrays.equals(expectedHash, exportFileHash);
            if (!signaturesMatch) {
                System.err.println(String.format("Export file: %s/%s hash doesn't
match.\tExpected: %s Actual: %s",
                        s3Bucket, fileS3ObjectKey,
                        Hex.encodeHexString(expectedHash),
Hex.encodeHexString(exportFileHash)));
            } else {
                System.out.println(String.format("Export file: %s/%s hash match",
                        s3Bucket, fileS3ObjectKey));
            }

            hashList.add(file.getString("fileHashValue"));
        }
        String hashListString = hashList.stream().collect(Collectors.joining(" "));

        /*
            NOTE:
            To find the right public key to verify the signature, call CloudTrail
ListPublicKey API to get a list
            of public keys, then match by the publicKeyFingerprint in the sign file.
Also, the public key bytes
            returned from ListPublicKey API are DER encoded in PKCS#1 format:

            PublicKeyInfo ::= SEQUENCE {
                algorithm       AlgorithmIdentifier,
                PublicKey       BIT STRING
            }

            AlgorithmIdentifier ::= SEQUENCE {
                algorithm       OBJECT IDENTIFIER,
                parameters      ANY DEFINED BY algorithm OPTIONAL
            }
        */
        byte[] pkcs1PublicKeyBytes =
 getPublicKey(signFile.getString("queryCompleteTime"),
                signFile.getString("publicKeyFingerprint"));
        byte[] signatureContent = Hex.decodeHex(signFile.getString("hashSignature"));

        // Transform the PKCS#1 formatted public key to x.509 format.
        RSAPublicKey rsaPublicKey = RSAPublicKey.getInstance(pkcs1PublicKeyBytes);
        AlgorithmIdentifier rsaEncryption = new
 AlgorithmIdentifier(PKCSObjectIdentifiers.rsaEncryption, null);
        SubjectPublicKeyInfo publicKeyInfo = new SubjectPublicKeyInfo(rsaEncryption,
 rsaPublicKey);

        // Create the PublicKey object needed for the signature validation
        PublicKey publicKey = KeyFactory.getInstance("RSA", "BC")
                .generatePublic(new X509EncodedKeySpec(publicKeyInfo.getEncoded()));

        // Verify signature
        Signature signature = Signature.getInstance("SHA256withRSA", "BC");
        signature.initVerify(publicKey);
        signature.update(hashListString.getBytes("UTF-8"));

        if (signature.verify(signatureContent)) {
            System.out.println("Sign file signature is valid.");
        } else {
            System.err.println("Sign file signature failed validation.");
        }

        System.out.println("Sign file validation completed.");
    }
}
```

Run and manage CloudTrail Lake queries with the AWS CLI

You can use the AWS CLI to run and manage your CloudTrail Lake queries. When using the AWS CLI, remember that your commands run in the AWS Region configured for your profile. If you want to run the commands in a different Region, either change the default Region for your profile, or use the **--region** parameter with the command.

Available commands for CloudTrail Lake queries

Commands for running and managing queries in CloudTrail Lake include:

- `start-query` to run a query.
- `describe-query` to return metadata about a query.
- `get-query-results` to return query results for the specified query ID.
- `list-queries` to get a list queries for the specified event data store.
- `cancel-query` to cancel a running query.

For a list of available commands for CloudTrail Lake event data stores, see Available commands for event data stores.

For a list of available commands for CloudTrail Lake integrations, see Available commands for CloudTrail Lake integrations.

Start a query with the AWS CLI

The following example AWS CLI **start-query** command runs a query on the event data store specified as an ID in the query statement and delivers the query results to a specified S3 bucket. The `--query-statement` parameter provides a SQL query, enclosed in single quotation marks. Optional parameters include `--delivery-s3uri`, to deliver the query results to a specified S3 bucket. For more information about the query language you can use in CloudTrail Lake, see CloudTrail Lake SQL constraints.

```
aws cloudtrail start-query
--query-statement 'SELECT eventID, eventTime FROM EXAMPLE-f852-4e8f-8bd1-bcf6cEXAMPLE
 LIMIT 10'
--delivery-s3uri "s3://aws-cloudtrail-lake-query-results-123456789012-us-east-1"
```

The response is a QueryId string. To get the status of a query, run **describe-query** using the QueryId value returned by **start-query**. If the query is successful, you can run **get-query-results** to get results.

Output

```
{
    "QueryId": "EXAMPLE2-0add-4207-8135-2d8a4EXAMPLE"
}
```

> **Note**
>
> Queries that run for longer than one hour might time out. You can still get partial results that were processed before the query timed out.
>
> If you are delivering the query results to an S3 bucket using the optional --delivery-s3uri parameter, the bucket policy must grant CloudTrail permission to delivery query results to the bucket. For information about manually editing the bucket policy, see Amazon S3 bucket policy for CloudTrail Lake query results.

Get metadata about a query with the AWS CLI

The following example AWS CLI **describe-query** command gets metadata about a query, including query run time in milliseconds, number of events scanned and matched, total number of bytes scanned, and query status. The BytesScanned value matches the number of bytes for which your account is billed for the query, unless the query is still running. If the query results were delivered to an S3 bucket, the response also provides the S3 URI and the delivery status.

You must specify a value for either the --query-id or the --query-alias parameter. Specifying the --query-alias parameter returns information about the last query run for the alias.

```
aws cloudtrail describe-query --query-id EXAMPLEd-17a7-47c3-a9a1-eccf7EXAMPLE
```

The following is an example response.

```
{
    "QueryId": "EXAMPLE2-0add-4207-8135-2d8a4EXAMPLE",
    "QueryString": "SELECT eventID, eventTime FROM EXAMPLE-f852-4e8f-8bd1-bcf6cEXAMPLE
 LIMIT 10",
    "QueryStatus": "RUNNING",
    "QueryStatistics": {
        "EventsMatched": 10,
        "EventsScanned": 1000,
        "BytesScanned": 35059,
        "ExecutionTimeInMillis": 3821,
        "CreationTime": "1598911142"
    }
}
```

Get query results with the AWS CLI

The following example AWS CLI **get-query-results** command gets event data results of a query. You must specify the `--query-id` returned by the **start-query** command. The `BytesScanned` value matches the number of bytes for which your account is billed for the query, unless the query is still running. Optional parameters include `--max-query-results`, to specify a maximum number of results that you want the command to return on a single page. If there are more results than your specified `--max-query-results` value, run the command again adding the returned `NextToken` value to get the next page of results.

```
aws cloudtrail get-query-results
--query-id EXAMPLEd-17a7-47c3-a9a1-eccf7EXAMPLE
```

Output

```
{
    "QueryStatus": "RUNNING",
    "QueryStatistics": {
        "ResultsCount": 244,
        "TotalResultsCount": 1582,
        "BytesScanned":27044
    },
    "QueryResults": [
      {
        "key": "eventName",
        "value": "StartQuery",
      }
   ],
    "QueryId": "EXAMPLE2-0add-4207-8135-2d8a4EXAMPLE",
    "QueryString": "SELECT eventID, eventTime FROM EXAMPLE-f852-4e8f-8bd1-bcf6cEXAMPLE
 LIMIT 10",
    "NextToken": "20add42078135EXAMPLE"
}
```

List all queries on an event data store with the AWS CLI

The following example AWS CLI **list-queries** command returns a list of queries and query statuses on a specified event data store for the past seven days. You must specify an ARN or the ID suffix of an ARN value for `--event-data-store`. Optionally, to shorten the list of results, you can specify a time range, formatted as timestamps, by adding `--start-time` and `--end-time` parameters, and a `--query-status` value. Valid values for `QueryStatus` include `QUEUED`, `RUNNING`, `FINISHED`, `FAILED`, or `CANCELLED`.

list-queries also has optional pagination parameters. Use `--max-results` to specify a maximum number of results that you want the command to return on a single page. If there are more results than your specified `--max-results` value, run the command again adding the returned `NextToken` value to get the next page of results.

```
aws cloudtrail list-queries
--event-data-store EXAMPLE-f852-4e8f-8bd1-bcf6cEXAMPLE
--query-status CANCELLED
--start-time 1598384589
--end-time 1598384602
--max-results 10
```

Output

```
{
    "Queries": [
        {
          "QueryId": "EXAMPLE2-0add-4207-8135-2d8a4EXAMPLE",
          "QueryStatus": "CANCELLED",
          "CreationTime": 1598911142
        },
        {
          "QueryId": "EXAMPLE2-4e89-9230-2127-5dr3aEXAMPLE",
          "QueryStatus": "CANCELLED",
          "CreationTime": 1598296624
        }
     ],
    "NextToken": "20add42078135EXAMPLE"
}
```

Cancel a running query with the AWS CLI

The following example AWS CLI **cancel-query** command cancels a query with a status of RUNNING. You must specify a value for `--query-id`. When you run **cancel-query**, the query status might show as CANCELLED even if the **cancel-query** operation is not yet finished.

> **Note**
>
> A canceled query can incur charges. Your account is still charged for the amount of data that was scanned before you canceled the query.

The following is a CLI example.

```
aws cloudtrail cancel-query
--query-id EXAMPLEd-17a7-47c3-a9a1-eccf7EXAMPLE
```

Output

```
QueryId -> (string)
QueryStatus -> (string)
```

CloudTrail Lake SQL constraints

CloudTrail Lake queries are SQL strings. This section provides information about the supported functions, operators, and schemas.

Only SELECT statements are allowed. No query strings can change or mutate data.

The CloudTrail Lake syntax for a SELECT statement is as follows. The event data store ID—the ID portion of the event data store's ARN—is specified for the FROM value.

```
SELECT [ DISTINCT ] columns [ Aggregate ]
[ FROM table event_data_store_ID]
[ WHERE columns [ Conditions ] ]
[ GROUP BY columns [ DISTINCT | Aggregate ] ]
[ HAVING columns [ Aggregate | Conditions ] ]
[ ORDER BY columns [ Aggregate | ASC | DESC | NULLS | FIRST | LAST ]
[ LIMIT [ INT ] ]
```

CloudTrail Lake supports all valid Presto SQL SELECT statements, functions, and operators. For more information about the supported SQL functions and operators, see Functions and Operators on the Presto documentation website.

The CloudTrail console provides a number of sample queries that can help you get started writing your own queries. For more information, see View sample queries with the CloudTrail console.

Topics

- Supported functions, condition and join operators
- Advanced, multi-table query support

Supported functions, condition and join operators

Supported functions

CloudTrail Lake supports all Presto functions. For more information about the supported functions, see Functions and Operators on the Presto documentation website.

CloudTrail Lake does not support the INTERVAL keyword.

Supported condition operators

The following are supported condition operators.

```
AND
OR
IN
NOT
IS (NOT) NULL
LIKE
BETWEEN
GREATEST
LEAST
IS DISTINCT FROM
IS NOT DISTINCT FROM
<
>
<=
>=
<>
!=
( conditions ) #parenthesised conditions
```

Supported join operators

The following are the supported `JOIN` operators. For more information about running multi-table queries, see Advanced, multi-table query support.

```
UNION
UNION ALL
EXCEPT
INTERSECT
LEFT JOIN
RIGHT JOIN
INNER JOIN
```

Advanced, multi-table query support

CloudTrail Lake supports advanced query language across multiple event data stores.

- `UNION|UNION ALL|EXCEPT|INTERSECT`
- `LEFT|RIGHT|INNER JOIN`

To run your query, use the **start-query** command in the AWS CLI. The following is an example, using one of the sample queries in this section.

```
aws cloudtrail start-query
--query-statement "Select eventId, eventName from EXAMPLEf852-4e8f-8bd1-bcf6cEXAMPLE
 UNION Select eventId, eventName from EXAMPLEg741-6y1x-9p3v-bnh6iEXAMPLE UNION ALL
 Select eventId, eventName from EXAMPLEb529-4e8f9l3d-6m2z-lkp5sEXAMPLE ORDER BY eventId
 LIMIT 10;"
```

The response is a `QueryId` string. To get the status of a query, run `describe-query`, using the `QueryId` value returned by `start-query`. If the query is successful, you can run `get-query-results` to get results.

`UNION|UNION ALL|EXCEPT|INTERSECT`

The following is an example query that uses `UNION` and `UNION ALL` to find events by their event ID and event name in three event data stores, EDS1, EDS2, and EDS3. The results are selected from each event data store first, then results are concatenated, ordered by event ID, and limited to ten events.

```
Select eventId, eventName from EDS1
UNION
Select eventId, eventName from EDS2
UNION ALL
Select eventId, eventName from EDS3
ORDER BY eventId LIMIT 10;
```

LEFT|RIGHT|INNER JOIN

The following is an example query that uses LEFT JOIN to find all events from an event data store named eds2, mapped to edsB, that match those in a primary (left) event data store, edsA. The returned events occur on or before January 1, 2020, and only the event names are returned.

```
SELECT edsA.eventName, edsB.eventName, element_at(edsA.map, 'test')
FROM eds1 as edsA
LEFT JOIN eds2 as edsB
ON edsA.eventId = edsB.eventId
WHERE edsA.eventtime <= '2020-01-01'
ORDER BY edsB.eventName;
```

Supported SQL schemas for event data stores

The following sections provide the supported SQL schema for each event data store type.

Topics

Supported schema for CloudTrail event record fields

The following is the valid SQL schema for CloudTrail management and data event record fields. For more information about CloudTrail event record fields, see CloudTrail record contents.

```
[
    {
        "Name": "eventversion",
        "Type": "string"
    },
    {
        "Name": "useridentity",
        "Type":
 "struct<type:string,principalid:string,arn:string,accountid:string,accesskeyid:string,

 username:string,sessioncontext:struct<attributes:struct<creationdate:timestamp,

 mfaauthenticated:string>,sessionissuer:struct<type:string,principalid:string,arn:string,

 accountid:string,username:string>,webidfederationdata:struct<federatedprovider:string,

 attributes:map<string,string>>,sourceidentity:string,ec2roledelivery:string,
                 ec2issuedinvpc:string>,invokedby:string,identityprovider:string>"
    },
    {
        "Name": "eventtime",
        "Type": "timestamp"
    },
    {
        "Name": "eventsource",
        "Type": "string"
    },
    {
        "Name": "eventname",
        "Type": "string"
    },
    {
        "Name": "awsregion",
        "Type": "string"
    },
    {
        "Name": "sourceipaddress",
        "Type": "string"
    },
    {
        "Name": "useragent",
        "Type": "string"
    },
    {
        "Name": "errorcode",
        "Type": "string"
    },
    {
        "Name": "errormessage",
        "Type": "string"
    },
    {
        "Name": "requestparameters",
        "Type": "map<string,string>"
    },
    {
        "Name": "responseelements",
        "Type": "map<string,string>"
    },
    {
        "Name": "additionaleventdata",
        "Type": "map<string,string>"
    },
    {
        "Name": "requestid",
        "Type": "string"
    },
    {
        "Name": "eventid",
        "Type": "string"
    },
    {
        "Name": "readonly",
        "Type": "boolean"
    },
    {
        "Name": "resources",
        "Type":
 "array<struct<accountid:string,type:string,arn:string,arnprefix:string>>"
    },
    {
        "Name": "eventtype",
        "Type": "string"
    },
    {
        "Name": "apiversion",
        "Type": "string"
    },
    {
        "Name": "managementevent",
        "Type": "boolean"
    },
    {
        "Name": "recipientaccountid",
        "Type": "string"
    },
    {
        "Name": "sharedeventid",
        "Type": "string"
    },
    {
        "Name": "annotation",
        "Type": "string"
    },
    {
        "Name": "vpcendpointid",
        "Type": "string"
    },
    {
        "Name": "serviceeventdetails",
        "Type": "map<string,string>"
    },
    {
        "Name": "addendum",
        "Type": "map<string,string>"
    },
    {
        "Name": "edgedevicedetails",
        "Type": "map<string,string>"
    },
    {
        "Name": "insightdetails",
        "Type": "map<string,string>"
    },
    {
        "Name": "eventcategory",
        "Type": "string"
    },
    {
        "Name": "tlsdetails",
        "Type":
 "struct<tlsversion:string,ciphersuite:string,clientprovidedhostheader:string>"
    },
    {
        "Name": "sessioncredentialfromconsole",
        "Type": "string"
    },
    {
        "Name": "eventjson",
        "Type": "string"
    }
    {
        "Name": "eventjsonchecksum",
        "Type": "string"
    }
]
```

Supported schema for CloudTrail Insights event record fields

The following is the valid SQL schema for Insights event record fields. For Insights events, the value of `eventcategory` is `Insight`, and the value of `eventtype` is `AwsCloudTrailInsight`.

```
[
    {
        "Name": "eventversion",
        "Type": "string"
    },
    {
        "Name": "eventcategory",
        "Type": "string"
    },
    {
        "Name": "eventtype",
        "Type": "string"
    },
        "Name": "eventid",
        "Type": "string"
    },
    {
        "Name": "eventtime",
        "Type": "timestamp"
    },
    {
        "Name": "awsregion",
        "Type": "string"
    },
    {
        "Name": "recipientaccountid",
        "Type": "string"
    },
    {
        "Name": "sharedeventid",
        "Type": "string"
    },
    {
        "Name": "addendum",
        "Type": "map<string,string>"
    },
    {
        "Name": "insightsource",
        "Type": "string"
    },
    {
        "Name": "insightstate",
        "Type": "string"
    },
    {
        "Name": "insighteventsource",
        "Type": "string"
    },
    {
        "Name": "insighteventname",
        "Type": "string"
    },
    {
        "Name": "insighterrorcode",
        "Type": "string"
    },
    {
        "Name": "insighttype",
        "Type": "string"
    },
    {
        "Name": "insightContext",
        "Type":
 "struct<baselineaverage:double,insightaverage:double,baselineduration:integer,
 insightduration:integer,attributions:struct<attribute:string,insightvalue:string,
                 insightaverage:double,baselinevalue:string,baselineaverage:double>>"
    }
]
```

Supported schema for AWS Config configuration item record fields

The following is the valid SQL schema for configuration item record fields. For configuration items, the value of `eventcategory` is `ConfigurationItem`, and the value of `eventtype` is `AwsConfigurationItem`.

```
[
    {
        "Name": "eventversion",
        "Type": "string"
    },
    {
        "Name": "eventcategory",
        "Type": "string"
    },
    {
        "Name": "eventtype",
        "Type": "string"
    },
        "Name": "eventid",
        "Type": "string"
    },
    {
        "Name": "eventtime",
        "Type": "timestamp"
    },
    {
        "Name": "awsregion",
        "Type": "string"
    },
    {
        "Name": "recipientaccountid",
        "Type": "string"
    },
    {
        "Name": "addendum",
        "Type": "map<string,string>"
    },
    {
        "Name": "eventdata",
        "Type": "struct<configurationitemversion:string,configurationitemcapturetime:

 string,configurationitemstatus:string,configurationitemstateid:string,accountid:string,

 resourcetype:string,resourceid:string,resourcename:string,arn:string,awsregion:string,

 availabilityzone:string,resourcecreationtime:string,configuration:map<string,string>,
                 supplementaryconfiguration:map<string,string>,relatedevents:string,

 relationships:struct<name:string,resourcetype:string,resourceid:string,
                 resourcename:string>,tags:map<string,string>>"
    }
]
```

Supported schema for AWS Audit Manager evidence record fields

The following is the valid SQL schema for Audit Manager evidence record fields. For Audit Manager evidence record fields, the value of `eventcategory` is `Evidence`, and the value of `eventtype` is `AwsAuditManagerEvidence`. For more information about aggregating evidence in CloudTrail Lake using Audit Manager, see Evidence finder in the *AWS Audit Manager User Guide*.

```
[
    {
        "Name": "eventversion",
        "Type": "string"
    },
    {
        "Name": "eventcategory",
        "Type": "string"
    },
    {
        "Name": "eventtype",
        "Type": "string"
    },
        "Name": "eventid",
        "Type": "string"
    },
    {
        "Name": "eventtime",
        "Type": "timestamp"
    },
    {
        "Name": "awsregion",
        "Type": "string"
    },
    {
        "Name": "recipientaccountid",
        "Type": "string"
    },
    {
        "Name": "addendum",
        "Type": "map<string,string>"
    },
    {
        "Name": "eventdata",
        "Type":
 "struct<attributes:map<string,string>,awsaccountid:string,awsorganization:string,

 compliancecheck:string,datasource:string,eventname:string,eventsource:string,

 evidenceawsaccountid:string,evidencebytype:string,iamid:string,evidenceid:string,

 time:timestamp,assessmentid:string,controlsetid:string,controlid:string,

 controlname:string,controldomainname:string,frameworkname:string,frameworkid:string,

 service:string,servicecategory:string,resourcearn:string,resourcetype:string,

 evidencefolderid:string,description:string,manualevidences3resourcepath:string,
                 evidencefoldername:string,resourcecompliancecheck:string>"
    }
]
```

Supported schema for non-AWS event fields

The following is the valid SQL schema for non-AWS events. For non-AWS events, the value of `eventcategory` is `ActivityAuditLog`, and the value of `eventtype` is `ActivityLog`.

```
[
    {
        "Name": "eventversion",
        "Type": "string"
    },
    {
        "Name": "eventcategory",
        "Type": "string"
    },
    {
        "Name": "eventtype",
        "Type": "string"
    },
        "Name": "eventid",
        "Type": "string"
    },
    {
        "Name": "eventtime",
        "Type": "timestamp"
    },
    {
        "Name": "awsregion",
        "Type": "string"
    },
    {
        "Name": "recipientaccountid",
        "Type": "string"
    },
    {
        "Name": "addendum",
        "Type":
 "struct<reason:string,updatedfields:string,originalUID:string,originaleventid:string>"
    },
    {
        "Name": "metadata",
        "Type": "struct<ingestiontime:string,channelarn:string>"
    },
    {
        "Name": "eventdata",
        "Type": "struct<version:string,useridentity:struct<type:string,

 principalid:string,details:map<string,string>>,useragent:string,eventsource:string,

 eventname:string,eventtime:string,uid:string,requestparameters:map<string,string>>,

 responseelements":map<string,string>>,errorcode:string,errormssage:string,sourceipaddress:stri
                 recipientaccountid:string,additionaleventdata":map<string,string>>"
    }
```

```
]
```

Supported CloudWatch metrics

CloudTrail Lake supports Amazon CloudWatch metrics. CloudWatch is a monitoring service for AWS resources. You can use CloudWatch to collect and track metrics, set alarms, and automatically react to changes in your AWS resources.

The `AWS/CloudTrail` namespace includes the following metrics for CloudTrail Lake.

Metric	Description	Units
`HourlyDataIngested`	The amount of data ingested into the event data store during the last hour. This metric is updated every hour. This metric is available for all event data store types.	Bytes
`TotalDataRetained`	The amount of data retained in the event data store during its entire retention period. This metric is updated nightly. This metric is available for all event data store types.	Bytes
`TotalStorageBytes`	The total compressed bytes in the event data store as of the current day. This metric is available for all event data store types.	Bytes
`TotalPaidStorageBytes`	For event data stores using the one-year extendable retention pricing option, this is the total compressed	Bytes

Metric	Description	Units
	bytes after 366 days to the maximum retention period configured for the event data store. For event data stores using the one-year extendable retention pricing option, storage is included at no additional cost with ingestion pricing for the first 366 days, which is the default retention period for the event data store. After 366 days, storage is pay-as-you-go. For information about pricing, see AWS CloudTrail Pricing. This metric is only available for event data stores using the one-year extendable retention pricing option.	
`HourlyEventsAnalyzed`	The total number of events analyzed by CloudTrail Insights in the event data store. This metric is updated every hour. This metric is for CloudTrail event data stores that enable CloudTrail Insights.	Count

For more information about CloudWatch metrics, see the following topics.

- Using Amazon CloudWatch metrics

- Using Amazon CloudWatch alarms

Working with CloudTrail trails

Trails capture a record of AWS activities, delivering and storing these events in an Amazon S3 bucket, with optional delivery to CloudWatch Logs and Amazon EventBridge.

You can deliver one copy of your ongoing management events to your S3 bucket at no charge from CloudTrail by creating a trail, however, there are Amazon S3 storage charges. For more information about CloudTrail pricing, see AWS CloudTrail Pricing. For information about Amazon S3 pricing, see Amazon S3 Pricing.

You can create two types of trails for an AWS account: multi-Region trails and single-Region trails.

Multi-Region trails

When you create a multi-Region trail, CloudTrail records events in all AWS Regions in the AWS partition in which you are working and delivers the CloudTrail event log files to an S3 bucket that you specify. If an AWS Region is added after you create a multi-Region trail, that new Region is automatically included, and events in that Region are logged. Creating a multi-Region trail is a recommended best practice since you capture activity in all Regions in your account. All trails you create using the CloudTrail console are multi-Region. You can convert a single-Region trail to a multi-Region trail by using the AWS CLI. For more information, see Creating a trail in the console and Converting a trail that applies to one Region to apply to all Regions.

Single-Region trails

When you create a single-Region trail, CloudTrail records the events in that Region only. It then delivers the CloudTrail event log files to an Amazon S3 bucket that you specify. You can only create a single-Region trail by using the AWS CLI. If you create additional single trails, you can have those trails deliver CloudTrail event log files to the same S3 bucket or to separate buckets. This is the default option when you create a trail using the AWS CLI or the CloudTrail API. For more information, see Creating, updating, and managing trails with the AWS CLI.

> **Note**
>
> For both types of trails, you can specify an Amazon S3 bucket from any Region.

If you have created an organization in AWS Organizations, you can create an *organization trail* that logs all events for all AWS accounts in that organization. Organization trails can apply to all

AWS Regions, or the current Region. Organization trails must be created using the management account or delegated administrator account, and when specified as applying to an organization, are automatically applied to all member accounts in the organization. Member accounts can see the organization trail, but cannot modify or delete it. By default, member accounts do not have access to the log files for an organization trail in the Amazon S3 bucket. For more information, see Creating a trail for an organization.

Topics

Creating a trail for your AWS account

When you create a trail, you enable ongoing delivery of events as log files to an Amazon S3 bucket that you specify. Creating a trail has many benefits, including:

- A record of events that extends past 90 days.
- The option to automatically monitor and alarm on specified events by sending log events to Amazon CloudWatch Logs.
- The option to query logs and analyze AWS service activity with Amazon Athena.

Beginning on April 12, 2019, you can view trails only in the AWS Regions where they log events. If you create a trail that logs events in all AWS Regions, it appears in the console in all Regions in the AWS partition in which you are working. If you create a trail that only logs events in a single Region, you can view and manage it only in that Region. Creating a multi-Region trail is the default option if you create a trail by using the AWS CloudTrail console, and is a recommended best practice. To create a single-Region trail, you must use the AWS CLI.

If you use AWS Organizations, you can create a trail that will log events for all AWS accounts in the organization. A trail with the same name will be created in each member account, and events from each trail will be delivered to the Amazon S3 bucket that you specify.

> **Note**
>
> Only the management account or delegated administrator account for an organization can create a trail for the organization. Creating a trail for an organization automatically enables integration between CloudTrail and Organizations. For more information, see Creating a trail for an organization.

Topics

- Creating and updating a trail with the console
- Creating, updating, and managing trails with the AWS CLI
- Creating multiple trails

Creating and updating a trail with the console

You can use the CloudTrail console to create, update, or delete your trails. Trails created using the console are multi-Region. To create a trail that logs events in only one AWS Region, use the AWS CLI.

You can create up to five trails for each Region. After you create a trail, CloudTrail automatically starts logging API calls and related events in your account to the Amazon S3 bucket that you specify.

You can change the following settings for your trail using the CloudTrail console:

- You can change the S3 bucket location and specify a prefix.
- The management account for an AWS Organizations organization can convert an account-level trail to an organization trail, or can convert an organization trail to an account-level trail.
- You can enable or disable KMS key encryption.
- You can enable or disable log file validation. Log file validation allows you to determine whether a log file was modified, deleted, or unchanged after CloudTrail delivered it. By default, log file validation is enabled.

- You can configure a trail to send notifications to an Amazon SNS topic.
- You can configure a trail to send events to a CloudWatch Logs log group. Both the log group and IAM role must exist in your own account.
- You can update settings for management events, data events, and Insights events.
- You can add or remove tags. You can add up to 50 tag key pairs to help you identify your trails.

Using the CloudTrail console to create or update a trail provides the following advantages.

- If this is your first time creating a trail, using the CloudTrail console lets you view the available feature and options.
- If you are configuring a trail to log data events, using the CloudTrail console lets you view the available data types. For more information about logging data events, see Logging data events.
- If your trail is already logging data events, using the console allows you to easily add logging for additional data event types.

For information specific to creating a trail for an organization in AWS Organizations, see Creating a trail for an organization.

Topics

Creating a trail with the CloudTrail console

As a best practice, create a trail that applies to all AWS Regions. This is the default setting when you create a trail in the CloudTrail console. When a trail applies to all Regions, CloudTrail delivers log files from all Regions in the AWS partition in which you are working to an S3 bucket that you specify. After you create the trail, AWS CloudTrail automatically starts logging the events that you specified.

Note

After you create a trail, you can configure other AWS services to further analyze and act upon the event data collected in CloudTrail logs. For more information, see AWS service integrations with CloudTrail logs.

Topics

- Creating a trail in the console
- Next steps

Creating a trail in the console

Use the following procedure to create a trail that logs events in all AWS Regions in the AWS partition in which you are working. This is a recommended best practice. To log events in a single Region (not recommended), use the AWS CLI.

To create a CloudTrail trail with the AWS Management Console

1. Sign in to the AWS Management Console and open the CloudTrail console at https://console.aws.amazon.com/cloudtrail/.
2. On the CloudTrail service home page, the **Trails** page, or the **Trails** section of the **Dashboard** page, choose **Create trail**.
3. On the **Create Trail** page, for **Trail name**, type a name for your trail. For more information, see Naming requirements for CloudTrail resources, S3 buckets, and KMS keys.
4. If this is an AWS Organizations organization trail, you can enable the trail for all accounts in your organization. To see this option, you must sign in to the console with a user or role in the management or delegated administrator account. To successfully create an organization trail, be sure that the user or role has sufficient permissions. For more information, see Creating a trail for an organization.
5. For **Storage location**, choose **Create new S3 bucket** to create a bucket. When you create a bucket, CloudTrail creates and applies the required bucket policies. If you choose to create a new S3 bucket, your IAM policy needs to include permission for the `s3:PutEncryptionConfiguration` action because by default server-side encryption is enabled for the bucket.

> **Note**
>
> If you chose **Use existing S3 bucket**, specify a bucket in **Trail log bucket name**, or choose **Browse** to choose a bucket in your own account. If you want to use a bucket in another account, you'll need to specify the bucket name. The bucket policy must grant CloudTrail permission to write to it. For information about manually editing the bucket policy, see Amazon S3 bucket policy for CloudTrail.

To make it easier to find your logs, create a new folder (also known as a *prefix*) in an existing bucket to store your CloudTrail logs. Enter the prefix in **Prefix**.

6. For **Log file SSE-KMS encryption**, choose **Enabled** if you want to encrypt your log files using SSE-KMS encryption instead of SSE-S3 encryption. The default is **Enabled**. If you don't enable SSE-KMS encryption, your logs are encrypted using SSE-S3 encryption. For more information about SSE-KMS encryption, see Using server-side encryption with AWS Key Management Service (SSE-KMS). For more information about SSE-S3 encryption, see Using Server-Side Encryption with Amazon S3-Managed Encryption Keys (SSE-S3).

   If you enable SSE-KMS encryption, choose a **New** or **Existing** AWS KMS key. In **AWS KMS Alias**, specify an alias, in the format `alias/MyAliasName`. For more information, see Updating a resource to use your KMS key with the console. CloudTrail also supports AWS KMS multi-Region keys. For more information about multi-Region keys, see Using multi-Region keys in the *AWS Key Management Service Developer Guide*.

   > **Note**
   >
   > You can also type the ARN of a key from another account. For more information, see Updating a resource to use your KMS key with the console. The key policy must allow CloudTrail to use the key to encrypt your log files, and allow the users you specify to read log files in unencrypted form. For information about manually editing the key policy, see Configure AWS KMS key policies for CloudTrail.

7. In **Additional settings**, configure the following.

   a. For **Log file validation**, choose **Enabled** to have log digests delivered to your S3 bucket. You can use the digest files to verify that your log files did not change after CloudTrail delivered them. For more information, see Validating CloudTrail log file integrity.

b. For **SNS notification delivery**, choose **Enabled** to be notified each time a log is delivered to your bucket. CloudTrail stores multiple events in a log file. SNS notifications are sent for every log file, not for every event. For more information, see Configuring Amazon SNS notifications for CloudTrail.

   If you enable SNS notifications, for **Create a new SNS topic**, choose **New** to create a topic, or choose **Existing** to use an existing topic. If you are creating a trail that applies to all Regions, SNS notifications for log file deliveries from all Regions are sent to the single SNS topic that you create.

   If you choose **New**, CloudTrail specifies a name for the new topic for you, or you can type a name. If you choose **Existing**, choose an SNS topic from the drop-down list. You can also enter the ARN of a topic from another Region or from an account with appropriate permissions. For more information, see Amazon SNS topic policy for CloudTrail.

   If you create a topic, you must subscribe to the topic to be notified of log file delivery. You can subscribe from the Amazon SNS console. Due to the frequency of notifications, we recommend that you configure the subscription to use an Amazon SQS queue to handle notifications programmatically. For more information, see Getting started with Amazon SNS in the *Amazon Simple Notification Service Developer Guide*.

8. Optionally, configure CloudTrail to send log files to CloudWatch Logs by choosing **Enabled** in **CloudWatch Logs**. For more information, see Sending events to CloudWatch Logs.

   a. If you enable integration with CloudWatch Logs, choose **New** to create a new log group, or **Existing** to use an existing one. If you choose **New**, CloudTrail specifies a name for the new log group for you, or you can type a name.

   b. If you choose **Existing**, choose a log group from the drop-down list.

   c. Choose **New** to create a new IAM role for permissions to send logs to CloudWatch Logs. Choose **Existing** to choose an existing IAM role from the drop-down list. The policy statement for the new or existing role is displayed when you expand **Policy document**. For more information about this role, see Role policy document for CloudTrail to use CloudWatch Logs for monitoring.

      > **Note**
      >
      > - When you configure a trail, you can choose an S3 bucket and SNS topic that belong to another account. However, if you want CloudTrail to deliver events to

> a CloudWatch Logs log group, you must choose a log group that exists in your current account.
>
> - Only the management account can configure a CloudWatch Logs log group for an organization trail using the console. The delegated administrator can configure a CloudWatch Logs log group using the AWS CLI or CloudTrail `CreateTrail` or `UpdateTrail` API operations.

9. For **Tags**, add one or more custom tags (key-value pairs) to your trail. Tags can help you identify both your CloudTrail trails and the Amazon S3 buckets that contain CloudTrail log files. You can then use resource groups for your CloudTrail resources. For more information, see AWS Resource Groups and Tags.

10. On the **Choose log events** page, choose the event types that you want to log. For **Management events**, do the following.

    a. For **API activity**, choose if you want your trail to log **Read** events, **Write** events, or both. For more information, see Management events.

    b. Choose **Exclude AWS KMS events** to filter AWS Key Management Service (AWS KMS) events out of your trail. The default setting is to include all AWS KMS events.

       The option to log or exclude AWS KMS events is available only if you log management events on your trail. If you choose not to log management events, AWS KMS events are not logged, and you cannot change AWS KMS event logging settings.

       AWS KMS actions such as `Encrypt`, `Decrypt`, and `GenerateDataKey` typically generate a large volume (more than 99%) of events. These actions are now logged as **Read** events. Low-volume, relevant AWS KMS actions such as `Disable`, `Delete`, and `ScheduleKey` (which typically account for less than 0.5% of AWS KMS event volume) are logged as **Write** events.

       To exclude high-volume events like `Encrypt`, `Decrypt`, and `GenerateDataKey`, but still log relevant events such as `Disable`, `Delete` and `ScheduleKey`, choose to log **Write** management events, and clear the check box for **Exclude AWS KMS events**.

    c. Choose **Exclude Amazon RDS Data API events** to filter Amazon Relational Database Service Data API events out of your trail. The default setting is to include all Amazon RDS Data API events. For more information about Amazon RDS Data API events, see Logging Data API calls with AWS CloudTrail in the *Amazon RDS User Guide for Aurora*.

11. To log data events, choose **Data events**. Additional charges apply for logging data events. For more information, see AWS CloudTrail Pricing.

12. > ⚠ **Important**
    >
    > Steps 12-16 are for configuring data events using advanced event selectors, which is the default. Advanced event selectors let you configure more data event types and offer fine-grained control over which data events your trail captures. If you opted to use basic event selectors, complete the steps in Configure data event settings using basic event selectors, then return to step 17 of this procedure.

    For **Data event type**, choose the resource type on which you want to log data events. For more information about available data event types, see Data events.

    > ⓘ **Note**
    >
    > To log data events for AWS Glue tables created by Lake Formation, choose **Lake Formation**.

13. Choose a log selector template. CloudTrail includes predefined templates that log all data events for the resource type. To build a custom log selector template, choose **Custom**.

    > ⓘ **Note**
    >
    > Choosing a predefined template for S3 buckets enables data event logging for all buckets currently in your AWS account and any buckets you create after you finish creating the trail. It also enables logging of data event activity performed by any IAM identity in your AWS account, even if that activity is performed on a bucket that belongs to another AWS account.
    > If the trail applies only to one Region, choosing a predefined template that logs all S3 buckets enables data event logging for all buckets in the same Region as your trail and any buckets you create later in that Region. It will not log data events for Amazon S3 buckets in other Regions in your AWS account.
    > If you are creating a trail for all Regions, choosing a predefined template for Lambda functions enables data event logging for all functions currently in your AWS account, and any Lambda functions you might create in any Region after you finish creating the trail. If you are creating a trail for a single Region (done by using the AWS CLI), this

selection enables data event logging for all functions currently in that Region in your AWS account, and any Lambda functions you might create in that Region after you finish creating the trail. It does not enable data event logging for Lambda functions created in other Regions.

Logging data events for all functions also enables logging of data event activity performed by any IAM identity in your AWS account, even if that activity is performed on a function that belongs to another AWS account.

14. (Optional) In **Selector name**, enter a name to identify your selector. The selector name is a descriptive name for an advanced event selector, such as "Log data events for only two S3 buckets". The selector name is listed as `Name` in the advanced event selector and is viewable if you expand the **JSON view**.

15. In **Advanced event selectors**, build an expression for the specific resources on which you want to log data events. You can skip this step if you are using a predefined log template.

    a. Choose from the following fields.

    - **`readOnly`** - `readOnly` can be set to **equals** a value of `true` or `false`. Read-only data events are events that do not change the state of a resource, such as `Get*` or `Describe*` events. Write events add, change, or delete resources, attributes, or artifacts, such as `Put*`, `Delete*`, or `Write*` events. To log both `read` and `write` events, don't add a `readOnly` selector.
    - **`eventName`** - `eventName` can use any operator. You can use it to include or exclude any data event logged to CloudTrail, such as `PutBucket`, `PutItem`, or `GetSnapshotBlock`.
    - **`resources.ARN`** - You can use any operator with `resources.ARN`, but if you use **equals** or **does not equal**, the value must exactly match the ARN of a valid resource of the type you've specified in the template as the value of `resources.type`.

      The following table shows the valid ARN format for each `resources.type`.

      > **Note**
      >
      > You can't use the `resources.ARN` field to filter resource types that do not have ARNs.

resources.type	resources.ARN
`AWS::DynamoDB::Table` [1]	`arn:partition:dynamodb:region:account_ID:table/table_name`
`AWS::Lambda::Function`	`arn:partition:lambda:region:account_ID:function:function_name`
`AWS::S3::Object` [2]	`arn:partition:s3:::DOC-EXAMPLE-BUCKET/` `arn:partition:s3:::DOC-EXAMPLE-BUCKET/object_or_file_name/`
`AWS::AppConfig::Configuration`	`arn:partition:appconfig:region:account_ID:application/application_ID/environment/environment_ID/configuration/configuration_profile_ID`
`AWS::B2BI::Transformer`	`arn:partition:b2bi:region:account_ID:transformer/transformer_ID`
`AWS::Bedrock::AgentAlias`	`arn:partition:bedrock:region:account_ID:agent-alias/agent_ID/alias_ID`
`AWS::Bedrock::FlowAlias`	`arn:partition:bedrock:region:account_ID:flow/flow_ID/alias/alias_ID`
`AWS::Bedrock::Guardrail`	`arn:partition:bedrock:region:account_ID:guardrail/guardrail_ID`

resources.type	resources.ARN
AWS::Bedrock::KnowledgeBase	arn:*partition* :bedrock: *region*:*account_ID* :knowledge-base/*knowledge_base_ID*
AWS::Cassandra::Table	arn:*partition* :cassandra: *region*:*account_ID* :keyspace / *keyspace_name* /table/*table_name*
AWS::CloudFront::KeyValueStore	arn:*partition* :cloudfront: *region*:*account_ID* :key-value-store/*KVS_name*
AWS::CloudTrail::Channel	arn:*partition* :cloudtrail: *region*:*account_ID* :channel/ *channel_UUID*
AWS::CodeWhisperer::Customization	arn:*partition* :codewhisperer: *region*:*account_ID* :customization/ *customization_ID*
AWS::CodeWhisperer::Profile	arn:*partition* :codewhisperer: *region*:*account_ID* :profile/ *profile_ID*
AWS::Cognito::IdentityPool	arn:*partition* :cognito-identity: *region*:*account_ID* :identity pool/ *identity_pool_ID*
AWS::DynamoDB::Stream	arn:*partition* :dynamodb : *region*:*account_ID* :table/*table_name* / stream/*date_time*

resources.type	resources.ARN
`AWS::EC2::Snapshot`	arn:*partition*:ec2:*region*::snapshot/*snapshot_ID*
`AWS::EMRWAL::Workspace`	arn:*partition*:emrwal:*region*:*account_ID*:workspace/*workspace_name*
`AWS::FinSpace::Environment`	arn:*partition*:finspace:*region*:*account_ID*:environment/*environment_ID*
`AWS::Glue::Table`	arn:*partition*:glue:*region*:*account_ID*:table/*database_name*/*table_name*
`AWS::GreengrassV2::ComponentVersion`	arn:*partition*:greengrass:*region*:*account_ID*:components/*component_name*
`AWS::GreengrassV2::Deployment`	arn:*partition*:greengrass:*region*:*account_ID*:deployments/*deployment_ID*
`AWS::GuardDuty::Detector`	arn:*partition*:guardduty:*region*:*account_ID*:detector/*detector_ID*
`AWS::IoT::Certificate`	arn:*partition*:iot:*region*:*account_ID*:cert/*certificate_ID*
`AWS::IoT::Thing`	arn:*partition*:iot:*region*:*account_ID*:thing/*thing_ID*

resources.type	resources.ARN
`AWS::IoTSiteWise::Asset`	`arn:partition:iotsitewise:region:account_ID:asset/asset_ID`
`AWS::IoTSiteWise::TimeSeries`	`arn:partition:iotsitewise:region:account_ID:timeseries/timeseries_ID`
`AWS::IoTTwinMaker::Entity`	`arn:partition:iottwinmaker:region:account_ID:workspace/workspace_ID/entity/entity_ID`
`AWS::IoTTwinMaker::Workspace`	`arn:partition:iottwinmaker:region:account_ID:workspace/workspace_ID`
`AWS::KendraRanking::ExecutionPlan`	`arn:partition:kendra-ranking:region:account_ID:rescore-execution-plan/rescore_execution_plan_ID`
`AWS::Kinesis::Stream`	`arn:partition:kinesis:region:account_ID:stream/stream_name`
`AWS::Kinesis::StreamConsumer`	`arn:partition:kinesis:region:account_ID:stream_type/stream_name/consumer/consumer_name:consumer_creation_timestamp`
`AWS::KinesisVideo::Stream`	`arn:partition:kinesisvideo:region:account_ID:stream/stream_name/creation_time`

resources.type	resources.ARN
AWS::MachineLearning::MlModel	arn:*partition*:machinelearning:*region*:*account_ID*:mlmodel/*model_ID*
AWS::ManagedBlockchain::Network	arn:*partition*:managedblockchain:::networks/*network_name*
AWS::ManagedBlockchain::Node	arn:*partition*:managedblockchain:*region*:*account_ID*:nodes/*node_ID*
AWS::MedicalImaging::Datastore	arn:*partition*:medical-imaging:*region*:*account_ID*:datastore/*data_store_ID*
AWS::NeptuneGraph::Graph	arn:*partition*:neptune-graph:*region*:*account_ID*:graph/*graph_ID*
AWS::PaymentCryptography::Alias	arn:*partition*:payment-cryptography:*region*:*account_ID*:alias/*alias*
AWS::PaymentCryptography::Key	arn:*partition*:payment-cryptography:*region*:*account_ID*:key/*key_ID*
AWS::PCAConnectorAD::Connector	arn:*partition*:pca-connector-ad:*region*:*account_ID*:connector/*connector_ID*
AWS::PCAConnectorSCEP::Connector	arn:*partition*:pca-connector-scep:*region*:*account_ID*:connector/*connector_ID*

resources.type	resources.ARN
AWS::QApps:QApp	arn:*partition*:qapps:*region*:*account_ID*:application/*application_UUID*/qapp/*qapp_UUID*
AWS::QBusiness::Application	arn:*partition*:qbusiness:*region*:*account_ID*:application/*application_ID*
AWS::QBusiness::DataSource	arn:*partition*:qbusiness:*region*:*account_ID*:application/*application_ID*/index/*index_ID*/data-source/*datasource_ID*
AWS::QBusiness::Index	arn:*partition*:qbusiness:*region*:*account_ID*:application/*application_ID*/index/*index_ID*
AWS::QBusiness::WebExperience	arn:*partition*:qbusiness:*region*:*account_ID*:application/*application_ID*/web-experience/*web_experienc_ID*
AWS::RDS::DBCluster	arn:*partition*:rds:*region*:*account_ID*:cluster/*cluster_name*
AWS::S3::AccessPoint [3]	arn:*partition*:s3:*region*:*account_ID*:accesspoint/*access_point_name*
AWS::S3Express::Object	arn:*partition*:s3express:*region*:*account_ID*:bucket/*bucket_name*

resources.type	resources.ARN
`AWS::S3ObjectLambda::AccessPoint`	`arn:partition:s3-object-lambda:region:account_ID:accesspoint/access_point_name`
`AWS::S3Outposts::Object`	`arn:partition:s3-outposts:region:account_ID:object_path`
`AWS::SageMaker::Endpoint`	`arn:partition:sagemaker:region:account_ID:endpoint/endpoint_name`
`AWS::SageMaker::ExperimentTrialComponent`	`arn:partition:sagemaker:region:account_ID:experiment-trial-component/experiment_trial_component_name`
`AWS::SageMaker::FeatureGroup`	`arn:partition:sagemaker:region:account_ID:feature-group/feature_group_name`
`AWS::SCN::Instance`	`arn:partition:scn:region:account_ID:instance/instance_ID`
`AWS::ServiceDiscovery::Namespace`	`arn:partition:servicediscovery:region:account_ID:namespace/namespace_ID`
`AWS::ServiceDiscovery::Service`	`arn:partition:servicediscovery:region:account_ID:service/service_ID`

resources.type	resources.ARN
`AWS::SNS::PlatformEndpoint`	`arn:partition:sns:region:account_ID:endpoint/endpoint_type/endpoint_name/endpoint_ID`
`AWS::SNS::Topic`	`arn:partition:sns:region:account_ID:topic_name`
`AWS::SQS::Queue`	`arn:partition:sqs:region:account_ID:queue_name`
`AWS::SSM::ManagedNode`	The ARN must be in one of the following formats: • `arn:partition:ssm:region:account_ID:managed-instance/instance_ID` • `arn:partition:ec2:region:account_ID:instance/instance_ID`
`AWS::SSMMessages::ControlChannel`	`arn:partition:ssmmessages:region:account_ID:control-channel/control_channel_ID`
`AWS::StepFunctions::StateMachine`	The ARN must be in one of the following formats: • `arn:partition:states:region:account_ID:stateMachine:stateMachine_name` • `arn:partition:states:region:account_ID:stateMachine:stateMachine_name/label_name`

resources.type	resources.ARN
`AWS::SWF::Domain`	`arn:partition :swf:region:account_ID :/domain/ domain_name`
`AWS::ThinClient::Device`	`arn:partition :thinclient: region:account_ID :device/device_ID`
`AWS::ThinClient::Environment`	`arn:partition :thinclient: region:account_ID :environment/ environment_ID`
`AWS::Timestream::Database`	`arn:partition :timestream: region:account_ID :database/ database_name`
`AWS::Timestream::Table`	`arn:partition :timestream: region:account_ID :database/ database_name /table/table_name`
`AWS::VerifiedPermissions::PolicyStore`	`arn:partition :verifiedpermissions: region:account_ID :policy-store/ policy_store_ID`

[1] For tables with streams enabled, the `resources` field in the data event contains both `AWS::DynamoDB::Stream` and `AWS::DynamoDB::Table`. If you specify `AWS::DynamoDB::Table` for the `resources.type`, it will log both DynamoDB table and DynamoDB streams events by default. To exclude streams events, add a filter on the `eventName` field.

[2] To log all data events for all objects in a specific S3 bucket, use the `StartsWith` operator, and include only the bucket ARN as the matching value. The trailing slash is intentional; do not exclude it.

[3] To log events on all objects in an S3 access point, we recommend that you use only the access point ARN, don't include the object path, and use the `StartsWith` or `NotStartsWith` operators.

For more information about the ARN formats of data event resources, see [Actions, resources, and condition keys]() in the *AWS Identity and Access Management User Guide*.

b. For each field, choose **+ Condition** to add as many conditions as you need, up to a maximum of 500 specified values for all conditions. For example, to exclude data events for two S3 buckets from data events that are logged on your trail, you can set the field to **resources.ARN**, set the operator for **does not start with**, and then either paste in an S3 bucket ARN, or browse for the S3 buckets for which you do not want to log events.

   To add the second S3 bucket, choose **+ Condition**, and then repeat the preceding instruction, pasting in the ARN for or browsing for a different bucket.

   > **Note**
   >
   > You can have a maximum of 500 values for all selectors on a trail. This includes arrays of multiple values for a selector such as `eventName`. If you have single values for all selectors, you can have a maximum of 500 conditions added to a selector.
   > If you have more than 15,000 Lambda functions in your account, you cannot view or select all functions in the CloudTrail console when creating a trail. You can still log all functions with a predefined selector template, even if they are not displayed. If you want to log data events for specific functions, you can manually add a function if you know its ARN. You can also finish creating the trail in the console, and then use the AWS CLI and the **put-event-selectors** command to configure data event logging for specific Lambda functions. For more information, see [Managing trails with the AWS CLI]().

c. Choose **+ Field** to add additional fields as required. To avoid errors, do not set conflicting or duplicate values for fields. For example, do not specify an ARN in one selector to be equal to a value, then specify that the ARN not equal the same value in another selector.

16. To add another data type on which to log data events, choose **Add data event type**. Repeat steps 12 through this step to configure advanced event selectors for the data event type.

17. Choose **Insights events** if you want your trail to log CloudTrail Insights events.

In **Event type**, select **Insights events**. You must be logging **Write** management events to log Insights events for **API call rate**. You must be logging **Read** or **Write** management events to log Insights events for **API error rate**.

CloudTrail Insights analyzes management events for unusual activity, and logs events when anomalies are detected. By default, trails don't log Insights events. For more information about Insights events, see Logging Insights events. Additional charges apply for logging Insights events. For CloudTrail pricing, see AWS CloudTrail Pricing.

Insights events are delivered to a different folder named `/CloudTrail-Insight`of the same S3 bucket that is specified in the **Storage location** area of the trail details page. CloudTrail creates the new prefix for you. For example, if your current destination S3 bucket is named `DOC-EXAMPLE-BUCKET/AWSLogs/CloudTrail/`, the S3 bucket name with a new prefix is named `DOC-EXAMPLE-BUCKET/AWSLogs/CloudTrail-Insight/`.

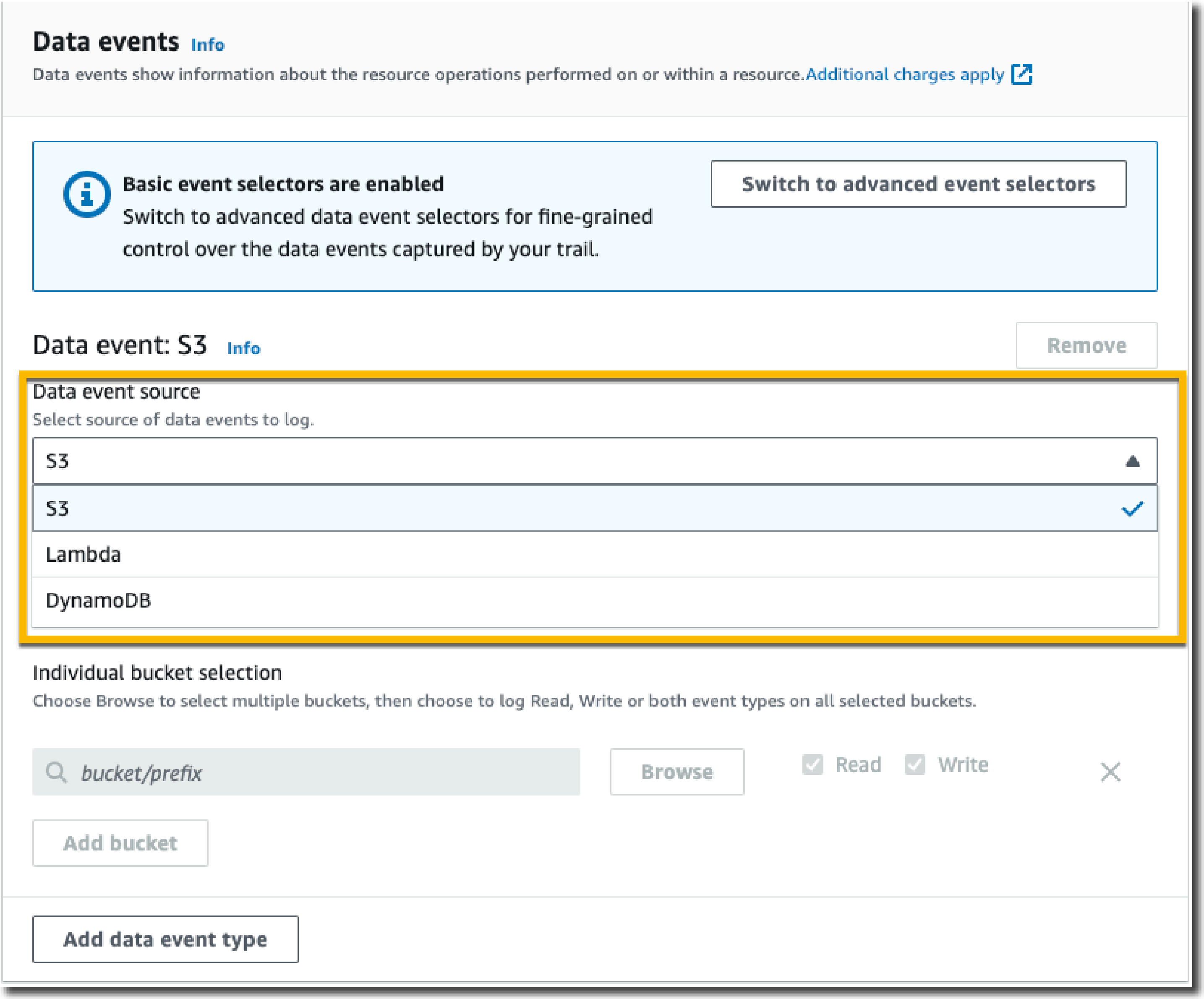
18. When you are finished choosing event types to log, choose **Next**.
19. On the **Review and create** page, review your choices. Choose **Edit** in a section to change the trail settings shown in that section. When you are ready to create the trail, choose **Create trail**.
20. The new trail appears on the **Trails** page. In about 5 minutes, CloudTrail publishes log files that show the AWS API calls made in your account. You can see the log files in the S3 bucket that you specified. It can take up to 36 hours for CloudTrail to deliver the first Insights event, if you have enabled Insights event logging, and unusual activity is detected.

> **Note**
>
> CloudTrail typically delivers logs within an average of about 5 minutes of an API call. This time is not guaranteed. Review the AWS CloudTrail Service Level Agreement for more information.
> If you misconfigure your trail (for example, the S3 bucket is unreachable), CloudTrail will attempt to redeliver the log files to your S3 bucket for 30 days, and these attempted-to-deliver events will be subject to standard CloudTrail charges. To avoid charges on a misconfigured trail, you need to delete the trail.

Configure data event settings using basic event selectors

You can use advanced event selectors to configure all data event types. Advanced event selectors let you create fine-grained selectors to log only those events of interest.

If you use basic event selectors to log data events, you're limited to logging data events for Amazon S3 buckets, AWS Lambda functions, and Amazon DynamoDB tables. You can't filter on the `eventName` field using basic event selectors.



Data events Info

Data events show information about the resource operations performed on or within a resource.Additional charges apply

Basic event selectors are enabled

Switch to advanced data event selectors for fine-grained control over the data events captured by your trail.

Data event: S3 Info

Data event source

Select source of data events to log.

S3

Individual bucket selection

Choose Browse to select multiple buckets, then choose to log Read, Write or both event types on all selected buckets.

Use the following procedure to configure data event settings using basic event selectors.

To configure data event settings using basic event selectors

1. In **Events**, choose **Data events** to log data events. Additional charges apply for logging data events. For more information, see AWS CloudTrail Pricing.
2. For Amazon S3 buckets:

a. For **Data event source**, choose **S3**.

b. You can choose to log **All current and future S3 buckets**, or you can specify individual buckets or functions. By default, data events are logged for all current and future S3 buckets.

> **Note**
>
> Keeping the default **All current and future S3 buckets** option enables data event logging for all buckets currently in your AWS account and any buckets you create after you finish creating the trail. It also enables logging of data event activity performed by any IAM identity in your AWS account, even if that activity is performed on a bucket that belongs to another AWS account.
> If you are creating a trail for a single Region (done by using the AWS CLI), choosing **All current and future S3 buckets** enables data event logging for all buckets in the same Region as your trail and any buckets you create later in that Region.
> It will not log data events for Amazon S3 buckets in other Regions in your AWS account.

c. If you leave the default, **All current and future S3 buckets**, choose to log **Read** events, **Write** events, or both.

d. To select individual buckets, empty the **Read** and **Write** check boxes for **All current and future S3 buckets**. In **Individual bucket selection**, browse for a bucket on which to log data events. Find specific buckets by typing a bucket prefix for the bucket you want. You can select multiple buckets in this window. Choose **Add bucket** to log data events for more buckets. Choose to log **Read** events, such as `GetObject`, **Write** events, such as `PutObject`, or both.

   This setting takes precedence over individual settings you configure for individual buckets. For example, if you specify logging **Read** events for all S3 buckets, and then choose to add a specific bucket for data event logging, **Read** is already selected for the bucket you added. You cannot clear the selection. You can only configure the option for **Write**.

   To remove a bucket from logging, choose **X**.

3. To add another data type on which to log data events, choose **Add data event type**.

4. For Lambda functions:

a. For **Data event source**, choose **Lambda**.

b. In **Lambda function**, choose **All regions** to log all Lambda functions, or **Input function as ARN** to log data events on a specific function.

   To log data events for all Lambda functions in your AWS account, select **Log all current and future functions**. This setting takes precedence over individual settings you configure for individual functions. All functions are logged, even if all functions are not displayed.

   > **Note**
   >
   > If you are creating a trail for all Regions, this selection enables data event logging for all functions currently in your AWS account, and any Lambda functions you might create in any Region after you finish creating the trail. If you are creating a trail for a single Region (done by using the AWS CLI), this selection enables data event logging for all functions currently in that Region in your AWS account, and any Lambda functions you might create in that Region after you finish creating the trail. It does not enable data event logging for Lambda functions created in other Regions.
   > Logging data events for all functions also enables logging of data event activity performed by any IAM identity in your AWS account, even if that activity is performed on a function that belongs to another AWS account.

c. If you choose **Input function as ARN**, enter the ARN of a Lambda function.

   > **Note**
   >
   > If you have more than 15,000 Lambda functions in your account, you cannot view or select all functions in the CloudTrail console when creating a trail. You can still select the option to log all functions, even if they are not displayed. If you want to log data events for specific functions, you can manually add a function if you know its ARN. You can also finish creating the trail in the console, and then use the AWS CLI and the **put-event-selectors** command to configure data event logging for specific Lambda functions. For more information, see Managing trails with the AWS CLI.

5. For DynamoDB tables:

   a. For **Data event source**, choose **DynamoDB**.

   b. In **DynamoDB table selection**, choose **Browse** to select a table, or paste in the ARN of a DynamoDB table to which you have access. A DynamoDB table ARN uses the following format:

      ```
      arn:partition:dynamodb:region:account_ID:table/table_name
      ```

      To add another table, choose **Add row**, and browse for a table or paste in the ARN of a table to which you have access.

6. To configure Insights events and other settings for your trail, go back to the preceding procedure in this topic, ???.

Next steps

After you create your trail, you can return to the trail to make changes:

- If you haven't already, you can configure CloudTrail to send log files to CloudWatch Logs. For more information, see Sending events to CloudWatch Logs.
- Create a table and use it to run a query in Amazon Athena to analyze your AWS service activity. For more information, see Creating a Table for CloudTrail Logs in the CloudTrail Console in the Amazon Athena User Guide.
- Add custom tags (key-value pairs) to the trail.
- To create another trail, open the **Trails** page, and choose **Create trail**.

Updating a trail with the CloudTrail console

This section describes how to change trail settings.

To update a single-Region trail to log events in all AWS Regions in the AWS partition in which you are working, or update an multi-Region trail to log events in only a single Region, you must use the AWS CLI. For more information about how to update a single-Region trail to log events in all Regions, see Converting a trail that applies to one Region to apply to all Regions. For more information about how to update an multi-Region trail to log events in a single Region, see Converting a multi-Region trail to a single-Region trail.

If you've enabled CloudTrail management events in Amazon Security Lake, you are required to maintain at least one organizational trail that is multi-Region and logs both `read` and `write` management events. You cannot update a qualifying trail in such a way that it fails to meet the Security Lake requirement. For example, by changing the trail to single-Region, or by turning off the logging of `read` or `write` management events.

> **Note**
>
> CloudTrail updates organization trails in member accounts even if a resource validation fails. Examples of validation failures include:
>
> - an incorrect Amazon S3 bucket policy
> - an incorrect Amazon SNS topic policy
> - inability to deliver to a CloudWatch Logs log group
> - insufficient permission to encrypt using a KMS key
>
> A member account with CloudTrail permissions can see any validation failures for an organization trail by viewing the trail's details page on the CloudTrail console, or by running the AWS CLI get-trail-status command.

To update a trail with the AWS Management Console

1. Sign in to the AWS Management Console and open the CloudTrail console at https://console.aws.amazon.com/cloudtrail/.
2. In the navigation pane, choose **Trails**, and then choose a trail name.
3. In **General details**, choose **Edit** to change the following settings. You cannot change the name of a trail.

   - **Apply trail to my organization** - Change whether this trail is an AWS Organizations organization trail.

> **Note**
>
> Only the management account for the organization can convert an organization trail to a non-organization trail, or convert a non-organization trail to an organization trail.

- **Trail log location** - Change the name of the S3 bucket or prefix in which you are storing logs for this trail.
- **Log file SSE-KMS encryption** - Choose to enable or disable encrypting log files with SSE-KMS instead of SSE-S3.
- **Log file validation** - Choose to enable or disable validation of the integrity of log files.
- **SNS notification delivery** - Choose to enable or disable Amazon Simple Notification Service (Amazon SNS) notifications that log files have been delivered to the bucket specified for the trail.

a. To change the trail to an AWS Organizations organization trail, you can choose to enable the trail for all accounts in your organization. For more information, see Creating a trail for an organization.

b. To change the specified bucket in **Storage location**, choose **Create new S3 bucket** to create a bucket. When you create a bucket, CloudTrail creates and applies the required bucket policies. If you choose to create a new S3 bucket, your IAM policy needs to include permission for the `s3:PutEncryptionConfiguration` action because by default server-side encryption is enabled for the bucket.

> **Note**
>
> If you chose **Use existing S3 bucket**, specify a bucket in **Trail log bucket name**, or choose **Browse** to choose a bucket. The bucket policy must grant CloudTrail permission to write to it. For information about manually editing the bucket policy, see Amazon S3 bucket policy for CloudTrail.

To make it easier to find your logs, create a new folder (also known as a *prefix*) in an existing bucket to store your CloudTrail logs. Enter the prefix in **Prefix**.

c. For **Log file SSE-KMS encryption**, choose **Enabled** if you want to encrypt your log files using SSE-KMS encryption instead of SSE-S3 encryption. The default is **Enabled**. If you don't enable SSE-KMS encryption, your logs are encrypted using SSE-S3 encryption. For more information about SSE-KMS encryption, see Using server-side encryption with AWS Key Management Service (SSE-KMS). For more information about SSE-S3 encryption, see Using Server-Side Encryption with Amazon S3-Managed Encryption Keys (SSE-S3).

   If you enable SSE-KMS encryption, choose a **New** or **Existing** AWS KMS key. In **AWS KMS Alias**, specify an alias, in the format `alias/MyAliasName`. For more information, see Updating a resource to use your KMS key with the console. CloudTrail also supports AWS KMS multi-Region keys. For more information about multi-Region keys, see Using multi-Region keys in the *AWS Key Management Service Developer Guide*.

   > **Note**
   >
   > You can also type the ARN of a key from another account. For more information, see Updating a resource to use your KMS key with the console. The key policy must allow CloudTrail to use the key to encrypt your log files, and allow the users you specify to read log files in unencrypted form. For information about manually editing the key policy, see Configure AWS KMS key policies for CloudTrail.

d. For **Log file validation**, choose **Enabled** to have log digests delivered to your S3 bucket. You can use the digest files to verify that your log files did not change after CloudTrail delivered them. For more information, see Validating CloudTrail log file integrity.

e. For **SNS notification delivery**, choose **Enabled** to be notified each time a log is delivered to your bucket. CloudTrail stores multiple events in a log file. SNS notifications are sent for every log file, not for every event. For more information, see Configuring Amazon SNS notifications for CloudTrail.

   If you enable SNS notifications, for **Create a new SNS topic**, choose **New** to create a topic, or choose **Existing** to use an existing topic. If you are creating a trail that applies to all Regions, SNS notifications for log file deliveries from all Regions are sent to the single SNS topic that you create.

   If you choose **New**, CloudTrail specifies a name for the new topic for you, or you can type a name. If you choose **Existing**, choose an SNS topic from the drop-down list. You can also enter the ARN of a topic from another Region or from an account with appropriate permissions. For more information, see Amazon SNS topic policy for CloudTrail.

If you create a topic, you must subscribe to the topic to be notified of log file delivery. You can subscribe from the Amazon SNS console. Due to the frequency of notifications, we recommend that you configure the subscription to use an Amazon SQS queue to handle notifications programmatically. For more information, see Getting started with Amazon SNS in the *Amazon Simple Notification Service Developer Guide*.

4. In **CloudWatch Logs**, choose **Edit** to change settings for sending CloudTrail log files to CloudWatch Logs. Choose **Enabled** in **CloudWatch Logs** to enable sending log files. For more information, see Sending events to CloudWatch Logs.

   a. If you enable integration with CloudWatch Logs, choose **New** to create a new log group, or **Existing** to use an existing one. If you choose **New**, CloudTrail specifies a name for the new log group for you, or you can type a name.

   b. If you choose **Existing**, choose a log group from the drop-down list.

   c. Choose **New** to create a new IAM role for permissions to send logs to CloudWatch Logs. Choose **Existing** to choose an existing IAM role from the drop-down list. The policy statement for the new or existing role is displayed when you expand **Policy document**. For more information about this role, see Role policy document for CloudTrail to use CloudWatch Logs for monitoring.

> **Note**
>
> - When you configure a trail, you can choose an S3 bucket and SNS topic that belong to another account. However, if you want CloudTrail to deliver events to a CloudWatch Logs log group, you must choose a log group that exists in your current account.
> - Only the management account can configure a CloudWatch Logs log group for an organization trail using the console. The delegated administrator can configure a CloudWatch Logs log group using the AWS CLI or CloudTrail `CreateTrail` or `UpdateTrail` API operations.

5. In **Tags**, choose **Edit** to change, add, or delete tags on the trail. Add one or more custom tags (key-value pairs) to your trail. Tags can help you identify both your CloudTrail trails and the Amazon S3 buckets that contain CloudTrail log files. You can then use resource groups for your CloudTrail resources. For more information, see AWS Resource Groups and Tags.

6. In **Management events**, choose **Edit** to change management event logging settings.

a. For **API activity**, choose if you want your trail to log **Read** events, **Write** events, or both. For more information, see Management events.

b. Choose **Exclude AWS KMS events** to filter AWS Key Management Service (AWS KMS) events out of your trail. The default setting is to include all AWS KMS events.

   The option to log or exclude AWS KMS events is available only if you log management events on your trail. If you choose not to log management events, AWS KMS events are not logged, and you cannot change AWS KMS event logging settings.

   AWS KMS actions such as `Encrypt`, `Decrypt`, and `GenerateDataKey` typically generate a large volume (more than 99%) of events. These actions are now logged as **Read** events. Low-volume, relevant AWS KMS actions such as `Disable`, `Delete`, and `ScheduleKey` (which typically account for less than 0.5% of AWS KMS event volume) are logged as **Write** events.

   To exclude high-volume events like `Encrypt`, `Decrypt`, and `GenerateDataKey`, but still log relevant events such as `Disable`, `Delete` and `ScheduleKey`, choose to log **Write** management events, and clear the check box for **Exclude AWS KMS events**.

c. Choose **Exclude Amazon RDS Data API events** to filter Amazon Relational Database Service Data API events out of your trail. The default setting is to include all Amazon RDS Data API events. For more information about Amazon RDS Data API events, see Logging Data API calls with AWS CloudTrail in the *Amazon RDS User Guide for Aurora*.

7.

> ⚠ **Important**
>
> Steps 7-11 are for configuring data events using advanced event selectors. Advanced event selectors let you configure more data event types and offer fine-grained control over which data events your trail captures. If you are using basic event selectors, see Updating data event settings with basic event selectors, then return to step 12 of this procedure.

In **Data events**, choose **Edit** to change data event logging settings. By default, trails don't log data events. Additional charges apply for logging data events. For CloudTrail pricing, see AWS CloudTrail Pricing.

For **Data event type**, choose the resource type on which you want to log data events. For more information about available data event types, see Data events.

> **Note**
>
> To log data events for AWS Glue tables created by Lake Formation, choose **Lake Formation**.

8. Choose a log selector template. CloudTrail includes predefined templates that log all data events for the resource type. To build a custom log selector template, choose **Custom**.

> **Note**
>
> Choosing a predefined template for S3 buckets enables data event logging for all buckets currently in your AWS account and any buckets you create after you finish creating the trail. It also enables logging of data event activity performed by any user or role in your AWS account, even if that activity is performed on a bucket that belongs to another AWS account.
> If the trail applies only to one Region, choosing a predefined template that logs all S3 buckets enables data event logging for all buckets in the same Region as your trail and any buckets you create later in that Region. It will not log data events for Amazon S3 buckets in other Regions in your AWS account.
> If you are creating a trail for all Regions, choosing a predefined template for Lambda functions enables data event logging for all functions currently in your AWS account, and any Lambda functions you might create in any Region after you finish creating the trail. If you are creating a trail for a single Region (done by using the AWS CLI), this selection enables data event logging for all functions currently in that Region in your AWS account, and any Lambda functions you might create in that Region after you finish creating the trail. It does not enable data event logging for Lambda functions created in other Regions.
> Logging data events for all functions also enables logging of data event activity performed by any user or role in your AWS account, even if that activity is performed on a function that belongs to another AWS account.

9. (Optional) In **Selector name**, enter a name to identify your selector. The selector name is a descriptive name for an advanced event selector, such as "Log data events for only two S3

buckets". The selector name is listed as `Name` in the advanced event selector and is viewable if you expand the **JSON view**.

10. In **Advanced event selectors**, build an expression for the specific resources on which you want to collect data events. You can skip this step if you are using a predefined log template.

    a. Choose from the following fields.

    - **`readOnly`** - `readOnly` can be set to **equals** a value of `true` or `false`. To log both `read` and `write` events, don't add a `readOnly` selector.
    - **`eventName`** - `eventName` can use any operator. You can use it to include or exclude any data event logged to CloudTrail, such as `PutBucket` or `GetSnapshotBlock`.
    - **`resources.ARN`** - You can use any operator with `resources.ARN`, but if you use **equals** or **does not equal**, the value must exactly match the ARN of a valid resource of the type you've specified in the template as the value of `resources.type`.

      The following table shows the valid ARN format for each `resources.type`.

      Note

      You can't use the `resources.ARN` field to filter resource types that do not have ARNs.

resources.type	resources.ARN
`AWS::DynamoDB::Table` [1]	`arn:partition :dynamodb : region:account_ID :table/table_name`
`AWS::Lambda::Function`	`arn:partition :lambda:region:account_ID :function: function_name`

`AWS::S3::Object` [2]	`arn:partition :s3:::DOC-EXAMPLE-BUCKET /`
	`arn:partition :s3:::DOC-EXAMPLE-BUCKET /object_or_file_name /`

resources.type	resources.ARN
`AWS::AppConfig::Configuration`	`arn:partition :appconfig: region:account_ID :application/ application_ID /environment/ environment_ID /configuration/ configuration_profile_ID`
`AWS::B2BI::Transformer`	`arn:partition :b2bi:region:account_ID :transformer/ transformer_ID`
`AWS::Bedrock::AgentAlias`	`arn:partition :bedrock: region:account_ID :agent-alias/ agent_ID/alias_ID`
`AWS::Bedrock::FlowAlias`	`arn:partition :bedrock: region:account_ID :flow/flow_ID/alias/alias_ID`
`AWS::Bedrock::Guardrail`	`arn:partition :bedrock: region:account_ID :guardrail/ guardrail_ID`
`AWS::Bedrock::KnowledgeBase`	`arn:partition :bedrock: region:account_ID :knowledge-base/knowledge_base_ID`
`AWS::Cassandra::Table`	`arn:partition :cassandra: region:account_ID :keyspace/ keyspace_name /table/table_name`
`AWS::CloudFront::KeyValueStore`	`arn:partition :cloudfront: region:account_ID :key-value-store/KVS_name`

resources.type	resources.ARN
`AWS::CloudTrail::Channel`	`arn:partition:cloudtrail:region:account_ID:channel/channel_UUID`
`AWS::CodeWhisperer::Customization`	`arn:partition:codewhisperer:region:account_ID:customization/customization_ID`
`AWS::CodeWhisperer::Profile`	`arn:partition:codewhisperer:region:account_ID:profile/profile_ID`
`AWS::Cognito::IdentityPool`	`arn:partition:cognito-identity:region:account_ID:identitypool/identity_pool_ID`
`AWS::DynamoDB::Stream`	`arn:partition:dynamodb:region:account_ID:table/table_name/stream/date_time`
`AWS::EC2::Snapshot`	`arn:partition:ec2:region::snapshot/snapshot_ID`
`AWS::EMRWAL::Workspace`	`arn:partition:emrwal:region:account_ID:workspace/workspace_name`
`AWS::FinSpace::Environment`	`arn:partition:finspace:region:account_ID:environment/environment_ID`
`AWS::Glue::Table`	`arn:partition:glue:region:account_ID:table/database_name/table_name`

resources.type	resources.ARN
`AWS::GreengrassV2::ComponentVersion`	`arn:partition:greengrass:region:account_ID:components/component_name`
`AWS::GreengrassV2::Deployment`	`arn:partition:greengrass:region:account_ID:deployments/deployment_ID`
`AWS::GuardDuty::Detector`	`arn:partition:guardduty:region:account_ID:detector/detector_ID`
`AWS::IoT::Certificate`	`arn:partition:iot:region:account_ID:cert/certificate_ID`
`AWS::IoT::Thing`	`arn:partition:iot:region:account_ID:thing/thing_ID`
`AWS::IoTSiteWise::Asset`	`arn:partition:iotsitewise:region:account_ID:asset/asset_ID`
`AWS::IoTSiteWise::TimeSeries`	`arn:partition:iotsitewise:region:account_ID:timeseries/timeseries_ID`
`AWS::IoTTwinMaker::Entity`	`arn:partition:iottwinmaker:region:account_ID:workspace/workspace_ID/entity/entity_ID`
`AWS::IoTTwinMaker::Workspace`	`arn:partition:iottwinmaker:region:account_ID:workspace/workspace_ID`

resources.type	resources.ARN
`AWS::KendraRanking::ExecutionPlan`	arn:*partition*:kendra-ranking:*region*:*account_ID*:rescore-execution-plan/*rescore_execution_plan_ID*
`AWS::Kinesis::Stream`	arn:*partition*:kinesis:*region*:*account_ID*:stream/*stream_name*
`AWS::Kinesis::StreamConsumer`	arn:*partition*:kinesis:*region*:*account_ID*:*stream_type*/*stream_name*/consumer/*consumer_name*:*consumer_creation_timestamp*
`AWS::KinesisVideo::Stream`	arn:*partition*:kinesisvideo:*region*:*account_ID*:stream/*stream_name*/*creation_time*
`AWS::MachineLearning::MlModel`	arn:*partition*:machinelearning:*region*:*account_ID*:mlmodel/*model_ID*
`AWS::ManagedBlockchain::Network`	arn:*partition*:managedblockchain:::networks/*network_name*
`AWS::ManagedBlockchain::Node`	arn:*partition*:managedblockchain:*region*:*account_ID*:nodes/*node_ID*
`AWS::MedicalImaging::Datastore`	arn:*partition*:medical-imaging:*region*:*account_ID*:datastore/*data_store_ID*

resources.type	resources.ARN
`AWS::NeptuneGraph::Graph`	`arn:partition:neptune-graph:region:account_ID:graph/graph_ID`
`AWS::PaymentCryptography::Alias`	`arn:partition:payment-cryptography:region:account_ID:alias/alias`
`AWS::PaymentCryptography::Key`	`arn:partition:payment-cryptography:region:account_ID:key/key_ID`
`AWS::PCAConnectorAD::Connector`	`arn:partition:pca-connector-ad:region:account_ID:connector/connector_ID`
`AWS::PCAConnectorSCEP::Connector`	`arn:partition:pca-connector-scep:region:account_ID:connector/connector_ID`
`AWS::QApps:QApp`	`arn:partition:qapps:region:account_ID:application/application_UUID/qapp/qapp_UUID`
`AWS::QBusiness::Application`	`arn:partition:qbusiness:region:account_ID:application/application_ID`
`AWS::QBusiness::DataSource`	`arn:partition:qbusiness:region:account_ID:application/application_ID/index/index_ID/data-source/datasource_ID`

resources.type	resources.ARN
`AWS::QBusiness::Index`	`arn:partition:qbusiness:region:account_ID:application/application_ID/index/index_ID`
`AWS::QBusiness::WebExperience`	`arn:partition:qbusiness:region:account_ID:application/application_ID/web-experience/web_experienc_ID`
`AWS::RDS::DBCluster`	`arn:partition:rds:region:account_ID:cluster/cluster_name`
`AWS::S3::AccessPoint` [3]	`arn:partition:s3:region:account_ID:accesspoint/access_point_name`
`AWS::S3Express::Object`	`arn:partition:s3express:region:account_ID:bucket/bucket_name`
`AWS::S3ObjectLambda::AccessPoint`	`arn:partition:s3-object-lambda:region:account_ID:accesspoint/access_point_name`
`AWS::S3Outposts::Object`	`arn:partition:s3-outposts:region:account_ID:object_path`
`AWS::SageMaker::Endpoint`	`arn:partition:sagemaker:region:account_ID:endpoint/endpoint_name`

resources.type	resources.ARN
AWS::SageMaker::ExperimentTrialComponent	arn:*partition*:sagemaker:*region*:*account_ID*:experiment-trial-component/*experiment_trial_component_name*
AWS::SageMaker::FeatureGroup	arn:*partition*:sagemaker:*region*:*account_ID*:feature-group/*feature_group_name*
AWS::SCN::Instance	arn:*partition*:scn:*region*:*account_ID*:instance/*instance_ID*
AWS::ServiceDiscovery::Namespace	arn:*partition*:servicediscovery:*region*:*account_ID*:namespace/*namespace_ID*
AWS::ServiceDiscovery::Service	arn:*partition*:servicediscovery:*region*:*account_ID*:service/*service_ID*
AWS::SNS::PlatformEndpoint	arn:*partition*:sns:*region*:*account_ID*:endpoint/*endpoint_type*/*endpoint_name*/*endpoint_ID*
AWS::SNS::Topic	arn:*partition*:sns:*region*:*account_ID*:*topic_name*
AWS::SQS::Queue	arn:*partition*:sqs:*region*:*account_ID*:*queue_name*

resources.type	resources.ARN
`AWS::SSM::ManagedNode`	The ARN must be in one of the following formats: - `arn:partition:ssm:region:account_ID:managed-instance/instance_ID` - `arn:partition:ec2:region:account_ID:instance/instance_ID`
`AWS::SSMMessages::ControlChannel`	`arn:partition:ssmmessages:region:account_ID:control-channel/control_channel_ID`
`AWS::StepFunctions::StateMachine`	The ARN must be in one of the following formats: - `arn:partition:states:region:account_ID:stateMachine:stateMachine_name` - `arn:partition:states:region:account_ID:stateMachine:stateMachine_name/label_name`
`AWS::SWF::Domain`	`arn:partition:swf:region:account_ID:/domain/domain_name`
`AWS::ThinClient::Device`	`arn:partition:thinclient:region:account_ID:device/device_ID`

resources.type	resources.ARN
`AWS::ThinClient::Environment`	`arn:partition:thinclient:region:account_ID:environment/environment_ID`
`AWS::Timestream::Database`	`arn:partition:timestream:region:account_ID:database/database_name`
`AWS::Timestream::Table`	`arn:partition:timestream:region:account_ID:database/database_name/table/table_name`
`AWS::VerifiedPermissions::PolicyStore`	`arn:partition:verifiedpermissions:region:account_ID:policy-store/policy_store_ID`

[1] For tables with streams enabled, the `resources` field in the data event contains both `AWS::DynamoDB::Stream` and `AWS::DynamoDB::Table`. If you specify `AWS::DynamoDB::Table` for the `resources.type`, it will log both DynamoDB table and DynamoDB streams events by default. To exclude streams events, add a filter on the `eventName` field.

[2] To log all data events for all objects in a specific S3 bucket, use the `StartsWith` operator, and include only the bucket ARN as the matching value. The trailing slash is intentional; do not exclude it.

[3] To log events on all objects in an S3 access point, we recommend that you use only the access point ARN, don't include the object path, and use the `StartsWith` or `NotStartsWith` operators.

For more information about the ARN formats of data event resources, see Actions, resources, and condition keys in the *AWS Identity and Access Management User Guide*.

b. For each field, choose **+ Condition** to add as many conditions as you need, up to a maximum of 500 specified values for all conditions. For example, to exclude data events for two S3 buckets from data events that are logged on your trail, you can set the field to **resources.ARN**, set the operator for **does not start with**, and then either paste in an S3 bucket ARN, or browse for the S3 buckets for which you do not want to log events.

   To add the second S3 bucket, choose **+ Condition**, and then repeat the preceding instruction, pasting in the ARN for or browsing for a different bucket.

   > **Note**
   >
   > You can have a maximum of 500 values for all selectors on a trail. This includes arrays of multiple values for a selector such as `eventName`. If you have single values for all selectors, you can have a maximum of 500 conditions added to a selector.
   > If you have more than 15,000 Lambda functions in your account, you cannot view or select all functions in the CloudTrail console when creating a trail. You can still log all functions with a predefined selector template, even if they are not displayed. If you want to log data events for specific functions, you can manually add a function if you know its ARN. You can also finish creating the trail in the console, and then use the AWS CLI and the **put-event-selectors** command to configure data event logging for specific Lambda functions. For more information, see Managing trails with the AWS CLI.

c. Choose **+ Field** to add additional fields as required. To avoid errors, do not set conflicting or duplicate values for fields. For example, do not specify an ARN in one selector to be equal to a value, then specify that the ARN not equal the same value in another selector.

11. To add another data type on which to log data events, choose **Add data event type**. Repeat steps 3 through this step to configure advanced event selectors for the data event type.

12. In **Insights events** choose **Edit** if you want your trail to log CloudTrail Insights events.

    In **Event type**, select **Insights events**.

    In **Insights events**, choose **API call rate**, **API error rate**, or both. You must be logging **Write** management events to log Insights events for **API call rate**. You must be logging **Read** or **Write** management events to log Insights events for **API error rate**.

CloudTrail Insights analyzes management events for unusual activity, and logs events when anomalies are detected. By default, trails don't log Insights events. For more information about Insights events, see Logging Insights events. Additional charges apply for logging Insights events. For CloudTrail pricing, see AWS CloudTrail Pricing.

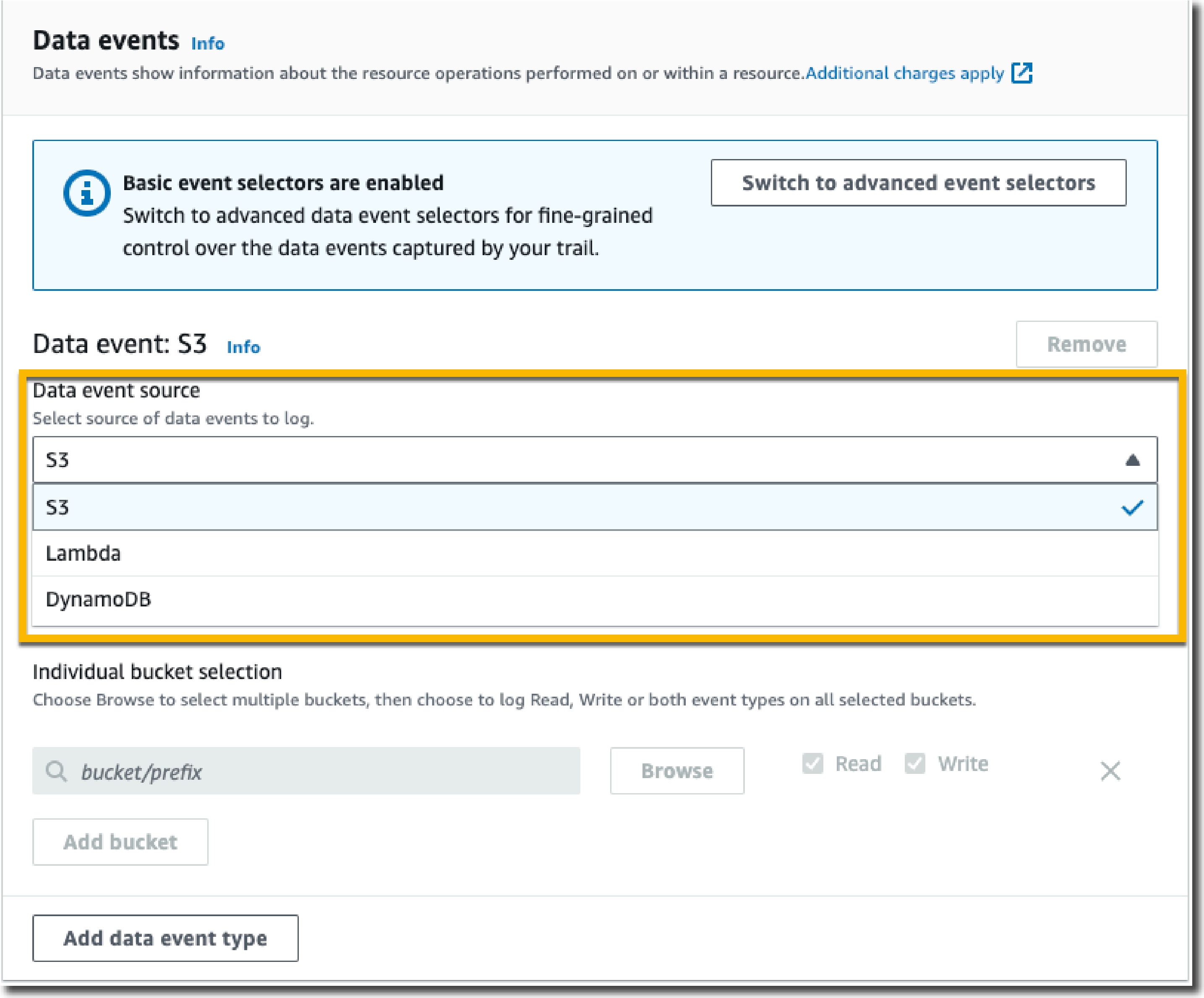
Insights events are delivered to a different folder named `/CloudTrail-Insight`of the same S3 bucket that is specified in the **Storage location** area of the trail details page. CloudTrail creates the new prefix for you. For example, if your current destination S3 bucket is named `DOC-EXAMPLE-BUCKET/AWSLogs/CloudTrail/`, the S3 bucket name with a new prefix is named `DOC-EXAMPLE-BUCKET/AWSLogs/CloudTrail-Insight/`.

13. When you are finished changing settings on your trail, choose **Update trail**.

Updating data event settings with basic event selectors

You can use advanced event selectors to configure all data event types. Advanced event selectors let you create fine-grained selectors to log only those events of interest.

If you use basic event selectors to log data events, you're limited to logging data events for Amazon S3 buckets, AWS Lambda functions, and Amazon DynamoDB tables. You can't filter on the `eventName` field using basic event selectors.



Data events Info

Data events show information about the resource operations performed on or within a resource. Additional charges apply

Basic event selectors are enabled

Switch to advanced data event selectors for fine-grained control over the data events captured by your trail.

Data event: S3 Info

Data event source

Select source of data events to log.

S3

Individual bucket selection

Choose Browse to select multiple buckets, then choose to log Read, Write or both event types on all selected buckets.

Use the following procedure to configure data event settings using basic event selectors.

1. In **Data events**, choose **Edit** to change data event logging settings. With basic event selectors, you can specify logging data events for Amazon S3 buckets, AWS Lambda functions, DynamoDBtables, or a combination of those resources. Additional data event types are supported with advanced event selectors. By default, trails don't log data events. Additional charges apply for logging data events. For more information, see Data events. For CloudTrail pricing, see AWS CloudTrail Pricing.

   For Amazon S3 buckets:

   a. For **Data event source**, choose **S3**.

b. You can choose to log **All current and future S3 buckets**, or you can specify individual buckets or functions. By default, data events are logged for all current and future S3 buckets.

> **Note**
>
> Keeping the default **All current and future S3 buckets** option enables data event logging for all buckets currently in your AWS account and any buckets you create after you finish creating the trail. It also enables logging of data event activity performed by any user or role in your AWS account, even if that activity is performed on a bucket that belongs to another AWS account.
>
> If the trail applies only to one Region, choosing **All current and future S3 buckets** enables data event logging for all buckets in the same Region as your trail and any buckets you create later in that Region. It will not log data events for Amazon S3 buckets in other Regions in your AWS account.

c. If you leave the default, **All current and future S3 buckets**, choose to log **Read** events, **Write** events, or both.

d. To select individual buckets, empty the **Read** and **Write** check boxes for **All current and future S3 buckets**. In **Individual bucket selection**, browse for a bucket on which to log data events. To find specific buckets, type a bucket prefix for the bucket you want. You can select multiple buckets in this window. Choose **Add bucket** to log data events for more buckets. Choose to log **Read** events, such as `GetObject`, **Write** events, such as `PutObject`, or both.

   This setting takes precedence over individual settings you configure for individual buckets. For example, if you specify logging **Read** events for all S3 buckets, and then choose to add a specific bucket for data event logging, **Read** is already selected for the bucket you added. You cannot clear the selection. You can only configure the option for **Write**.

   To remove a bucket from logging, choose **X**.

2. To add another data type on which to log data events, choose **Add data event type**.

3. For Lambda functions:

   a. For **Data event source**, choose **Lambda**.

   b. In **Lambda function**, choose **All regions** to log all Lambda functions, or **Input function as ARN** to log data events on a specific function.

To log data events for all Lambda functions in your AWS account, select **Log all current and future functions**. This setting takes precedence over individual settings you configure for individual functions. All functions are logged, even if all functions are not displayed.

> **Note**
>
> If you are creating a trail for all Regions, this selection enables data event logging for all functions currently in your AWS account, and any Lambda functions you might create in any Region after you finish creating the trail. If you are creating a trail for a single Region (done by using the AWS CLI), this selection enables data event logging for all functions currently in that Region in your AWS account, and any Lambda functions you might create in that Region after you finish creating the trail. It does not enable data event logging for Lambda functions created in other Regions.
> Logging data events for all functions also enables logging of data event activity performed by any user or role in your AWS account, even if that activity is performed on a function that belongs to another AWS account.

c. If you choose **Input function as ARN**, enter the ARN of a Lambda function.

> **Note**
>
> If you have more than 15,000 Lambda functions in your account, you cannot view or select all functions in the CloudTrail console when creating a trail. You can still select the option to log all functions, even if they are not displayed. If you want to log data events for specific functions, you can manually add a function if you know its ARN. You can also finish creating the trail in the console, and then use the AWS CLI and the **put-event-selectors** command to configure data event logging for specific Lambda functions. For more information, see Managing trails with the AWS CLI.

4. To add another data type on which to log data events, choose **Add data event type**.

5. For DynamoDB tables:

   a. For **Data event source**, choose **DynamoDB**.

b. In **DynamoDB table selection**, choose **Browse** to select a table, or paste in the ARN of a DynamoDB table to which you have access. A DynamoDB table ARN is in the following format:

```
arn:partition:dynamodb:region:account_ID:table/table_name
```

To add another table, choose **Add row**, and browse for a table or paste in the ARN of a table to which you have access.

6. To configure Insights events and other settings for your trail, go back to the preceding procedure in this topic, [Updating a trail with the CloudTrail console](#).

Deleting a trail with the CloudTrail console

You can delete trails with the CloudTrail console. If an organization's management account or delegated administrator account deletes an organization trail, the trail is removed from all member accounts of the organization.

If you've enabled CloudTrail management events in Amazon Security Lake, you are required to maintain at least one organizational trail that is multi-Region and logs both `read` and `write` management events. You cannot delete a trail if it is the only trail you have that meets this requirement, unless you turn off CloudTrail management events in Security Lake.

To delete a trail with the CloudTrail console

1. Sign in to the AWS Management Console and open the CloudTrail console at https://console.aws.amazon.com/cloudtrail/.
2. Open the **Trails** page of the CloudTrail console.
3. Choose the trail name.
4. At the top of the trail details page, choose **Delete**.
5. When you are prompted to confirm, choose **Delete** to delete the trail permanently. The trail is removed from the list of trails. Log files that were already delivered to the Amazon S3 bucket are not deleted and continue to incur S3 charges.

> **Note**
>
> Content delivered to Amazon S3 buckets might contain customer content. For more information about removing sensitive data, see Emptying a bucket and Deleting a bucket in the *Amazon S3 User Guide*.

Turning off logging for a trail

When you create a trail, logging is turned on automatically. You can turn off logging for a trail from the trail's details page.

> **Note**
>
> When you turn off logging, existing logs are still stored in the trail's Amazon S3 bucket and continue to incur S3 charges. For information on S3 pricing, see Amazon S3 pricing.

To turn off logging for a trail with the CloudTrail console

1. Sign in to the AWS Management Console and open the CloudTrail console at https://console.aws.amazon.com/cloudtrail/.
2. In the navigation pane, choose **Trails**, and then choose the name of the trail.
3. At the top of the trail details page, choose **Stop logging** to turn off logging for the trail.
4. When you are prompted to confirm, choose **Stop logging**. CloudTrail stops logging activity for that trail.
5. To resume logging for that trail, choose **Start logging** on the trail configuration page.

Creating, updating, and managing trails with the AWS CLI

You can use the AWS CLI to create, update, and manage your trails. When using the AWS CLI, remember that your commands run in the AWS Region configured for your profile. If you want to run the commands in a different Region, either change the default Region for your profile, or use the **--region** parameter with the command.

Note

You need the AWS command line tools to run the AWS Command Line Interface (AWS CLI) commands in this topic. Make sure you have a recent version of the AWS CLI installed. For more information, see the AWS Command Line Interface User Guide. For help with CloudTrail commands at the AWS CLI command line, type `aws cloudtrail help`.

Commonly used commands for trail creation, management, and status

Some of the more commonly used commands for creating and updating trails in CloudTrail include:

- **create-trail** to create a trail.
- **update-trail** to change the configuration of an existing trail.
- **add-tags** to add one or more tags (key-value pairs) to an existing trail.
- **remove-tags** to remove one or more tags from a trail.
- **list-tags** to return a list of tags associated with a trail.
- **put-event-selectors** to add or modify event selectors for a trail.
- **put-insight-selectors** to add or modify Insights event selectors for an existing trail, and enable or disable Insights events.
- **start-logging** to begin logging events with your trail.
- **stop-logging** to pause logging events with your trail.
- **delete-trail** to delete a trail. This command does not delete the Amazon S3 bucket that contains the log files for that trail, if any.
- **describe-trails** to return information about trails in an AWS Region.
- **get-trail** to return settings information for a trail.
- **get-trail-status** to return information about the current status of a trail.
- **get-event-selectors** to return information about event selectors configured for a trail.
- **get-insight-selectors** to return information about Insights event selectors configured for a trail.

Supported commands for creating and updating trails: create-trail and update-trail

The `create-trail` and `update-trail` commands offer a variety of functionality for creating and managing trails, including:

- Creating a trail that receives logs across Regions, or update a trail with the `--is-multi-region-trail` option. In most circumstances, you should create trails that log events in all AWS Regions.
- Creating a trail that receives logs for all AWS accounts in an organization with the **--is-organization-trail** option.
- Converting a multi-Region trail to single-Region trail with the `--no-is-multi-region-trail` option.
- Enabling or disabling log file encryption with the `--kms-key-id` option. The option specifies an AWS KMS key that you already created and to which you have attached a policy that allows CloudTrail to encrypt your logs. For more information, see Enabling and disabling CloudTrail log file encryption with the AWS CLI.
- Enabling or disabling log file validation with the `--enable-log-file-validation` and `--no-enable-log-file-validation` options. For more information, see Validating CloudTrail log file integrity.
- Specifying a CloudWatch Logs log group and role so that CloudTrail can deliver events to a CloudWatch Logs log group. For more information, see Monitoring CloudTrail Log Files with Amazon CloudWatch Logs.

Deprecated commands: create-subscription and update-subscription

⚠ **Important**

The `create-subscription` and `update-subscription` commands were used to create and update trails, but are deprecated. Do not use these commands. They do not provide full functionality for creating and managing trails.

If you configured automation that uses one or both of these commands, we recommend that you update your code or scripts to use supported commands such as **create-trail**.

Using the `create-trail` command to create a trail

You can run the `create-trail` command to create trails that are specifically configured to meet your business needs. When using the AWS CLI, remember that your commands run in the AWS Region configured for your profile. If you want to run the commands in a different Region, either change the default Region for your profile, or use the **--region** parameter with the command.

Creating a trail that applies to all Regions

To create a trail that applies to all Regions, use the `--is-multi-region-trail` option. By default, the `create-trail` command creates a trail that logs events only in the AWS Region where the trail was created. To ensure that you log global service events and capture all management event activity in your AWS account, you should create trails that log events in all AWS Regions.

> **Note**
>
> When you create a trail, if you specify an Amazon S3 bucket that was not created with CloudTrail, you need to attach the appropriate policy. See Amazon S3 bucket policy for CloudTrail.

The following example creates a trail with the name *`my-trail`* and a tag with a key named *`Group`* with a value of *`Marketing`* that delivers logs from all Regions to an existing bucket named *`DOC-EXAMPLE-BUCKET`*.

```
aws cloudtrail create-trail --name my-trail --s3-bucket-name DOC-EXAMPLE-BUCKET --is-multi-region-trail --tags-list [key=Group,value=Marketing]
```

To confirm that your trail exists in all Regions, the `IsMultiRegionTrail` element in the output shows `true`.

```
{
    "IncludeGlobalServiceEvents": true,
    "Name": "my-trail",
    "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/my-trail",
    "LogFileValidationEnabled": false,
    "IsMultiRegionTrail": true,
    "IsOrganizationTrail": false,
    "S3BucketName": "DOC-EXAMPLE-BUCKET"
}
```

> **Note**
>
> Use the `start-logging` command to start logging for your trail.

Start logging for the trail

After the `create-trail` command completes, run the `start-logging` command to start logging for that trail.

> **Note**
>
> When you create a trail with the CloudTrail console, logging is turned on automatically.

The following example starts logging for a trail.

```
aws cloudtrail start-logging --name my-trail
```

This command doesn't return an output, but you can use the `get-trail-status` command to verify that logging has started.

```
aws cloudtrail get-trail-status --name my-trail
```

To confirm that the trail is logging, the `IsLogging` element in the output shows `true`.

```
{
    "LatestDeliveryTime": 1441139757.497,
    "LatestDeliveryAttemptTime": "2015-09-01T20:35:57Z",
    "LatestNotificationAttemptSucceeded": "2015-09-01T20:35:57Z",
    "LatestDeliveryAttemptSucceeded": "2015-09-01T20:35:57Z",
    "IsLogging": true,
    "TimeLoggingStarted": "2015-09-01T00:54:02Z",
    "StartLoggingTime": 1441068842.76,
    "LatestDigestDeliveryTime": 1441140723.629,
    "LatestNotificationAttemptTime": "2015-09-01T20:35:57Z",
    "TimeLoggingStopped": ""
}
```

Creating a single-Region trail

The following command creates a single-Region trail. The specified Amazon S3 bucket must already exist and have the appropriate CloudTrail permissions applied. For more information, see Amazon S3 bucket policy for CloudTrail.

```
aws cloudtrail create-trail --name my-trail --s3-bucket-name DOC-EXAMPLE-BUCKET
```

The following is example output.

```
{
    "IncludeGlobalServiceEvents": true,
    "Name": "my-trail",
    "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/my-trail",
    "LogFileValidationEnabled": false,
    "IsMultiRegionTrail": false,
    "IsOrganizationTrail": false,
    "S3BucketName": "DOC-EXAMPLE-BUCKET"
}
```

Creating a trail that applies to all Regions and that has log file validation enabled

To enable log file validation when using `create-trail`, use the `--enable-log-file-validation` option.

For information about log file validation, see Validating CloudTrail log file integrity.

The following example creates a trail that delivers logs from all Regions to the specified bucket. The command uses the `--enable-log-file-validation` option.

```
aws cloudtrail create-trail --name my-trail --s3-bucket-name DOC-EXAMPLE-BUCKET --is-
multi-region-trail --enable-log-file-validation
```

To confirm that log file validation is enabled, the `LogFileValidationEnabled` element in the output shows `true`.

```
{
    "IncludeGlobalServiceEvents": true,
    "Name": "my-trail",
    "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/my-trail",
    "LogFileValidationEnabled": true,
    "IsMultiRegionTrail": true,
    "IsOrganizationTrail": false,
    "S3BucketName": "DOC-EXAMPLE-BUCKET"
}
```

Using the `update-trail` command to update a trail

Important

As of November 22, 2021, AWS CloudTrail changed how trails capture global service events. Now, events created by Amazon CloudFront, AWS Identity and Access Management, and AWS STS are recorded in the Region in which they were created, the US East (N. Virginia) Region, us-east-1. This makes how CloudTrail treats these services consistent with that of other AWS global services. To continue receiving global service events outside of US East (N. Virginia), be sure to convert *single-Region trails* using global service events outside of US East (N. Virginia) into *multi-Region trails*. For more information about capturing global service events, see Enabling and disabling global service event logging later in this section. In contrast, the **Event history** in the CloudTrail console and the **aws cloudtrail lookup-events** command will show these events in the AWS Region where they occurred.

You can use the `update-trail` command to change the configuration settings for a trail. You can also use the **add-tags** and **remove-tags** commands to add and remove tags for a trail. You can only update trails from the AWS Region where the trail was created (its Home Region). When using the AWS CLI, remember that your commands run in the AWS Region configured for your profile. If you want to run the commands in a different Region, either change the default Region for your profile, or use the **--region** parameter with the command.

If you've enabled CloudTrail management events in Amazon Security Lake, you are required to maintain at least one organizational trail that is multi-Region and logs both `read` and `write` management events. You cannot update a qualifying trail in such a way that it fails to meet the Security Lake requirement. For example, by changing the trail to single-Region, or by turning off the logging of `read` or `write` management events.

Note

If you use the AWS CLI or one of the AWS SDKs to modify a trail, be sure that the trail's bucket policy is up-to-date. In order for your bucket to automatically receive events from a new AWS Region, the policy must contain the full service name, `cloudtrail.amazonaws.com`. For more information, see Amazon S3 bucket policy for CloudTrail.

Topics

Converting a trail that applies to one Region to apply to all Regions

To change an existing trail so that it applies to all Regions, use the `--is-multi-region-trail` option.

```
aws cloudtrail update-trail --name my-trail --is-multi-region-trail
```

To confirm that the trail now applies to all Regions, the `IsMultiRegionTrail` element in the output shows `true`.

```
{
    "IncludeGlobalServiceEvents": true,
    "Name": "my-trail",
    "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/my-trail",
    "LogFileValidationEnabled": false,
    "IsMultiRegionTrail": true,
    "IsOrganizationTrail": false,
    "S3BucketName": "DOC-EXAMPLE-BUCKET"
}
```

Converting a multi-Region trail to a single-Region trail

To change an existing multi-Region trail so that it applies only to the Region in which it was created, use the `--no-is-multi-region-trail` option.

```
aws cloudtrail update-trail --name my-trail --no-is-multi-region-trail
```

To confirm that the trail now applies to a single Region, the `IsMultiRegionTrail` element in the output shows `false`.

```
{
    "IncludeGlobalServiceEvents": true,
    "Name": "my-trail",
    "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/my-trail",
    "LogFileValidationEnabled": false,
    "IsMultiRegionTrail": false,
    "IsOrganizationTrail": false,
    "S3BucketName": "DOC-EXAMPLE-BUCKET"
}
```

Enabling and disabling global service event logging

To change a trail so that it does not log global service events, use the `--no-include-global-service-events` option.

```
aws cloudtrail update-trail --name my-trail --no-include-global-service-events
```

To confirm that the trail no longer logs global service events, the `IncludeGlobalServiceEvents` element in the output shows `false`.

```
{
    "IncludeGlobalServiceEvents": false,
    "Name": "my-trail",
    "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/my-trail",
    "LogFileValidationEnabled": false,
    "IsMultiRegionTrail": false,
    "IsOrganizationTrail": false,
    "S3BucketName": "DOC-EXAMPLE-BUCKET"
}
```

To change a trail so that it logs global service events, use the `--include-global-service-events` option.

Single-Region trails will no longer receive global service events beginning November 22, 2021, unless the trail already appears in US East (N. Virginia) Region, us-east-1. To continue capturing global service events, update the trail configuration to a multi-Region trail. For example, this command updates a single-Region trail in US East (Ohio), us-east-2, into a multi-Region trail. Replace *`myExistingSingleRegionTrailWithGSE`* with the appropriate trail name for your configuration.

```
aws cloudtrail --region us-east-2 update-trail --
name myExistingSingleRegionTrailWithGSE --is-multi-region-trail
```

Because global service events are only available in US East (N. Virginia) beginning November 22, 2021, you can also create a single-Region trail to subscribe to global service events in the US East (N. Virginia) Region, us-east-1. The following command creates a single-Region trail in us-east-1 to receive CloudFront, IAM, and AWS STS events:

```
aws cloudtrail --region us-east-1 create-trail --include-global-service-events --
name myTrail --s3-bucket-name DOC-EXAMPLE-BUCKET
```

Enabling log file validation

To enable log file validation for a trail, use the `--enable-log-file-validation` option. Digest files are delivered to the Amazon S3 bucket for that trail.

```
aws cloudtrail update-trail --name my-trail --enable-log-file-validation
```

To confirm that log file validation is enabled, the `LogFileValidationEnabled` element in the output shows `true`.

```
{
    "IncludeGlobalServiceEvents": true,
    "Name": "my-trail",
    "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/my-trail",
    "LogFileValidationEnabled": true,
    "IsMultiRegionTrail": false,
    "IsOrganizationTrail": false,
    "S3BucketName": "DOC-EXAMPLE-BUCKET"
}
```

Disabling log file validation

To disable log file validation for a trail, use the `--no-enable-log-file-validation` option.

```
aws cloudtrail update-trail --name my-trail-name --no-enable-log-file-validation
```

To confirm that log file validation is disabled, the `LogFileValidationEnabled` element in the output shows `false`.

```
{
    "IncludeGlobalServiceEvents": true,
    "Name": "my-trail",
    "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/my-trail",
    "LogFileValidationEnabled": false,
    "IsMultiRegionTrail": false,
    "IsOrganizationTrail": false,
    "S3BucketName": "DOC-EXAMPLE-BUCKET"
}
```

To validate log files with the AWS CLI, see Validating CloudTrail log file integrity with the AWS CLI.

Managing trails with the AWS CLI

The AWS CLI includes several other commands that help you manage your trails. These commands add tags to trails, get trail status, start and stop logging for trails, and delete a trail. You must run these commands from the same AWS Region where the trail was created (its Home Region). When using the AWS CLI, remember that your commands run in the AWS Region configured for your profile. If you want to run the commands in a different Region, either change the default Region for your profile, or use the **--region** parameter with the command.

Topics

Add one or more tags to a trail

To add one or more tags to an existing trail, run the **add-tags** command.

The following example adds a tag with the name *Owner* and the value of *Mary* to a trail with the ARN of *arn:aws:cloudtrail:us-east-2:123456789012:trail/my-trail* in the US East (Ohio) Region.

```
aws cloudtrail add-tags --resource-id arn:aws:cloudtrail:us-east-2:123456789012:trail/my-trail --tags-list Key=Owner,Value=Mary --region us-east-2
```

If successful, this command returns nothing.

List tags for one or more trails

To view the tags associated with one or more existing trails, use the **list-tags** command.

The following example lists the tags for *`Trail1`* and *`Trail2`*.

```
aws cloudtrail list-tags --resource-id-list arn:aws:cloudtrail:us-east-2:123456789012:trail/Trail1 arn:aws:cloudtrail:us-east-2:123456789012:trail/Trail2
```

If successful, this command returns output similar to the following.

```
{
 "ResourceTagList": [
     {
         "ResourceId": "arn:aws:cloudtrail:us-east-2:123456789012:trail/Trail1",
         "TagsList": [
             {
                 "Value": "Alice",
                 "Key": "Name"
             },
             {
                 "Value": "Ohio",
                 "Key": "Location"
             }
         ]
     },
     {
         "ResourceId": "arn:aws:cloudtrail:us-east-2:123456789012:trail/Trail2",
         "TagsList": [
             {
                 "Value": "Bob",
                 "Key": "Name"
             }
         ]
     }
  ]
}
```

Remove one or more tags from a trail

To remove one or more tags from an existing trail, run the **remove-tags** command.

The following example removes tags with the names *Location* and *Name* from a trail with the ARN of *arn:aws:cloudtrail:us-east-2:123456789012:trail/Trail1* in the US East (Ohio) Region.

```
aws cloudtrail remove-tags --resource-id arn:aws:cloudtrail:us-
east-2:123456789012:trail/Trail1 --tags-list Key=Name Key=Location --region us-east-2
```

If successful, this command returns nothing.

Retrieving trail settings and the status of a trail

Run the `describe-trails` command to retrieve information about trails in an AWS Region. The following example returns information about trails configured in the US East (Ohio) Region.

```
aws cloudtrail describe-trails --region us-east-2
```

If the command succeeds, you see output similar to the following.

```
{
  "trailList": [
    {
      "Name": "my-trail",
      "S3BucketName": "DOC-EXAMPLE-BUCKET1",
      "S3KeyPrefix": "my-prefix",
      "IncludeGlobalServiceEvents": true,
      "IsMultiRegionTrail": true,
      "HomeRegion": "us-east-2"
      "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/my-trail",
      "LogFileValidationEnabled": false,
      "HasCustomEventSelectors": false,
      "SnsTopicName": "my-topic",
      "IsOrganizationTrail": false,
    },
    {
      "Name": "my-special-trail",
      "S3BucketName": "DOC-EXAMPLE-BUCKET2",
      "S3KeyPrefix": "example-prefix",
      "IncludeGlobalServiceEvents": false,
      "IsMultiRegionTrail": false,
      "HomeRegion": "us-east-2",
      "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/my-special-trail",
      "LogFileValidationEnabled": false,
      "HasCustomEventSelectors": true,
      "IsOrganizationTrail": false
    },
    {
      "Name": "my-org-trail",
      "S3BucketName": "DOC-EXAMPLE-BUCKET3",
      "S3KeyPrefix": "my-prefix",
      "IncludeGlobalServiceEvents": true,
      "IsMultiRegionTrail": true,
      "HomeRegion": "us-east-1"
      "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/my-org-trail",
      "LogFileValidationEnabled": false,
      "HasCustomEventSelectors": false,
      "SnsTopicName": "my-topic",
      "IsOrganizationTrail": true
    }
  ]
}
```

Run the `get-trail` command to retrieve settings information about a specific trail. The following example returns settings information for a trail named *my-trail*.

```
aws cloudtrail get-trail - -name my-trail
```

If successful, this command returns output similar to the following.

```
{
   "Trail": {
      "Name": "my-trail",
      "S3BucketName": "DOC-EXAMPLE-BUCKET",
      "S3KeyPrefix": "my-prefix",
      "IncludeGlobalServiceEvents": true,
      "IsMultiRegionTrail": true,
      "HomeRegion": "us-east-2"
      "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/my-trail",
      "LogFileValidationEnabled": false,
      "HasCustomEventSelectors": false,
      "SnsTopicName": "my-topic",
      "IsOrganizationTrail": false,
   }
}
```

Run the `get-trail-status` command to retrieve the status of a trail. You must either run this command from the AWS Region where it was created (the Home Region), or you must specify that Region by adding the **--region** parameter.

> **Note**
>
> If the trail is an organization trail and you are a member account in the organization in AWS Organizations, you must provide the full ARN of that trail, and not just the name.

```
aws cloudtrail get-trail-status --name my-trail
```

If the command succeeds, you see output similar to the following.

```
{
    "LatestDeliveryTime": 1441139757.497,
    "LatestDeliveryAttemptTime": "2015-09-01T20:35:57Z",
    "LatestNotificationAttemptSucceeded": "2015-09-01T20:35:57Z",
    "LatestDeliveryAttemptSucceeded": "2015-09-01T20:35:57Z",
    "IsLogging": true,
    "TimeLoggingStarted": "2015-09-01T00:54:02Z",
    "StartLoggingTime": 1441068842.76,
    "LatestDigestDeliveryTime": 1441140723.629,
    "LatestNotificationAttemptTime": "2015-09-01T20:35:57Z",
    "TimeLoggingStopped": ""
}
```

In addition to the fields shown in the preceding JSON code, the status contains the following fields if there are Amazon SNS or Amazon S3 errors:

- `LatestNotificationError`. Contains the error emitted by Amazon SNS if a subscription to a topic fails.
- `LatestDeliveryError`. Contains the error emitted by Amazon S3 if CloudTrail cannot deliver a log file to a bucket.

Configuring CloudTrail Insights event selectors

Enable Insights events on a trail by running the **put-insight-selectors**, and specifying `ApiCallRateInsight`, `ApiErrorRateInsight`, or both as the value of the `InsightType` attribute. To view the Insights selector settings for a trail, run the `get-insight-selectors` command. You must either run this command from the AWS Region where the trail was created (the Home Region), or you must specify that Region by adding the **--region** parameter to the command.

> **Note**
>
> To log Insights events for `ApiCallRateInsight`, the trail must log `write` management events. To log Insights events for `ApiErrorRateInsight`, the trail must log `read` or `write` management events.

Example trail that logs Insights events

The following example uses **put-insight-selectors** to create an Insights event selector for a trail named *`TrailName3`*. This enables Insights event collection for the *`TrailName3`* trail. The Insights event selector logs both `ApiErrorRateInsight` and `ApiCallRateInsight` Insights event types.

```
aws cloudtrail put-insight-selectors --trail-name TrailName3 --insight-selectors
 '[{"InsightType": "ApiCallRateInsight"},{"InsightType": "ApiErrorRateInsight"}]'
```

The example returns the Insights event selector that is configured for the trail.

```
{
   "InsightSelectors":
      [
         {
            "InsightType": "ApiErrorRateInsight"
         },
         {
            "InsightType": "ApiCallRateInsight"
         }
      ],
   "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/TrailName3"
}
```

Example: Turn off collection of Insights events

The following example uses **put-insight-selectors** to remove the Insights event selector for a trail named *TrailName3*. Clearing the JSON string of Insights selectors disables Insights event collection for the *TrailName3* trail.

```
aws cloudtrail put-insight-selectors --trail-name TrailName3 --insight-selectors '[]'
```

The example returns the now-empty Insights event selector that is configured for the trail.

```
{
   "InsightSelectors": [ ],
   "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/TrailName3"
}
```

Configuring advanced event selectors

You can use advanced event selectors to log management events and data events for all resource types. In contrast, you can use basic event selectors to log management events and data events for the `AWS::DynamoDB::Table`, `AWS::Lambda::Function`, and `AWS::S3::Object` resource types. You can use either basic event selectors, or advanced event selectors, but not both. If you apply advanced event selectors to a trail that is using basic event selectors, the basic event selectors are overwritten.

To convert a trail to advanced event selectors, run the **get-event-selectors** command to confirm the current event selectors, and then configure the advanced event selectors to match the coverage of the previous event selectors, then add any additional selectors.

You must either run the `get-event-selectors` command from the AWS Region where the trail was created (the Home Region), or you must specify that Region by adding the **--region** parameter.

```
aws cloudtrail get-event-selectors --trail-name TrailName
```

> **Note**
>
> If the trail is an organization trail, and you are signed in with a member account in the organization in AWS Organizations, you must provide the full ARN of the trail, and not just the name.

The following example shows the settings for a trail that is using advanced event selectors to log management events. By default, a trail is configured to log all management events and no data events.

```
{
    "TrailARN": "arn:aws:cloudtrail:us-east-1:123456789012:trail/management-events-
trail",
    "AdvancedEventSelectors": [
        {
            "Name": "Management events selector",
            "FieldSelectors": [
                {
                    "Field": "eventCategory",
                    "Equals": [
                        "Management"
                    ]
                }
            ]
        }
    ]
}
```

To create an advanced event selector, run the `put-event-selectors` command. When an event occurs in your account, CloudTrail evaluates the configuration for your trails. If the event matches any advanced event selector for a trail, the trail processes and logs the event. You can configure up to 500 conditions on a trail, including all values specified for all advanced event selectors on your trail. For more information, see Logging data events.

Topics

Example trail with specific advanced event selectors

The following example creates custom advanced event selectors for a trail named *`TrailName`* to include read and write management events (by omitting the `readOnly` selector), `PutObject` and `DeleteObject` data events for all Amazon S3 bucket/prefix combinations except for a bucket named `DOC-EXAMPLE-BUCKET`, and data events for an AWS Lambda function named `MyLambdaFunction`. Because these are custom advanced event selectors, each set of selectors has a descriptive name. Note that a trailing slash is part of the ARN value for S3 buckets.

```
aws cloudtrail put-event-selectors --trail-name TrailName --advanced-event-selectors
'[
  {
    "Name": "Log readOnly and writeOnly management events",
    "FieldSelectors": [
      { "Field": "eventCategory", "Equals": ["Management"] }
    ]
  },
  {
    "Name": "Log PutObject and DeleteObject events for all but one bucket",
    "FieldSelectors": [
      { "Field": "eventCategory", "Equals": ["Data"] },
      { "Field": "resources.type", "Equals": ["AWS::S3::Object"] },
      { "Field": "eventName", "Equals": ["PutObject","DeleteObject"] },
      { "Field": "resources.ARN", "NotStartsWith": ["arn:aws:s3:::DOC-EXAMPLE-
BUCKET/"] }
    ]
  },
  {
    "Name": "Log data plane actions on MyLambdaFunction",
    "FieldSelectors": [
      { "Field": "eventCategory", "Equals": ["Data"] },
      { "Field": "resources.type", "Equals": ["AWS::Lambda::Function"] },
      { "Field": "resources.ARN", "Equals": ["arn:aws:lambda:us-
east-2:111122223333:function/MyLambdaFunction"] }
    ]
  }
]'
```

The example returns the advanced event selectors that are configured for the trail.

```
{
  "AdvancedEventSelectors": [
    {
      "Name": "Log readOnly and writeOnly management events",
      "FieldSelectors": [
        {
          "Field": "eventCategory",
          "Equals": [ "Management" ]
        }
      ]
    },
    {
      "Name": "Log PutObject and DeleteObject events for all but one bucket",
      "FieldSelectors": [
        {
          "Field": "eventCategory",
          "Equals": [ "Data" ]
        },
        {
          "Field": "resources.type",
          "Equals": [ "AWS::S3::Object" ]
        },
        {
          "Field": "resources.ARN",
          "NotStartsWith": [ "arn:aws:s3:::DOC-EXAMPLE-BUCKET/" ]
        },
      ]
    },
    {
      "Name": "Log data plane actions on MyLambdaFunction",
      "FieldSelectors": [
        {
          "Field": "eventCategory",
          "Equals": [ "Data" ]
        },
        {
          "Field": "resources.type",
          "Equals": [ "AWS::Lambda::Function" ]
        },
        {
          "Field": "eventName",
          "Equals": [ "Invoke" ]
        },
        {
          "Field": "resources.ARN",
          "Equals": [ "arn:aws:lambda:us-east-2:123456789012:function/
MyLambdaFunction" ]
        }
      ]
    }
  ],
  "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/TrailName"
}
```

Example trail that uses custom advanced event selectors to log Amazon S3 on AWS Outposts data events

The following example shows how to configure your trail to include all data events for all Amazon S3 on AWS Outposts objects in your outpost. In this release, the supported value for S3 on AWS Outposts events for the `resources.type` field is `AWS::S3Outposts::Object`.

```
aws cloudtrail put-event-selectors --trail-name TrailName --region region \
--advanced-event-selectors \
'[
    {
            "Name": "OutpostsEventSelector",
            "FieldSelectors": [
                { "Field": "eventCategory", "Equals": ["Data"] },
                { "Field": "resources.type", "Equals": ["AWS::S3Outposts::Object"] }
            ]
        }
]'
```

The command returns the following example output.

```
{
    "AdvancedEventSelectors": [
        {
            "Name": "OutpostsEventSelector",
            "FieldSelectors": [
                {
                    "Field": "eventCategory",
                    "Equals": [
                        "Data"
                    ]
                },
                {
                    "Field": "resources.type",
                    "Equals": [
                        "AWS::S3Outposts::Object"
                    ]
                }
            ]
        }
    ],
  "TrailARN": "arn:aws:cloudtrail:region:123456789012:trail/TrailName"
}
```

Example trail that uses advanced event selectors to exclude AWS Key Management Service events

The following example creates an advanced event selector for a trail named *TrailName* to include read-only and write-only management events (by omitting the `readOnly` selector), but to exclude AWS Key Management Service (AWS KMS) events. Because AWS KMS events are treated as management events, and there can be a high volume of them, they can have a substantial impact on your CloudTrail bill if you have more than one trail that captures management events.

If you choose not to log management events, AWS KMS events are not logged, and you cannot change AWS KMS event logging settings.

To start logging AWS KMS events to a trail again, remove the `eventSource` selector, and run the command again.

```
aws cloudtrail put-event-selectors --trail-name TrailName \
--advanced-event-selectors '
[
  {
    "Name": "Log all management events except KMS events",
    "FieldSelectors": [
      { "Field": "eventCategory", "Equals": ["Management"] },
      { "Field": "eventSource", "NotEquals": ["kms.amazonaws.com"] }
    ]
  }
]'
```

The example returns the advanced event selectors that are configured for the trail.

```
{
  "AdvancedEventSelectors": [
    {
      "Name": "Log all management events except KMS events",
      "FieldSelectors": [
        {
          "Field": "eventCategory",
          "Equals": [ "Management" ]
        },
        {
          "Field": "eventSource",
          "NotEquals": [ "kms.amazonaws.com" ]
        }
      ]
    }
  ],
  "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/TrailName"
}
```

To start logging excluded events to a trail again, remove the `eventSource` selector, as shown in the following command.

```
aws cloudtrail put-event-selectors --trail-name TrailName \
--advanced-event-selectors '
[
  {
    "Name": "Log all management events",
    "FieldSelectors": [
      { "Field": "eventCategory", "Equals": ["Management"] }
    ]
  }
]'
```

Example trail that uses advanced event selectors to exclude Amazon RDS Data API management events

The following example creates an advanced event selector for a trail named *`TrailName`* to include read-only and write-only management events (by omitting the `readOnly` selector), but to exclude Amazon RDS Data API management events. To exclude Amazon RDS Data API management events, specify the Amazon RDS Data API event source in the string value for the `eventSource` field: `rdsdata.amazonaws.com`.

If you choose not to log management events, Amazon RDS Data API management events are not logged, and you cannot change Amazon RDS Data API event logging settings.

To start logging Amazon RDS Data API management events to a trail again, remove the `eventSource` selector, and run the command again.

```
aws cloudtrail put-event-selectors --trail-name TrailName \
--advanced-event-selectors '
[
  {
    "Name": "Log all management events except Amazon RDS Data API management events",
    "FieldSelectors": [
      { "Field": "eventCategory", "Equals": ["Management"] },
      { "Field": "eventSource", "NotEquals": ["rdsdata.amazonaws.com"] }
    ]
  }
]'
```

The example returns the advanced event selectors that are configured for the trail.

```
{
  "AdvancedEventSelectors": [
    {
      "Name": "Log all management events except Amazon RDS Data API management events",
      "FieldSelectors": [
        {
          "Field": "eventCategory",
          "Equals": [ "Management" ]
        },
        {
          "Field": "eventSource",
          "NotEquals": [ "rdsdata.amazonaws.com" ]
        }
      ]
    }
  ],
  "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/TrailName"
}
```

To start logging excluded events to a trail again, remove the `eventSource` selector, as shown in the following command.

```
aws cloudtrail put-event-selectors --trail-name TrailName \
--advanced-event-selectors '
[
  {
    "Name": "Log all management events",
    "FieldSelectors": [
      { "Field": "eventCategory", "Equals": ["Management"] }
    ]
  }
]'
```

Configuring basic event selectors

You can only use basic event selectors to log management events and data events for the `AWS::DynamoDB::Table`, `AWS::Lambda::Function`, and `AWS::S3::Object` resource types. You can log management events and data events for all resource types by using advanced event selectors.

You can use either basic event selectors, or advanced event selectors, but not both. If you apply basic event selectors to a trail that is using advanced event selectors, the advanced event selectors are overwritten.

To view the event selector settings for a trail, run the `get-event-selectors` command. You must either run this command from the AWS Region where it was created (the Home Region), or you must specify that Region by using the **--region** parameter.

```
aws cloudtrail get-event-selectors --trail-name TrailName
```

> **Note**
>
> If the trail is an organization trail and you are a member account in the organization in AWS Organizations, you must provide the full ARN of that trail, and not just the name.

The following example shows the settings for a trail that is using basic event selectors to log management events.

```
{
    "EventSelectors": [
        {
            "ExcludeManagementEventSources": [],
            "IncludeManagementEvents": true,
            "DataResources": [],
            "ReadWriteType": "All"
        }
    ],
    "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/TrailName"
}
```

To create an event selector, run the `put-event-selectors` command. If you want to log Insights events on the trail, be sure the event selector enables logging of the Insights types you want configured your trail. For more information about logging Insights events, see Logging Insights events.

When an event occurs in your account, CloudTrail evaluates the configuration for your trails. If the event matches any event selector for a trail, the trail processes and logs the event. You can configure up to 5 event selectors for a trail and up to 250 data resources for a trail. For more information, see Logging data events.

Topics

Example trail with specific event selectors

The following example creates an event selector for a trail named *TrailName* to include read-only and write-only management events, data events for two Amazon S3 bucket/prefix combinations, and data events for a single AWS Lambda function named *hello-world-python-function*.

```
aws cloudtrail put-event-selectors --trail-name TrailName --event-
selectors '[{"ReadWriteType": "All","IncludeManagementEvents":
 true,"DataResources": [{"Type":"AWS::S3::Object", "Values": ["arn:aws:s3:::DOC-
EXAMPLE-BUCKET/prefix","arn:aws:s3:::DOC-EXAMPLE-BUCKET2/prefix2"]},
{"Type": "AWS::Lambda::Function","Values": ["arn:aws:lambda:us-
west-2:999999999999:function:hello-world-python-function"]}]}]'
```

The example returns the event selector configured for the trail.

```
{
    "EventSelectors": [
        {
            "ExcludeManagementEventSources": [],
            "IncludeManagementEvents": true,
            "DataResources": [
                {
                    "Values": [
                        "arn:aws:s3:::DOC-EXAMPLE-BUCKET/prefix",
                        "arn:aws:s3:::DOC-EXAMPLE-BUCKET2/prefix2"
                    ],
                    "Type": "AWS::S3::Object"
                },
                {
                    "Values": [
                        "arn:aws:lambda:us-west-2:123456789012:function:hello-world-
python-function"
                    ],
                    "Type": "AWS::Lambda::Function"
                },
            ],
            "ReadWriteType": "All"
        }
    ],
    "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/TrailName"
}
```

Example trail that logs all management and data events

The following example creates an event selector for a trail named *TrailName2* that includes all management events, including read-only and write-only management events, and data events for all Amazon S3 buckets, AWS Lambda functions, and Amazon DynamoDB tables in the AWS account. Because this example uses basic event selectors, it cannot configure logging for S3 events on AWS Outposts, Amazon Managed Blockchain JSON-RPC calls on Ethereum nodes, or other advanced event selector resource types. You must use advanced event selectors to log data events for all other resource types. For more information, see Configuring advanced event selectors.

Note

If the trail applies only to one Region, only events in that Region are logged, even though the event selector parameters specify all Amazon S3 buckets and Lambda functions. Event selectors apply only to the Regions where the trail is created.

```
aws cloudtrail put-event-selectors --trail-name TrailName2 --event-selectors
 '[{"ReadWriteType": "All","IncludeManagementEvents": true,"DataResources":
 [{"Type":"AWS::S3::Object", "Values": ["arn:aws:s3:::"]},{"Type":
 "AWS::Lambda::Function","Values": ["arn:aws:lambda"]},{"Type":
 "AWS::DynamoDB::Table","Values": ["arn:aws:dynamodb"]}]}]'
```

The example returns the event selectors configured for the trail.

```
{
    "EventSelectors": [
        {
            "ExcludeManagementEventSources": [],
            "IncludeManagementEvents": true,
            "DataResources": [
                {
                    "Values": [
                        "arn:aws:s3:::"
                    ],
                    "Type": "AWS::S3::Object"
                },
                {
                    "Values": [
                        "arn:aws:lambda"
                    ],
                    "Type": "AWS::Lambda::Function"
                },
{
                    "Values": [
                        "arn:aws:dynamodb"
                    ],
                    "Type": "AWS::DynamoDB::Table"
                }
            ],
            "ReadWriteType": "All"
        }
    ],
    "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/TrailName2"
}
```

Example trail that does not log AWS Key Management Service events

The following example creates an event selector for a trail named *TrailName* to include read-only and write-only management events, but to exclude AWS Key Management Service (AWS KMS) events. Because AWS KMS events are treated as management events, and there can be a high volume of them, they can have a substantial impact on your CloudTrail bill if you have more than one trail that captures management events. The user in this example has chosen to exclude AWS KMS events from every trail except for one. To exclude an event source, add `ExcludeManagementEventSources` to your event selectors, and specify an event source in the string value.

If you choose not to log management events, AWS KMS events are not logged, and you cannot change AWS KMS event logging settings.

To start logging AWS KMS events to a trail again, pass an empty array as the value of `ExcludeManagementEventSources`.

```
aws cloudtrail put-event-selectors --trail-name TrailName --event-
selectors '[{"ReadWriteType": "All","ExcludeManagementEventSources":
 ["kms.amazonaws.com"],"IncludeManagementEvents": true]}]'
```

The example returns the event selector that is configured for the trail.

```
{
    "EventSelectors": [
        {
            "ExcludeManagementEventSources": [ "kms.amazonaws.com" ],
            "IncludeManagementEvents": true,
            "DataResources": [],
            "ReadWriteType": "All"
        }
    ],
    "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/TrailName"
}
```

To start logging AWS KMS events to a trail again, pass an empty array as the value of `ExcludeManagementEventSources`, as shown in the following command.

```
aws cloudtrail put-event-selectors --trail-name TrailName --event-
selectors '[{"ReadWriteType": "All","ExcludeManagementEventSources":
 [],"IncludeManagementEvents": true]}]'
```

Example trail that logs relevant low-volume AWS Key Management Service events

The following example creates an event selector for a trail named *TrailName* to include write-only management events and AWS KMS events. Because AWS KMS events are treated as management events, and there can be a high volume of them, they can have a substantial impact on your CloudTrail bill if you have more than one trail that captures management events. The user in this example has chosen to include AWS KMS **Write** events, which will include `Disable`, `Delete` and `ScheduleKey`, but no longer include high-volume actions such as `Encrypt`, `Decrypt`, and `GenerateDataKey` (these are now treated as **Read** events).

```
aws cloudtrail put-event-selectors --trail-name TrailName --event-
selectors '[{"ReadWriteType": "WriteOnly","ExcludeManagementEventSources":
 [],"IncludeManagementEvents": true]}]'
```

The example returns the event selector that is configured for the trail. This logs write-only management events, including AWS KMS events.

```
{
    "EventSelectors": [
        {
            "ExcludeManagementEventSources": [],
            "IncludeManagementEvents": true,
            "DataResources": [],
            "ReadWriteType": "WriteOnly"
        }
    ],
    "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/TrailName"
}
```

Example trail that does not log Amazon RDS data API events

The following example creates an event selector for a trail named *TrailName* to include read-only and write-only management events, but to exclude Amazon RDS Data API events. Because Amazon RDS Data API events are treated as management events, and there can be a high volume of them, they can have a substantial impact on your CloudTrail bill if you have more than one trail that captures management events. The user in this example has chosen to exclude Amazon RDS Data API events from every trail except for one. To exclude an event source, add `ExcludeManagementEventSources` to your event selectors, and specify the Amazon RDS Data API event source in the string value: `rdsdata.amazonaws.com`.

If you choose not to log management events, Amazon RDS Data API events are not logged, and you cannot change event logging settings.

To start logging Amazon RDS Data API management events to a trail again, pass an empty array as the value of `ExcludeManagementEventSources`.

```
aws cloudtrail put-event-selectors --trail-name TrailName --event-
selectors '[{"ReadWriteType": "All","ExcludeManagementEventSources":
 ["rdsdata.amazonaws.com"],"IncludeManagementEvents": true]}]'
```

The example returns the event selector that is configured for the trail.

```
{
    "EventSelectors": [
        {
            "ExcludeManagementEventSources": [ "rdsdata.amazonaws.com" ],
            "IncludeManagementEvents": true,
            "DataResources": [],
            "ReadWriteType": "All"
        }
    ],
    "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/TrailName"
}
```

To start logging Amazon RDS Data API management events to a trail again, pass an empty array as the value of `ExcludeManagementEventSources`, as shown in the following command.

```
aws cloudtrail put-event-selectors --trail-name TrailName --event-
selectors '[{"ReadWriteType": "All","ExcludeManagementEventSources":
 [],"IncludeManagementEvents": true]}]'
```

Stopping and starting logging for a trail

The following commands start and stop CloudTrail logging.

```
aws cloudtrail start-logging --name awscloudtrail-example
```

```
aws cloudtrail stop-logging --name awscloudtrail-example
```

> **Note**
>
> Before deleting a bucket, run the `stop-logging` command to stop delivering events to the bucket. If you don't stop logging, CloudTrail attempts to deliver log files to a bucket with the same name for a limited period of time.
>
> If you stop logging or delete a trail, CloudTrail Insights is disabled on that trail.

Deleting a trail

If you've enabled CloudTrail management events in Amazon Security Lake, you are required to maintain at least one organizational trail that is multi-Region and logs both `read` and `write` management events. You cannot delete a trail if it is the only trail you have that meets this requirement, unless you turn off CloudTrail management events in Security Lake.

You can delete a trail with the following command. You can delete a trail only in the Region it was created (the Home Region).

```
aws cloudtrail delete-trail --name awscloudtrail-example
```

When you delete a trail, you do not delete the Amazon S3 bucket or the Amazon SNS topic associated with it. Use the AWS Management Console, AWS CLI, or service API to delete these resources separately.

Creating multiple trails

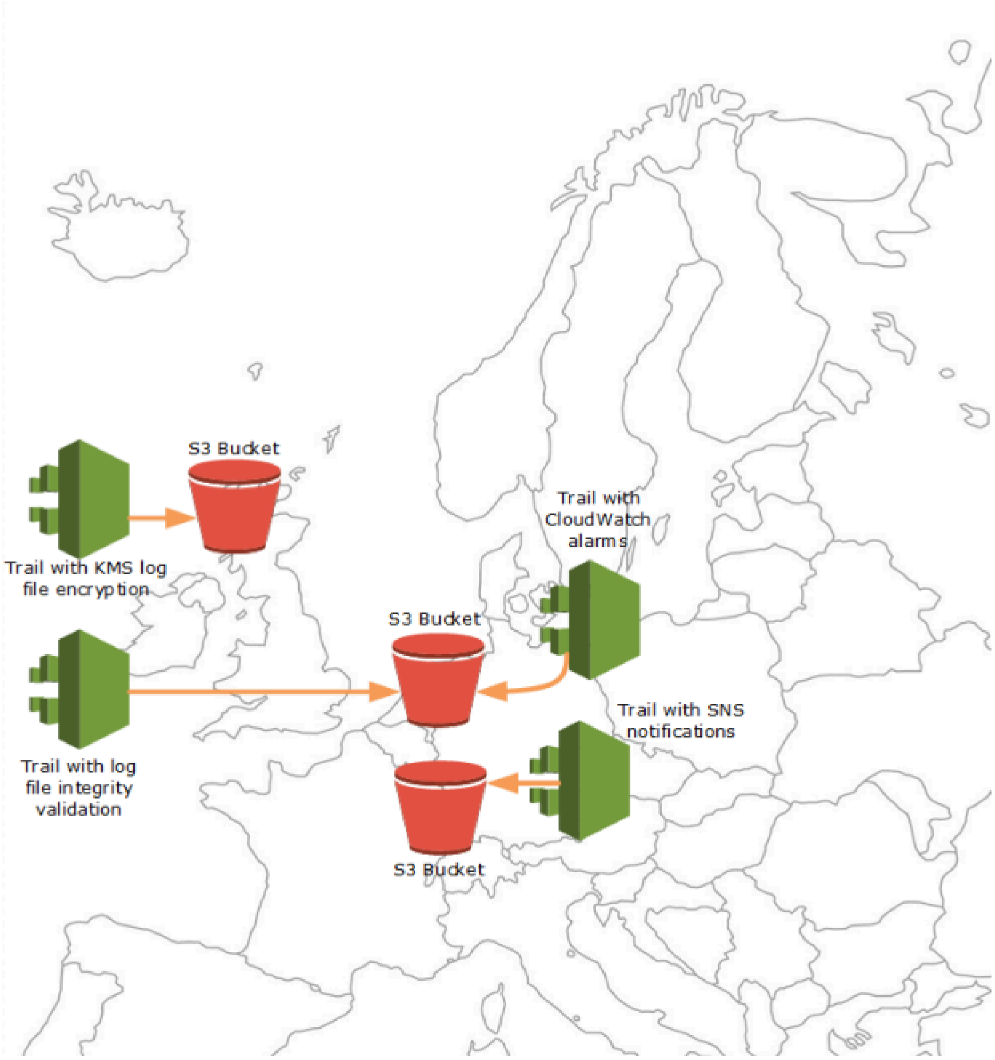
You can use CloudTrail log files to troubleshoot operational or security issues in your AWS account. You can create trails for different users, who can create and manage their own trails. You can configure trails to deliver log files to separate S3 buckets or shared S3 buckets.

> **Note**
>
> The first copy of management events in each AWS Region for an account is free. If you create more trails that deliver the same management events to other destinations, those subsequent deliveries incur CloudTrail costs. For more information about CloudTrail costs, see AWS CloudTrail Pricing and Managing CloudTrail trail costs.

For example, you might have the following users:

- A security administrator creates a trail in the Europe (Ireland) Region and configures KMS log file encryption. The trail delivers the log files to an S3 bucket in the Europe (Ireland) Region.
- An IT auditor creates a trail in the Europe (Ireland) Region and configures log file integrity validation to ensure the log files have not changed since CloudTrail delivered them. The trail is configured to deliver log files to an S3 bucket in the Europe (Frankfurt) Region
- A developer creates a trail in the Europe (Frankfurt) Region and configures CloudWatch alarms to receive notifications for specific API activity. The trail shares the same S3 bucket as the trail configured for log file integrity.
- Another developer creates a trail in the Europe (Frankfurt) Region and configures SNS. The log files are delivered to a separate S3 bucket in the Europe (Frankfurt) Region.

The following image illustrates this example.


S3 Bucket
Trail with KMS log file encryption
Trail with CloudWatch alarms
S3 Bucket
Trail with log file integrity validation
Trail with SNS notifications
S3 Bucket

Note

You can create up to five trails per AWS Region. A multi-Region trail counts as one trail per Region.

You can use resource-level permissions to manage a user's ability to perform specific operations on CloudTrail.

For example, you might grant one user permission to view trail activity, but restrict the user from starting or stopping logging for a trail. You might grant another user full permission to create and delete trails. This gives you granular control over your trails and user access.

For more information about resource-level permissions, see Examples: Creating and applying policies for actions on specific trails.

For more information about multiple trails, see the CloudTrail FAQs.

Creating a trail for an organization

If you have created an organization in AWS Organizations, you can create a trail that logs all events for all AWS accounts in that organization. This is sometimes called an *organization trail*.

The management account for the organization can assign a delegated administrator to create new organization trails or manage existing organization trails. For more information on adding a delegated administrator, see Add a CloudTrail delegated administrator.

The management account for the organization can edit an existing trail in their account, and apply it to an organization, making it an organization trail. Organization trails log events for the management account and all member accounts in the organization. For more information about AWS Organizations, see Organizations Terminology and Concepts.

Note

You must sign in with the management account or a delegated administrator account associated with an organization to create an organization trail. You must also have sufficient permissions for the user or role in the management or delegated administrator account to create the trail. If you don't have sufficient permissions, you won't have the option to apply the trail to an organization.

All organization trails created using the console are multi-Region organization trails that log events from the enabled AWS Regions in each member account in the organization. To log events in all AWS partitions in your organization, create a multi-Region organization trail in each partition. You can create either a single-Region or multi-Region organization trail by using the AWS CLI. If you create a single-Region trail, you log activity only in the trail's AWS Region (also referred to as the *Home* Region).

Although most AWS Regions are enabled by default for your AWS account, you must manually enable certain Regions (also referred to as *opt-in Regions*). For information about which Regions are enabled by default, see Considerations before enabling and disabling Regions in the *AWS Account Management Reference Guide*. For the list of Regions CloudTrail supports, see CloudTrail supported Regions.

When you create an organization trail, a copy of the trail with the name that you give it is created in the member accounts that belongs to your organization.

- If the organization trail is for a **single-Region** and the trail's home Region **is not an opt-Region**, a copy of the trail is created in the organization trail's home Region in each member account.
- If the organization trail is for a **single-Region** and the trail's home Region **is an opt-Region**, a copy of the trail is created in the organization trail's home Region in the member accounts that have enabled that Region.
- If the organization trail is **multi-Region** and the trail's home Region **is not an opt-in Region**, a copy of the trail is created in each enabled AWS Region in each member account. When a member account enables an opt-in Region, a copy of the multi-Region trail is created in the newly opted in Region for the member account after activation of that Region is complete.
- If the organization trail is **multi-Region** and the home Region **is an opt-in Region**, member accounts will not send activity to the organization trail unless they opt into the AWS Region where the multi-Region trail was created. For example, if you create a multi-Region trail and choose the Europe (Spain) Region as the home Region for the trail, only member accounts that enabled the Europe (Spain) Region for their account will send their account activity to the organization trail.

> **Note**
>
> CloudTrail creates organization trails in member accounts even if a resource validation fails. Examples of validation failures include:

- an incorrect Amazon S3 bucket policy
- an incorrect Amazon SNS topic policy
- inability to deliver to a CloudWatch Logs log group
- insufficient permission to encrypt using a KMS key

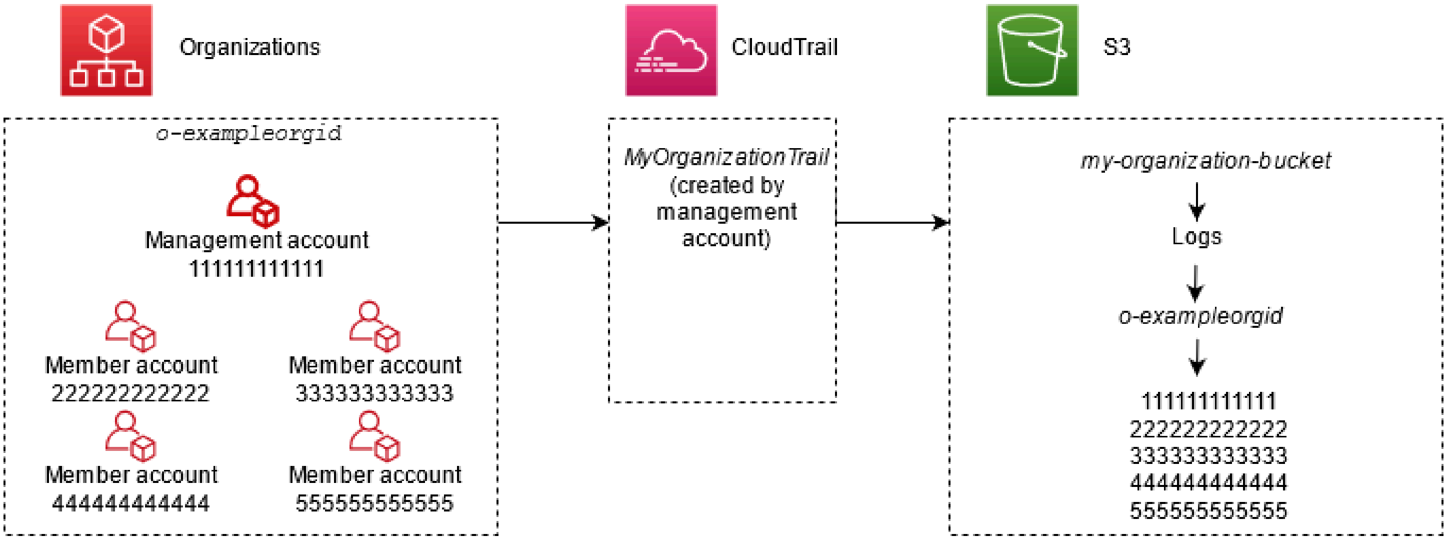
A member account with CloudTrail permissions can see any validation failures for an organization trail by viewing the trail's details page on the CloudTrail console, or by running the AWS CLI get-trail-status command.

Users with CloudTrail permissions in member accounts can see organization trails when they log into the AWS CloudTrail console from their AWS accounts, or when they run AWS CLI commands such as `describe-trails`. However, users in member accounts do not have sufficient permissions to delete organization trails, turn logging on or off, change what types of events are logged, or otherwise change an organization trail in any way.

When you create an organization trail in the console, or when you enable CloudTrail as a trusted service in Organizations, this creates a service-linked role to perform logging tasks in your organization's member accounts. This role is named **AWSServiceRoleForCloudTrail**, and is required for CloudTrail to log events for an organization. If an AWS account is added to an organization, the organization trail and service-linked role are added to that AWS account, and logging starts for that account automatically in the organization trail. If an AWS account is removed from an organization, the organization trail and service-linked role are deleted from the AWS account that is no longer part of the organization. However, log files for the removed account that were created before the account's removal remain in the Amazon S3 bucket where log files are stored for the trail.

If the management account for an AWS Organizations organization creates an organization trail, but then is subsequently removed as the organization's management account, any organization trail created using their account becomes a non-organization trail.

In the following example, the organization's management account 111111111111 creates a trail named *`MyOrganizationTrail`* for the organization *`o-exampleorgid`*. The trail logs activity for all accounts in the organization in the same Amazon S3 bucket. All accounts in the organization can see *`MyOrganizationTrail`* in their list of trails, but member accounts cannot remove or modify the organization trail. Only the management account or delegated administrator account can change or delete the trail for the organization. Only the management account can remove a member account from an organization. Similarly, by default, only the management account has access to the Amazon S3 bucket for the trail, and the logs contained within it. The high-level bucket structure for log files contains a folder named with the organization ID, and subfolders named with the account IDs for each account in the organization. Events for each member account are logged in the folder that corresponds to the member account ID. If member account 444444444444 is removed from the organization, *MyOrganizationTrail* and the service-linked role no longer appear in AWS account 444444444444, and no further events are logged for that account by the organization trail. However, the 444444444444 folder remains in the Amazon S3 bucket, with all logs created before the removal of the account from the organization.



In this example, the ARN of the trail created in the management account is `aws:cloudtrail:us-east-2:111111111111:trail/`*`MyOrganizationTrail`*. This ARN is the ARN for the trail in all member accounts as well.

Organization trails are similar to regular trails in many ways. You can create multiple trails for your organization, and choose whether to create an organization trail in all Regions or a single Region, and what kinds of events you want logged in your organization trail, just as in any other trail. However, there are some differences. For example, when you create a trail in the console and choose whether to log data events for Amazon S3 buckets or AWS Lambda functions, the only resources listed in the CloudTrail console are those for the management account, but you can add the ARNs for resources in member accounts. Data events for specified member account resources are logged without having to manually configure cross-account access to those resources. For more information about logging management events, Insights events, and data events, see Logging management events, Logging data events, and Logging Insights events.

Note

In the console, you create a multi-Region trail. This is a recommended best practice; logging activity in all Regions in your AWS account helps you keep your AWS environment more secure. To create a single-Region trail, use the AWS CLI.

When you view events in **Event history** for an organization in AWS Organizations, you can view the events only for the AWS account with which you are signed in. For example, if you are signed in with the organization management account, **Event history** shows the last 90 days of management events for the management account. Organization member account events are not shown in **Event history** for the management account. To view member account events in **Event history**, sign in with the member account.

You can configure other AWS services to further analyze and act upon the event data collected in CloudTrail logs for an organization trail the same way you would for any other trail. For example, you can analyze the data in an organization trail using Amazon Athena. For more information, see AWS service integrations with CloudTrail logs.

Topics

Moving from member account trails to organization trails

If you already have CloudTrail trails configured for individual member accounts, but want to move to an organization trail to log events in all accounts, you do not want to lose events by deleting individual member account trails before you create an organization trail. But when you have two trails, you incur higher costs because of the additional copy of events delivered to the organization trail.

To help manage costs, but avoid losing events before log delivery starts on the organization trail, consider keeping both your individual member account trails and your organization trail for up to one day. This ensures that the organization trail logs all events, but you incur duplicate event costs only for one day. After the first day, you can stop logging on (or delete) any individual member account trails.

Prepare for creating a trail for your organization

Before you create a trail for your organization, be sure that your organization management account or delegated administrator account is set up correctly for trail creation.

- Your organization must have all features enabled before you can create a trail for it. For more information, see Enabling All Features in Your Organization.
- The management account must have the **AWSServiceRoleForOrganizations** role. This role is created automatically by Organizations when you create your organization, and is required for CloudTrail to log events for an organization. For more information, see Organizations and service-linked roles.
- The user or role that creates the organization trail in the management or delegated administrator account must have sufficient permissions to create an organization trail. You must at least apply either the **AWSCloudTrail_FullAccess** policy, or an equivalent policy, to that role or user. You must also have sufficient permissions in IAM and Organizations to create the service-linked role and enable trusted access. If you choose to create a new S3 bucket for an organization trail using the CloudTrail console, your policy also needs to include the `s3:PutEncryptionConfiguration` action because by default server-side encryption is enabled for the bucket. The following example policy shows the minimum required permissions.

> **Note**
>
> You shouldn't share the **AWSCloudTrail_FullAccess** policy broadly across your AWS account. Instead, you should restrict it to AWS account administrators due to the highly sensitive nature of the information collected by CloudTrail. Users with this role have the ability to turn off or reconfigure the most sensitive and important auditing functions in their AWS accounts. For this reason, you must closely control and monitor access to this policy.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iam:GetRole",
                "organizations:EnableAWSServiceAccess",
                "organizations:ListAccounts",
                "iam:CreateServiceLinkedRole",
                "organizations:DisableAWSServiceAccess",
                "organizations:DescribeOrganization",
                "organizations:ListAWSServiceAccessForOrganization",
                "s3:PutEncryptionConfiguration"
            ],
            "Resource": "*"
        }
    ]
}
```

- To use the AWS CLI or the CloudTrail APIs to create an organization trail, you must enable trusted access for CloudTrail in Organizations, and you must manually create an Amazon S3 bucket with a policy that allows logging for an organization trail. For more information, see Creating a trail for an organization with the AWS CLI.
- To use an existing IAM role to add monitoring of an organization trail to Amazon CloudWatch Logs, you must manually modify the IAM role to allow delivery of CloudWatch Logs for member accounts to the CloudWatch Logs group for the management account, as shown in the following example.

> **Note**
>
> You must use an IAM role and CloudWatch Logs log group that exists in your own account. You cannot use an IAM role or CloudWatch Logs log group owned by a different account.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AWSCloudTrailCreateLogStream20141101",
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogStream"
            ],
            "Resource": [
                "arn:aws:logs:us-east-2:111111111111:log-group:CloudTrail/
DefaultLogGroupTest:log-stream:111111111111_CloudTrail_us-east-2*",
                "arn:aws:logs:us-east-2:111111111111:log-group:CloudTrail/
DefaultLogGroupTest:log-stream:o-exampleorgid_*"
            ]
        },
        {
            "Sid": "AWSCloudTrailPutLogEvents20141101",
            "Effect": "Allow",
            "Action": [
                "logs:PutLogEvents"
            ],
            "Resource": [
                "arn:aws:logs:us-east-2:111111111111:log-group:CloudTrail/
DefaultLogGroupTest:log-stream:111111111111_CloudTrail_us-east-2*",
                "arn:aws:logs:us-east-2:111111111111:log-group:CloudTrail/
DefaultLogGroupTest:log-stream:o-exampleorgid_*"
            ]
        }
    ]
}
```

You can learn more about CloudTrail and Amazon CloudWatch Logs in Monitoring CloudTrail Log Files with Amazon CloudWatch Logs. In addition, consider the limits on CloudWatch Logs and the pricing considerations for the service before deciding to enable the experience for an organization trail. For more information, see CloudWatch Logs Limits and Amazon CloudWatch Pricing.

- To log data events in your organization trail for specific resources in member accounts, have ready a list of Amazon Resource Names (ARNs) for each of those resources. Member account resources are not displayed in the CloudTrail console when you create a trail; you can browse for resources in the management account on which data event collection is supported, such as S3 buckets. Similarly, if you want to add specific member resources when creating or updating an organization trail at the command line, you need the ARNs for those resources.

> **Note**
>
> Additional charges apply for logging data events. For CloudTrail pricing, see AWS CloudTrail Pricing.

You should also consider reviewing how many trails already exist in the management account and in the member accounts before creating an organization trail. CloudTrail limits the number of trails that can be created in each Region. You cannot exceed this limit in the Region where you create the organization trail in the management account. However, the trail will be created in the member accounts even if member accounts have reached the limit of trails in a Region. While the first trail of management events in any Region is free, charges apply to additional trails. To reduce the potential cost of an organization trail, consider deleting any unneeded trails in the management and member accounts. For more information about CloudTrail pricing, see AWS CloudTrail Pricing.

Security best practices in organization trails

As a security best practice, we recommend adding the `aws:SourceArn` condition key to resource policies (such as those for S3 buckets, KMS keys, or SNS topics) that you use with an organization trail. The value of `aws:SourceArn` is the organization trail ARN (or ARNs, if you are using the same resource for more than one trail, such as the same S3 bucket to store logs for more than one trail). This ensures that the resource, such as an S3 bucket, accepts only data that is associated with the specific trail. The trail ARN must use the account ID of the management account. The following policy snippet shows an example where more than one trail is using the resource.

```
"Condition": {
    "StringEquals": {
      "aws:SourceArn": ["Trail_ARN_1",..., "Trail_ARN_n"]
    }
}
```

For information about how to add condition keys to resource policies, see the following:

- Amazon S3 bucket policy for CloudTrail
- Configure AWS KMS key policies for CloudTrail
- Amazon SNS topic policy for CloudTrail

Creating a trail for your organization in the console

To create an organization trail from the CloudTrail console, you must sign in to the console as a user or role in the management or delegated administrator account that has sufficient permissions. If you don't sign in with the management or delegated administrator account, you won't see the option to apply a trail to an organization when you create or edit a trail from the CloudTrail console.

To create an organization trail with the AWS Management Console

1. Sign in to the AWS Management Console and open the CloudTrail console at https://console.aws.amazon.com/cloudtrail/.

   You must be signed in using an IAM identity in the management or delegated administrator account with sufficient permissions to create an organization trail.
2. Choose **Trails**, and then choose **Create trail**.
3. On the **Create Trail** page, for **Trail name**, type a name for your trail. For more information, see Naming requirements for CloudTrail resources, S3 buckets, and KMS keys.
4. Select **Enable for all accounts in my organization**. You only see this option if you sign in to the console with a user or role in the management or delegated administrator account. To successfully create an organization trail, be sure that the user or role has sufficient permissions.
5. For **Storage location**, choose **Create new S3 bucket** to create a bucket. When you create a bucket, CloudTrail creates and applies the required bucket policies.

   > **Note**
   >
   > If you chose **Use existing S3 bucket**, specify a bucket in **Trail log bucket name**, or choose **Browse** to choose a bucket. You can choose a bucket belonging to any account, however, the bucket policy must grant CloudTrail permission to write to it. For information about manually editing the bucket policy, see Amazon S3 bucket policy for CloudTrail.

   To make it easier to find your logs, create a new folder (also known as a *prefix*) in an existing bucket to store your CloudTrail logs. Enter the prefix in **Prefix**.

6. For **Log file SSE-KMS encryption**, choose **Enabled** if you want to encrypt your log files using SSE-KMS encryption instead of SSE-S3 encryption. The default is **Enabled**. If you don't enable SSE-KMS encryption, your logs are encrypted using SSE-S3 encryption. For more information about SSE-KMS encryption, see Using server-side encryption with AWS Key Management Service (SSE-KMS). For more information about SSE-S3 encryption, see Using Server-Side Encryption with Amazon S3-Managed Encryption Keys (SSE-S3).

   If you enable SSE-KMS encryption, choose a **New** or **Existing** AWS KMS key. In **AWS KMS Alias**, specify an alias, in the format `alias/MyAliasName`. For more information, see Updating a resource to use your KMS key with the console.

   > **Note**
   >
   > You can also type the ARN of a key from another account. For more information, see Updating a resource to use your KMS key with the console. The key policy must allow CloudTrail to use the key to encrypt your log files, and allow the users you specify to read log files in unencrypted form. For information about manually editing the key policy, see Configure AWS KMS key policies for CloudTrail.

7. In **Additional settings**, configure the following.

   a. For **Log file validation**, choose **Enabled** to have log digests delivered to your S3 bucket. You can use the digest files to verify that your log files did not change after CloudTrail delivered them. For more information, see Validating CloudTrail log file integrity.

   b. For **SNS notification delivery**, choose **Enabled** to be notified each time a log is delivered to your bucket. CloudTrail stores multiple events in a log file. SNS notifications are sent for every log file, not for every event. For more information, see Configuring Amazon SNS notifications for CloudTrail.

      If you enable SNS notifications, for **Create a new SNS topic**, choose **New** to create a topic, or choose **Existing** to use an existing topic. If you are creating a trail that applies to all Regions, SNS notifications for log file deliveries from all Regions are sent to the single SNS topic that you create.

      If you choose **New**, CloudTrail specifies a name for the new topic for you, or you can type a name. If you choose **Existing**, choose an SNS topic from the drop-down list. You can also enter the ARN of a topic from another Region or from an account with appropriate permissions. For more information, see Amazon SNS topic policy for CloudTrail.

If you create a topic, you must subscribe to the topic to be notified of log file delivery. You can subscribe from the Amazon SNS console. Due to the frequency of notifications, we recommend that you configure the subscription to use an Amazon SQS queue to handle notifications programmatically. For more information, see Getting started with Amazon SNS in the *Amazon Simple Notification Service Developer Guide*.

8. Optionally, configure CloudTrail to send log files to CloudWatch Logs by choosing **Enabled** in **CloudWatch Logs**. For more information, see Sending events to CloudWatch Logs.

> **Note**
>
> Only the management account can configure a CloudWatch Logs log group for an organization trail using the console. The delegated administrator can configure a CloudWatch Logs log group using the AWS CLI or CloudTrail `CreateTrail` or `UpdateTrail` API operations.

   a. If you enable integration with CloudWatch Logs, choose **New** to create a new log group, or **Existing** to use an existing one. If you choose **New**, CloudTrail specifies a name for the new log group for you, or you can type a name.

   b. If you choose **Existing**, choose a log group from the drop-down list.

   c. Choose **New** to create a new IAM role for permissions to send logs to CloudWatch Logs. Choose **Existing** to choose an existing IAM role from the drop-down list. The policy statement for the new or existing role is displayed when you expand **Policy document**. For more information about this role, see Role policy document for CloudTrail to use CloudWatch Logs for monitoring.

   > **Note**
   >
   > When you configure a trail, you can choose an S3 bucket and Amazon SNS topic that belong to another account. However, if you want CloudTrail to deliver events to a CloudWatch Logs log group, you must choose a log group that exists in your current account.

9. For **Tags**, add one or more custom tags (key-value pairs) to your trail. Tags can help you identify both your CloudTrail trails and the Amazon S3 buckets that contain CloudTrail log

files. You can then use resource groups for your CloudTrail resources. For more information, see AWS Resource Groups and Tags.

10. On the **Choose log events** page, choose the event types that you want to log. For **Management events**, do the following.

    a. For **API activity**, choose if you want your trail to log **Read** events, **Write** events, or both. For more information, see Management events.

    b. Choose **Exclude AWS KMS events** to filter AWS Key Management Service (AWS KMS) events out of your trail. The default setting is to include all AWS KMS events.

       The option to log or exclude AWS KMS events is available only if you log management events on your trail. If you choose not to log management events, AWS KMS events are not logged, and you cannot change AWS KMS event logging settings.

       AWS KMS actions such as `Encrypt`, `Decrypt`, and `GenerateDataKey` typically generate a large volume (more than 99%) of events. These actions are now logged as **Read** events. Low-volume, relevant AWS KMS actions such as `Disable`, `Delete`, and `ScheduleKey` (which typically account for less than 0.5% of AWS KMS event volume) are logged as **Write** events.

       To exclude high-volume events like `Encrypt`, `Decrypt`, and `GenerateDataKey`, but still log relevant events such as `Disable`, `Delete` and `ScheduleKey`, choose to log **Write** management events, and clear the check box for **Exclude AWS KMS events**.

    c. Choose **Exclude Amazon RDS Data API events** to filter Amazon Relational Database Service Data API events out of your trail. The default setting is to include all Amazon RDS Data API events. For more information about Amazon RDS Data API events, see Logging Data API calls with AWS CloudTrail in the *Amazon RDS User Guide for Aurora*.

11. To log data events, choose **Data events**. Additional charges apply for logging data events. For more information, see AWS CloudTrail Pricing.

12. > ⚠ **Important**
    >
    > Steps 12-16 are for configuring data events using advanced event selectors, which is the default. Advanced event selectors let you configure more data event types and offer fine-grained control over which data events your trail captures. If you opted to use basic event selectors, complete the steps in Configure data event settings using basic event selectors, then return to step 17 of this procedure.

For **Data event type**, choose the resource type on which you want to log data events. For more information about available data event types, see Data events.

> **Note**
>
> To log data events for AWS Glue tables created by Lake Formation, choose **Lake Formation**.

13. Choose a log selector template. CloudTrail includes predefined templates that log all data events for the resource type. To build a custom log selector template, choose **Custom**.

> **Note**
>
> Choosing a predefined template for S3 buckets enables data event logging for all buckets currently in your AWS account and any buckets you create after you finish creating the trail. It also enables logging of data event activity performed by any IAM identity in your AWS account, even if that activity is performed on a bucket that belongs to another AWS account.
> If the trail applies only to one Region, choosing a predefined template that logs all S3 buckets enables data event logging for all buckets in the same Region as your trail and any buckets you create later in that Region. It will not log data events for Amazon S3 buckets in other Regions in your AWS account.
> If you are creating a trail for all Regions, choosing a predefined template for Lambda functions enables data event logging for all functions currently in your AWS account, and any Lambda functions you might create in any Region after you finish creating the trail. If you are creating a trail for a single Region (done by using the AWS CLI), this selection enables data event logging for all functions currently in that Region in your AWS account, and any Lambda functions you might create in that Region after you finish creating the trail. It does not enable data event logging for Lambda functions created in other Regions.
> Logging data events for all functions also enables logging of data event activity performed by any IAM identity in your AWS account, even if that activity is performed on a function that belongs to another AWS account.

14. (Optional) In **Selector name**, enter a name to identify your selector. The selector name is a descriptive name for an advanced event selector, such as "Log data events for only two S3

buckets". The selector name is listed as Name in the advanced event selector and is viewable if you expand the **JSON view**.

15. In **Advanced event selectors**, build an expression for the specific resources on which you want to log data events. You can skip this step if you are using a predefined log template.

    a. Choose from the following fields.

    - **`readOnly`** - `readOnly` can be set to **equals** a value of `true` or `false`. Read-only data events are events that do not change the state of a resource, such as `Get*` or `Describe*` events. Write events add, change, or delete resources, attributes, or artifacts, such as `Put*`, `Delete*`, or `Write*` events. To log both `read` and `write` events, don't add a `readOnly` selector.
    - **`eventName`** - `eventName` can use any operator. You can use it to include or exclude any data event logged to CloudTrail, such as `PutBucket`, `PutItem`, or `GetSnapshotBlock`.
    - **`resources.ARN`** - You can use any operator with `resources.ARN`, but if you use **equals** or **does not equal**, the value must exactly match the ARN of a valid resource of the type you've specified in the template as the value of `resources.type`.

      The following table shows the valid ARN format for each `resources.type`.

      Note

      You can't use the `resources.ARN` field to filter resource types that do not have ARNs.

resources.type	resources.ARN
`AWS::DynamoDB::Table`[1]	`arn:`*`partition`*` :dynamodb : `*`region`*`:`*`account_ID`*` :table/`*`table_name`*
`AWS::Lambda::Function`	`arn:`*`partition`*` :lambda:`*`region`*`:`*`account_ID`*` :function: `*`function_name`*

resources.type	resources.ARN
`AWS::S3::Object` [2]	`arn:partition:s3:::DOC-EXAMPLE-BUCKET/` `arn:partition:s3:::DOC-EXAMPLE-BUCKET/object_or_file_name/`
`AWS::AppConfig::Configuration`	`arn:partition:appconfig:region:account_ID:application/application_ID/environment/environment_ID/configuration/configuration_profile_ID`
`AWS::B2BI::Transformer`	`arn:partition:b2bi:region:account_ID:transformer/transformer_ID`
`AWS::Bedrock::AgentAlias`	`arn:partition:bedrock:region:account_ID:agent-alias/agent_ID/alias_ID`
`AWS::Bedrock::FlowAlias`	`arn:partition:bedrock:region:account_ID:flow/flow_ID/alias/alias_ID`
`AWS::Bedrock::Guardrail`	`arn:partition:bedrock:region:account_ID:guardrail/guardrail_ID`
`AWS::Bedrock::KnowledgeBase`	`arn:partition:bedrock:region:account_ID:knowledge-base/knowledge_base_ID`
`AWS::Cassandra::Table`	`arn:partition:cassandra:region:account_ID:keyspace/keyspace_name/table/table_name`

resources.type	resources.ARN
`AWS::CloudFront::KeyValueStore`	`arn:partition :cloudfront: region:account_ID :key-value-store/KVS_name`
`AWS::CloudTrail::Channel`	`arn:partition :cloudtrail: region:account_ID :channel/ channel_UUID`
`AWS::CodeWhisperer::Customization`	`arn:partition :codewhisperer: region:account_ID :customization/ customization_ID`
`AWS::CodeWhisperer::Profile`	`arn:partition :codewhisperer: region:account_ID :profile/ profile_ID`
`AWS::Cognito::IdentityPool`	`arn:partition :cognito-identity: region:account_ID :identitypool/ identity_pool_ID`
`AWS::DynamoDB::Stream`	`arn:partition :dynamodb: region:account_ID :table/table_name /stream/date_time`
`AWS::EC2::Snapshot`	`arn:partition :ec2:region::snapshot/ snapshot_ID`
`AWS::EMRWAL::Workspace`	`arn:partition :emrwal:region:account_ID :workspace/ workspace_name`

resources.type	resources.ARN
`AWS::FinSpace::Environment`	arn:*partition*:finspace:*region*:*account_ID*:environment/*environment_ID*
`AWS::Glue::Table`	arn:*partition*:glue:*region*:*account_ID*:table/*database_name*/*table_name*
`AWS::GreengrassV2::ComponentVersion`	arn:*partition*:greengrass:*region*:*account_ID*:components/*component_name*
`AWS::GreengrassV2::Deployment`	arn:*partition*:greengrass:*region*:*account_ID*:deployments/*deployment_ID*
`AWS::GuardDuty::Detector`	arn:*partition*:guardduty:*region*:*account_ID*:detector/*detector_ID*
`AWS::IoT::Certificate`	arn:*partition*:iot:*region*:*account_ID*:cert/*certificate_ID*
`AWS::IoT::Thing`	arn:*partition*:iot:*region*:*account_ID*:thing/*thing_ID*
`AWS::IoTSiteWise::Asset`	arn:*partition*:iotsitewise:*region*:*account_ID*:asset/*asset_ID*
`AWS::IoTSiteWise::TimeSeries`	arn:*partition*:iotsitewise:*region*:*account_ID*:timeseries/*timeseries_ID*

resources.type	resources.ARN
`AWS::IoTTwinMaker::Entity`	arn:*partition*:iottwinmaker:*region*:*account_ID*:workspace/*workspace_ID*/entity/*entity_ID*
`AWS::IoTTwinMaker::Workspace`	arn:*partition*:iottwinmaker:*region*:*account_ID*:workspace/*workspace_ID*
`AWS::KendraRanking::ExecutionPlan`	arn:*partition*:kendra-ranking:*region*:*account_ID*:rescore-execution-plan/*rescore_execution_plan_ID*
`AWS::Kinesis::Stream`	arn:*partition*:kinesis:*region*:*account_ID*:stream/*stream_name*
`AWS::Kinesis::StreamConsumer`	arn:*partition*:kinesis:*region*:*account_ID*:*stream_type*/*stream_name*/consumer/*consumer_name*:*consumer_creation_timestamp*
`AWS::KinesisVideo::Stream`	arn:*partition*:kinesisvideo:*region*:*account_ID*:stream/*stream_name*/*creation_time*
`AWS::MachineLearning::MlModel`	arn:*partition*:machinelearning:*region*:*account_ID*:mlmodel/*model_ID*
`AWS::ManagedBlockchain::Network`	arn:*partition*:managedblockchain:::networks/*network_name*

resources.type	resources.ARN
`AWS::ManagedBlockchain::Node`	`arn:partition :managedblockchain: region:account_ID :nodes/node_ID`
`AWS::MedicalImaging::Datastore`	`arn:partition :medical-imaging: region:account_ID :datastore/ data_store_ID`
`AWS::NeptuneGraph::Graph`	`arn:partition :neptune-graph: region:account_ID :graph/graph_ID`
`AWS::PaymentCryptography::Alias`	`arn:partition :payment-cryptography: region:account_ID :alias/alias`
`AWS::PaymentCryptography::Key`	`arn:partition :payment-cryptography: region:account_ID :key/key_ID`
`AWS::PCAConnectorAD::Connector`	`arn:partition :pca-connector-ad: region:account_ID :connector/ connector_ID`
`AWS::PCAConnectorSCEP::Connector`	`arn:partition :pca-connector-scep: region:account_ID :connector/ connector_ID`
`AWS::QApps:QApp`	`arn:partition :qapps:region:account_ID :application/ application_UUID / qapp/qapp_UUID`
`AWS::QBusiness::Application`	`arn:partition :qbusiness: region:account_ID :application/ application_ID`

resources.type	resources.ARN
AWS::QBusiness::DataSource	arn:*partition*:qbusiness:*region*:*account_ID*:application/*application_ID*/index/*index_ID*/data-source/*datasource_ID*
AWS::QBusiness::Index	arn:*partition*:qbusiness:*region*:*account_ID*:application/*application_ID*/index/*index_ID*
AWS::QBusiness::WebExperience	arn:*partition*:qbusiness:*region*:*account_ID*:application/*application_ID*/web-experience/*web_experienc_ID*
AWS::RDS::DBCluster	arn:*partition*:rds:*region*:*account_ID*:cluster/*cluster_name*
AWS::S3::AccessPoint [3]	arn:*partition*:s3:*region*:*account_ID*:accesspoint/*access_point_name*
AWS::S3Express::Object	arn:*partition*:s3express:*region*:*account_ID*:bucket/*bucket_name*
AWS::S3ObjectLambda::AccessPoint	arn:*partition*:s3-object-lambda:*region*:*account_ID*:accesspoint/*access_point_name*
AWS::S3Outposts::Object	arn:*partition*:s3-outposts:*region*:*account_ID*:*object_path*

resources.type	resources.ARN
AWS::SageMaker::Endpoint	arn:*partition*:sagemaker:*region*:*account_ID*:endpoint/*endpoint_name*
AWS::SageMaker::ExperimentTrialComponent	arn:*partition*:sagemaker:*region*:*account_ID*:experiment-trial-component/*experiment_trial_component_name*
AWS::SageMaker::FeatureGroup	arn:*partition*:sagemaker:*region*:*account_ID*:feature-group/*feature_group_name*
AWS::SCN::Instance	arn:*partition*:scn:*region*:*account_ID*:instance/*instance_ID*
AWS::ServiceDiscovery::Namespace	arn:*partition*:servicediscovery:*region*:*account_ID*:namespace/*namespace_ID*
AWS::ServiceDiscovery::Service	arn:*partition*:servicediscovery:*region*:*account_ID*:service/*service_ID*
AWS::SNS::PlatformEndpoint	arn:*partition*:sns:*region*:*account_ID*:endpoint/*endpoint_type*/*endpoint_name*/*endpoint_ID*
AWS::SNS::Topic	arn:*partition*:sns:*region*:*account_ID*:*topic_name*

resources.type	resources.ARN
`AWS::SQS::Queue`	`arn:partition:sqs:region:account_ID:queue_name`
`AWS::SSM::ManagedNode`	The ARN must be in one of the following formats: - `arn:partition:ssm:region:account_ID:managed-instance/instance_ID` - `arn:partition:ec2:region:account_ID:instance/instance_ID`
`AWS::SSMMessages::ControlChannel`	`arn:partition:ssmmessages:region:account_ID:control-channel/control_channel_ID`
`AWS::StepFunctions::StateMachine`	The ARN must be in one of the following formats: - `arn:partition:states:region:account_ID:stateMachine:stateMachine_name` - `arn:partition:states:region:account_ID:stateMachine:stateMachine_name/label_name`
`AWS::SWF::Domain`	`arn:partition:swf:region:account_ID:/domain/domain_name`
`AWS::ThinClient::Device`	`arn:partition:thinclient:region:account_ID:device/device_ID`

resources.type	resources.ARN
`AWS::ThinClient::Environment`	`arn:partition:thinclient:region:account_ID:environment/environment_ID`
`AWS::Timestream::Database`	`arn:partition:timestream:region:account_ID:database/database_name`
`AWS::Timestream::Table`	`arn:partition:timestream:region:account_ID:database/database_name/table/table_name`
`AWS::VerifiedPermissions::PolicyStore`	`arn:partition:verifiedpermissions:region:account_ID:policy-store/policy_store_ID`

[1] For tables with streams enabled, the `resources` field in the data event contains both `AWS::DynamoDB::Stream` and `AWS::DynamoDB::Table`. If you specify `AWS::DynamoDB::Table` for the `resources.type`, it will log both DynamoDB table and DynamoDB streams events by default. To exclude streams events, add a filter on the `eventName` field.

[2] To log all data events for all objects in a specific S3 bucket, use the `StartsWith` operator, and include only the bucket ARN as the matching value. The trailing slash is intentional; do not exclude it.

[3] To log events on all objects in an S3 access point, we recommend that you use only the access point ARN, don't include the object path, and use the `StartsWith` or `NotStartsWith` operators.

For more information about the ARN formats of data event resources, see Actions, resources, and condition keys in the *AWS Identity and Access Management User Guide*.

b. For each field, choose **+ Condition** to add as many conditions as you need, up to a maximum of 500 specified values for all conditions. For example, to exclude data events for two S3 buckets from data events that are logged on your trail, you can set the field to **resources.ARN**, set the operator for **does not start with**, and then either paste in an S3 bucket ARN, or browse for the S3 buckets for which you do not want to log events.

   To add the second S3 bucket, choose **+ Condition**, and then repeat the preceding instruction, pasting in the ARN for or browsing for a different bucket.

   > **Note**
   >
   > You can have a maximum of 500 values for all selectors on a trail. This includes arrays of multiple values for a selector such as `eventName`. If you have single values for all selectors, you can have a maximum of 500 conditions added to a selector.
   > If you have more than 15,000 Lambda functions in your account, you cannot view or select all functions in the CloudTrail console when creating a trail. You can still log all functions with a predefined selector template, even if they are not displayed. If you want to log data events for specific functions, you can manually add a function if you know its ARN. You can also finish creating the trail in the console, and then use the AWS CLI and the **put-event-selectors** command to configure data event logging for specific Lambda functions. For more information, see Managing trails with the AWS CLI.

c. Choose **+ Field** to add additional fields as required. To avoid errors, do not set conflicting or duplicate values for fields. For example, do not specify an ARN in one selector to be equal to a value, then specify that the ARN not equal the same value in another selector.

16. To add another data type on which to log data events, choose **Add data event type**. Repeat steps 12 through this step to configure advanced event selectors for the data event type.

17. Choose **Insights events** if you want your trail to log CloudTrail Insights events.

    In **Event type**, select **Insights events**. In **Insights events**, choose **API call rate**, **API error rate**, or both. You must be logging **Write** management events to log Insights events for **API call rate**. You must be logging **Read** or **Write** management events to log Insights events for **API error rate**.

CloudTrail Insights analyzes management events for unusual activity, and logs events when anomalies are detected. By default, trails don't log Insights events. For more information about Insights events, see Logging Insights events. Additional charges apply for logging Insights events. For CloudTrail pricing, see AWS CloudTrail Pricing.

Insights events are delivered to a different folder named `/CloudTrail-Insight`of the same S3 bucket that is specified in the **Storage location** area of the trail details page. CloudTrail creates the new prefix for you. For example, if your current destination S3 bucket is named `DOC-EXAMPLE-DESTINATION-BUCKET/AWSLogs/CloudTrail/`, the S3 bucket name with a new prefix is named `DOC-EXAMPLE-DESTINATION-BUCKET/AWSLogs/CloudTrail-Insight/`.

18. When you are finished choosing event types to log, choose **Next**.
19. On the **Review and create** page, review your choices. Choose **Edit** in a section to change the trail settings shown in that section. When you are ready to create the trail, choose **Create trail**.
20. The new trail appears on the **Trails** page. An organization trail might take up to 24 hours to be created in all Regions in all member accounts. The **Trails** page shows the trails in your account from all Regions. In about 5 minutes, CloudTrail publishes log files that show the AWS API calls made in your organization. You can see the log files in the Amazon S3 bucket that you specified.

Note

You can't rename a trail after it has been created. Instead, you can delete the trail and create a new one.

Next steps

After you create your trail, you can return to the trail to make changes:

- Change the configuration of your trail by editing it. For more information, see Updating a trail with the CloudTrail console.
- If needed, configure the Amazon S3 bucket to allow specific users in member accounts to read the log files for the organization. For more information, see Sharing CloudTrail log files between AWS accounts.

- Configure CloudTrail to send log files to CloudWatch Logs. For more information, see Sending events to CloudWatch Logs and the CloudWatch Logs item in Prepare for creating a trail for your organization.

  > **Note**
  >
  > Only the management account can configure a CloudWatch Logs log group for an organization trail.

- Create a table and use it to run a query in Amazon Athena to analyze your AWS service activity. For more information, see Creating a Table for CloudTrail Logs in the CloudTrail Console in the Amazon Athena User Guide.
- Add custom tags (key-value pairs) to the trail.
- To create another organization trail, return to the **Trails** page and choose **Create trail**.

> **Note**
>
> When you configure a trail, you can choose an Amazon S3 bucket and SNS topic that belong to another account. However, if you want CloudTrail to deliver events to a CloudWatch Logs log group, you must choose a log group that exists in your current account.

Creating a trail for an organization with the AWS CLI

You can create an organization trail by using the AWS CLI. The AWS CLI is regularly updated with additional functionality and commands. To help ensure success, be sure that you have installed or updated to a recent AWS CLI version before you begin.

> **Note**
>
> The examples in this section are specific to creating and updating organization trails. For examples of using the AWS CLI to manage trails, see Managing trails with the AWS CLI and Configuring CloudWatch Logs monitoring with the AWS CLI. When creating or updating an organization trail with the AWS CLI, you must use an AWS CLI profile in the management account or delegated administrator account with sufficient permissions.

If you are converting an organization trail to a non-organization trail, you must use the management account for the organization.
You must configure the Amazon S3 bucket used for an organization trail with sufficient permissions.

Create or update an Amazon S3 bucket to use to store the log files for an organization trail

You must specify an Amazon S3 bucket to receive the log files for an organization trail. This bucket must have a policy that allows CloudTrail to put the log files for the organization into the bucket.

The following is an example policy for an Amazon S3 bucket named *DOC-EXAMPLE-BUCKET*, which is owned by the organization's management account. Replace *DOC-EXAMPLE-BUCKET*, *region*, *managementAccountID*, *trailName*, and *o-organizationID* with the values for your organization

This bucket policy contains three statements.

- The first statement allows CloudTrail to call the Amazon S3 `GetBucketAcl` action on the Amazon S3 bucket.
- The second statement allows logging in the event the trail is changed from an organization trail to a trail for that account only.
- The third statement allows logging for an organization trail.

The example policy includes an `aws:SourceArn` condition key for the Amazon S3 bucket policy. The IAM global condition key `aws:SourceArn` helps ensure that CloudTrail writes to the S3 bucket only for a specific trail or trails. In an organization trail, the value of `aws:SourceArn` must be a trail ARN that is owned by the management account, and uses the management account ID.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AWSCloudTrailAclCheck20150319",
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "cloudtrail.amazonaws.com"
                ]
            },
            "Action": "s3:GetBucketAcl",
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
            "Condition": {
                "StringEquals": {
                    "aws:SourceArn":
 "arn:aws:cloudtrail:region:managementAccountID:trail/trailName"
                }
            }
        },
        {
            "Sid": "AWSCloudTrailWrite20150319",
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "cloudtrail.amazonaws.com"
                ]
            },
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/AWSLogs/managementAccountID/
*",
            "Condition": {
                "StringEquals": {
                    "s3:x-amz-acl": "bucket-owner-full-control",
                    "aws:SourceArn":
 "arn:aws:cloudtrail:region:managementAccountID:trail/trailName"
                }
            }
        },
        {
            "Sid": "AWSCloudTrailOrganizationWrite20150319",
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "cloudtrail.amazonaws.com"
                ]
            },
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/AWSLogs/o-organizationID/*",
            "Condition": {
                "StringEquals": {
                    "s3:x-amz-acl": "bucket-owner-full-control",
                    "aws:SourceArn":
 "arn:aws:cloudtrail:region:managementAccountID:trail/trailName"
                }
            }
        }
    ]
}
```

This example policy does not allow any users from member accounts to access the log files created for the organization. By default, organization log files are accessible only to the management account. For information about how to allow read access to the Amazon S3 bucket for IAM users in member accounts, see Sharing CloudTrail log files between AWS accounts.

Enabling CloudTrail as a trusted service in AWS Organizations

Before you can create an organization trail, you must first enable all features in Organizations. For more information, see Enabling All Features in Your Organization, or run the following command using a profile with sufficient permissions in the management account:

```
aws organizations enable-all-features
```

After you enable all features, you must configure Organizations to trust CloudTrail as a trusted service. .

To create the trusted service relationship between AWS Organizations and CloudTrail, open a terminal or command line and use a profile in the management account. Run the `aws organizations enable-aws-service-access` command, as demonstrated in the following example.

```
aws organizations enable-aws-service-access --service-principal
 cloudtrail.amazonaws.com
```

Using create-trail

Creating an organization trail that applies to all Regions

To create an organization trail that applies to all Regions, add the `--is-organization-trail` and `--is-multi-region-trail` options.

> **Note**
>
> When you create an organization trail with the AWS CLI, you must use an AWS CLI profile in the management account or delegated administrator account with sufficient permissions.

The following example creates an organization trail that delivers logs from all Regions to an existing bucket named *DOC-EXAMPLE-BUCKET*:

```
aws cloudtrail create-trail --name my-trail --s3-bucket-name DOC-EXAMPLE-BUCKET --is-
organization-trail --is-multi-region-trail
```

To confirm that your trail exists in all Regions, the `IsOrganizationTrail` and `IsMultiRegionTrail` parameters in the output are both set to `true`:

```
{
    "IncludeGlobalServiceEvents": true,
    "Name": "my-trail",
    "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/my-trail",
    "LogFileValidationEnabled": false,
    "IsMultiRegionTrail": true,
    "IsOrganizationTrail": true,
    "S3BucketName": "DOC-EXAMPLE-BUCKET"
}
```

> **Note**
>
> Run the `start-logging` command to start logging for your trail. For more information, see Stopping and starting logging for a trail.

Creating an organization trail as a single-Region trail

The following command creates an organization trail that only logs events in a single AWS Region, also known as a single-Region trail. The AWS Region where events are logged is the Region specified in the configuration profile for the AWS CLI.

```
aws cloudtrail create-trail --name my-trail --s3-bucket-name DOC-EXAMPLE-BUCKET --is-
organization-trail
```

For more information, see Naming requirements for CloudTrail resources, S3 buckets, and KMS keys.

Sample output:

```
{
    "IncludeGlobalServiceEvents": true,
    "Name": "my-trail",
    "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/my-trail",
    "LogFileValidationEnabled": false,
    "IsMultiRegionTrail": false,
    "IsOrganizationTrail": true,
    "S3BucketName": "DOC-EXAMPLE-BUCKET"
}
```

By default, the `create-trail` command creates a single-Region trail that does not enable log file validation.

> **Note**
>
> Run the `start-logging` command to start logging for your trail.

Running update-trail to update an organization trail

You can run the `update-trail` command to change the configuration settings for an organization trail, or to apply an existing trail for a single AWS account to an entire organization. Remember that you can run the `update-trail` command only from the Region in which the trail was created.

> **Note**
>
> If you use the AWS CLI or one of the AWS SDKs to update a trail, be sure that the trail's bucket policy is up-to-date. For more information, see Creating a trail for an organization with the AWS CLI.
> When you update an organization trail with the AWS CLI, you must use an AWS CLI profile in the management account or delegated administrator account with sufficient permissions. If you want to convert an organization trail to a non-organization trail, you must use the management account for the organization, because the management account is the owner of all organization resources.

CloudTrail updates organization trails in member accounts even if a resource validation fails. Examples of validation failures include:

- an incorrect Amazon S3 bucket policy
- an incorrect Amazon SNS topic policy
- inability to deliver to a CloudWatch Logs log group
- insufficient permission to encrypt using a KMS key

A member account with CloudTrail permissions can see any validation failures for an organization trail by viewing the trail's details page on the CloudTrail console, or by running the AWS CLI get-trail-status command.

Applying an existing trail to an organization

To change an existing trail so that it also applies to an organization instead of a single AWS account, add the `--is-organization-trail` option, as shown in the following example.

Note

Use the management account to change an existing non-organization trail to an organization trail.

```
aws cloudtrail update-trail --name my-trail --is-organization-trail
```

To confirm that the trail now applies to the organization, the `IsOrganizationTrail` parameter in the output has a value of `true`.

```
{
    "IncludeGlobalServiceEvents": true,
    "Name": "my-trail",
    "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/my-trail",
    "LogFileValidationEnabled": false,
    "IsMultiRegionTrail": true,
    "IsOrganizationTrail": true,
    "S3BucketName": "DOC-EXAMPLE-BUCKET"
}
```

In the preceding example, the trail was configured to apply to all Regions ("IsMultiRegionTrail": true). A trail that applied only to a single Region would show "IsMultiRegionTrail": false in the output.

Converting an organization trail that applies to one Region to apply to all Regions

To change an existing organization trail so that it applies to all Regions, add the --is-multi-region-trail option as shown in the following example.

```
aws cloudtrail update-trail --name my-trail --is-multi-region-trail
```

To confirm that the trail now applies to all Regions, the IsMultiRegionTrail parameter in the output has a value of true.

```
{
    "IncludeGlobalServiceEvents": true,
    "Name": "my-trail",
    "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/my-trail",
    "LogFileValidationEnabled": false,
    "IsMultiRegionTrail": true,
    "IsOrganizationTrail": true,
    "S3BucketName": "DOC-EXAMPLE-BUCKET"
}
```

Troubleshooting issues with an organization trail

This section provides information about how to troubleshoot issues with an organization trail.

Topics

CloudTrail is not delivering events

If CloudTrail is not delivering CloudTrail log files to the Amazon S3 bucket

Check if there is an issue with the S3 bucket.

- From the CloudTrail console, check the trail's details page. If there's an issue with the S3 bucket, the details page includes a warning that delivery to the S3 bucket failed.

- From the AWS CLI, run the get-trail-status command. If there's a failure, the command output includes the `LatestDeliveryError` field, which displays any Amazon S3 error that CloudTrail encountered when attempting to deliver log files to the designated bucket. This error occurs only when there is a problem with the destination S3 bucket, and does not occur for requests that time out. To resolve the issue, fix the bucket policy so that CloudTrail can write to the bucket; or create a new bucket, and then call `update-trail` to specify the new bucket. For information about the organization bucket policy, see Create or update an Amazon S3 bucket to use to store the log files for an organization trail.

If CloudTrail is not delivering logs to CloudWatch Logs

Check if there is an issue with the configuration of the CloudWatch Logs role policy.

- From the CloudTrail console, check the trail's details page. If there's an issue with CloudWatch Logs, the details page includes a warning that indicates CloudWatch Logs delivery failed.
- From the AWS CLI, run the get-trail-status command. If there's a failure, the command output includes the `LatestCloudWatchLogsDeliveryError` field, which displays any CloudWatch Logs error that CloudTrail encountered when attempting to deliver logs to CloudWatch Logs. To resolve the issue, fix the CloudWatch Logs role policy. For information about the CloudWatch Logs role policy, see Role policy document for CloudTrail to use CloudWatch Logs for monitoring.

If you're not seeing activity for a member account in an organization trail

If you're not seeing activity for a member account in an organization trail, check the following:

- **Check the home Region for the trail to see if it is an opt-in Region**

  Although most AWS Regions are enabled by default for your AWS account, you must manually enable certain Regions (also referred to as *opt-in Regions*). For information about which Regions are enabled by default, see Considerations before enabling and disabling Regions in the *AWS Account Management Reference Guide*. For the list of Regions CloudTrail supports, see CloudTrail supported Regions.

  If the organization trail is multi-Region and the home Region is an opt-in Region, member accounts will not send activity to the organization trail unless they opt into the AWS Region where the multi-Region trail was created. For example, if you create a multi-Region trail and choose the Europe (Spain) Region as the home Region for the trail, only member accounts that enabled the Europe (Spain) Region for their account will send their account activity to the organization trail. To resolve the issue, enable the opt-in Region in each member account in your organization. For information about enabling an opt-in Region, see Enable or disable a Region in your organization in the *AWS Account Management Reference Guide*.

- **Check if the organization resource-based policy conflicts with the CloudTrail service-linked role policy**

  CloudTrail uses the service-linked role named `AWSServiceRoleForCloudTrail` to support organization trails. This service-linked role allows CloudTrail to perform actions on organization resources, such as `organizations:DescribeOrganization`. If the organization's resource-based policy denies an action that is allowed in the service-linked role policy, CloudTrail will not be able to perform the action even though it is allowed in the service-linked role policy. To resolve the issue, fix the organization's resource-based policy so that it doesn't deny actions that are allowed in the service-linked role policy.

CloudTrail is not sending Amazon SNS notifications for a member account in an organization

When a member account with an AWS Organizations organization trail is not sending Amazon SNS notifications, there could be an issue with the configuration of the SNS topic policy. CloudTrail creates organization trails in member accounts even if a resource validation fails, for example, the organization trail's SNS topic does not include all member account IDs. If the SNS topic policy is incorrect, an authorization failure occurs.

To check whether a trail's SNS topic policy has an authorization failure:

- From the CloudTrail console, check the trail's details page. If there's an authorization failure, the details page includes a warning `SNS authorization failed` and indicates to fix the SNS topic policy.
- From the AWS CLI, run the get-trail-status command. If there's an authorization failure, the command output includes the `LastNotificationError` field with a value of `AuthorizationError`. To resolve the issue, fix the Amazon SNS topic policy. For information about the Amazon SNS topic policy, see Amazon SNS topic policy for CloudTrail.

For more information about SNS topics and subscribing to them, see Getting started with Amazon SNS in the *Amazon Simple Notification Service Developer Guide*.

Viewing CloudTrail Insights events for trails

After you enable CloudTrail Insights on a trail, you can view up to 90 days of Insights events by using the CloudTrail console or the AWS CLI. This section describes how to view, look up, and download a file of Insights events. For information about using the `LookupEvents` API to retrieve information from CloudTrail events, see the AWS CloudTrail API Reference. For more information about CloudTrail Insights, see Logging Insights events in this guide.

For information about how to create a trail, see Creating a trail with the CloudTrail console and Getting and viewing your CloudTrail log files.

> **Note**
>
> To log Insights events on API call volume, the trail must log `write` management events. To log Insights events on API error rate, the trail must log `read` or `write` management events.

Topics

- Viewing CloudTrail Insights events for trails with the console
- Viewing CloudTrail Insights events for trails with the AWS CLI

Viewing CloudTrail Insights events for trails with the console

After you enable CloudTrail Insights events on a trail, when CloudTrail detects unusual API or error rate activity, CloudTrail generates Insights events and displays them on the **Dashboard** and **Insights** pages in the AWS Management Console. You can view the Insights events in the console and troubleshoot the unusual activity. The most recent 90 days of Insights events are shown in the console. You can also download Insights events by using the AWS CloudTrail console. You can programmatically look up events by using the AWS SDKs or AWS Command Line Interface. For more information about CloudTrail Insights events, see Logging Insights events in this guide.

> **Note**
>
> To log Insights events on API call volume, the trail must log `write` management events. To log Insights events on API error rate, the trail must log `read` or `write` management events.

After Insights events are logged, the events are shown on the **Insights** page for 90 days. You cannot manually delete events from the **Insights** page. Because you must create a trail before you can enable CloudTrail Insights, you can view Insights events that are logged to your trail for as long as you store them in the S3 bucket that is configured in your trail settings.

Monitor your trail logs and be notified when specific Insights events activity occurs with Amazon CloudWatch Logs. For more information, see Monitoring CloudTrail Log Files with Amazon CloudWatch Logs.

To view Insights events

CloudTrail Insights events must be enabled on your trail to see Insights events in the console. Allow up to 36 hours for CloudTrail to deliver the first Insights events, if unusual activity is detected.

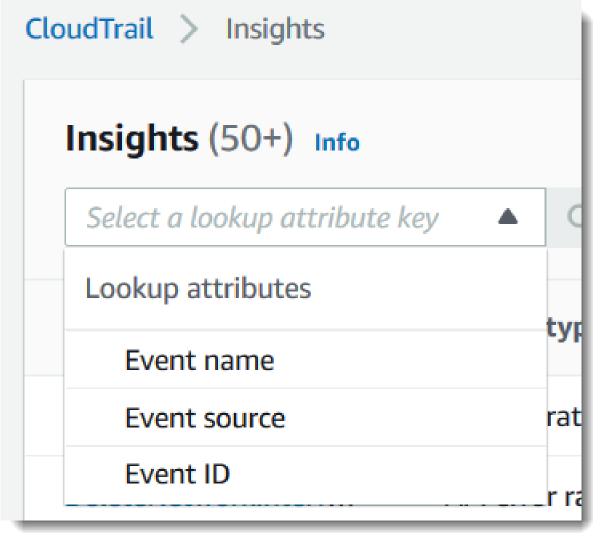
1. Sign in to the AWS Management Console and open the CloudTrail console at https://console.aws.amazon.com/cloudtrail/home/.
2. In the navigation pane, choose **Dashboard** to see the five most recent Insights events, or **Insights** to see all Insights events logged in your account in the last 90 days.

   On the **Insights** page, you can filter Insights events by criteria including event API source, event name, and event ID, and limit the events displayed to those occurring within a specific time range. For more information about filtering Insights events, see Filtering Insights events.

Contents

Filtering Insights events

The default display of events in **Insights** shows events in reverse chronological order. The newest Insights events, sorted by event start time, are at the top. The following list describes the available attributes. You can filter on the first three attributes: **Event name**, **Event source**, and **Event ID**.



Event name

The name of the event, typically the AWS API on which unusual levels of activity were recorded.

Insight type

The type of CloudTrail Insights event, which is either **API call rate** or **API error rate**. The **API call rate** insight type analyzes write-only management API calls that are aggregated per minute against a baseline API call volume. The **API error rate** insight type analyzes management API calls that result in error codes. The error is shown if the API call is unsuccessful.

Event source

The AWS service to which the request was made, such as `iam.amazonaws.com` or `s3.amazonaws.com`. You can scroll through a list of event sources after you choose the **Event source** filter.

Event ID

The ID of the Insights event. Event IDs are not shown in the **Insights** page table, but they are an attribute on which you can filter Insights events. The event IDs of management events that are analyzed to generate Insights events are different from the event IDs of Insights events.

Event start time

The start time of the Insights event, measured as the first minute in which unusual activity was recorded. This attribute is shown in the **Insights** table, but you cannot filter on event start time in the console.

Baseline average

The normal pattern of API call rate or error rate activity. The baseline average is calculated over the seven days preceding the start of an Insights event. Though the value of the baseline duration—the period that CloudTrail analyzes for normal activity on APIs—is approximately seven days, CloudTrail rounds the baseline duration to a whole integer day, so the exact baseline duration can vary.

Insight average

The average number of calls to an API, or the average number of a specific error that was returned on calls to an API, that triggered the Insights event. The CloudTrail Insights average for the start event is the rate of occurrences that triggered the Insights event. Typically, this is the first minute of unusual activity. The Insights average for the end event is the rate of occurrences over the duration of the unusual activity, between the start Insights event and the end Insights event.

Rate change

The difference between the value of **Baseline average** and **Insight average**, measured as a percentage. For example, if the baseline average of an `AccessDenied` error occurring is 1.0, and the Insight average is 3.0, the rate change is 300%. A rate change for an Insight average that exceeds a baseline average shows an up-arrow next to the value. If the Insights event was logged because the activity is below the baseline average, **Rate change** shows a down-arrow next to the percentage.

If there are no events logged for the attribute or time that you choose, the results list is empty. You can apply only one attribute filter in addition to the time range. If you choose a different attribute filter, your specified time range is preserved.

The following steps describe how to filter by attribute.

To filter by attribute

1. To filter the results by an attribute, choose a lookup attribute from the drop-down menu, and then type or choose a value in the **Enter a lookup value** box.
2. To remove an attribute filter, choose the **X** on the right of the attribute filter box.

The following steps describe how to filter by a start and end date and time.

To filter by a start and end date and time

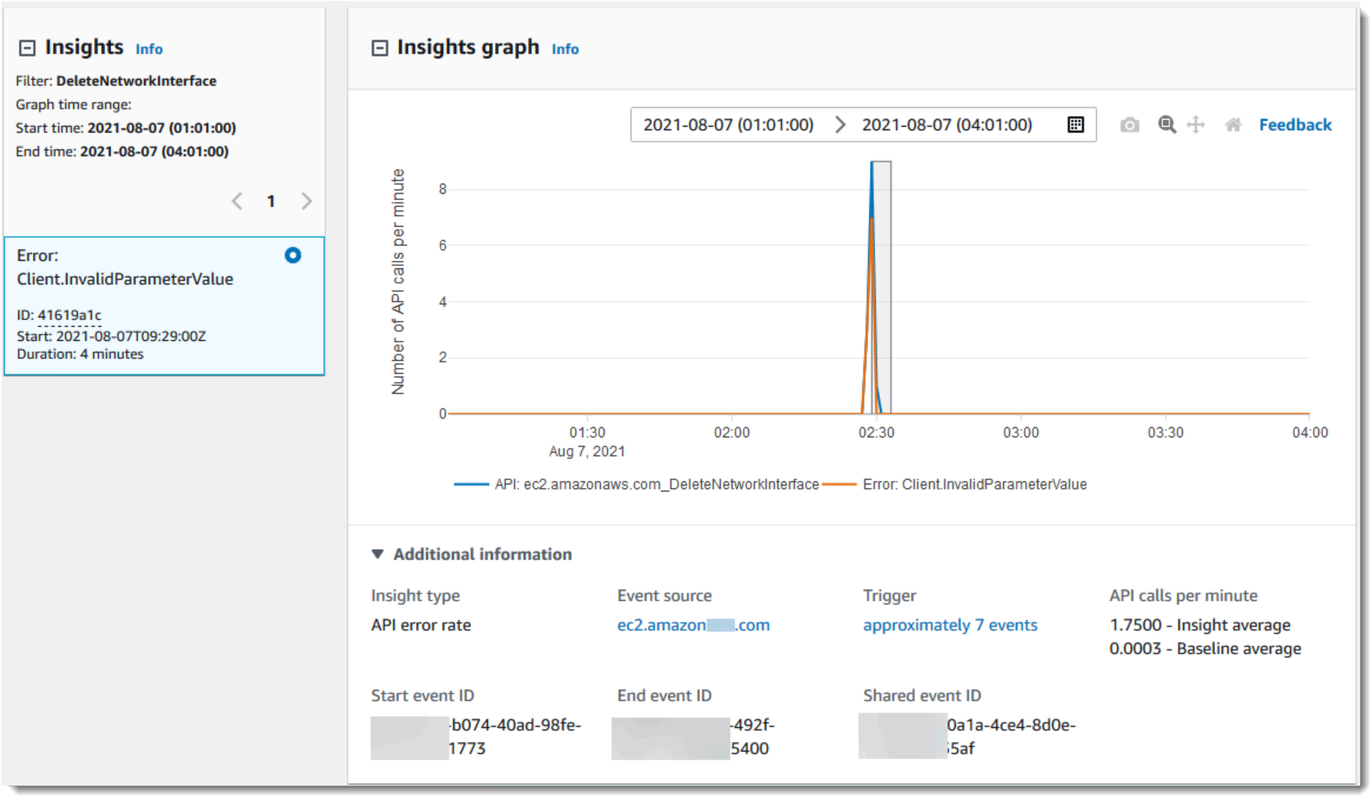
1. To narrow the time range for the events that you want to see, choose a time range on the time span bar at the top of the table. Preset time ranges include 30 minutes, 1 hour, 3 hours, or 12 hours. To specify a custom time range, choose **Custom**.
2. Choose one of the following tabs.
   - **Absolute** - Lets you choose a specific time. Go on to the next step.
   - **Relative to selected event** - Selected by default. Lets you choose a time period relative to the start time of an Insights event. Go on to step 4.
3. To set an **Absolute** time range, do the following.
   a. On the **Absolute** tab, choose the day that you want the time range to start. Enter a start time on the selected day. To enter a date manually, type the date in the format `yyyy/mm/dd`. The start and end times use a 24-hour clock, and values must be in the format `hh:mm:ss`. For example, to indicate a 6:30 p.m. start time, enter **`18:30:00`**.

   b. Choose an end date for the range on the calendar, or specify an end date and time below the calendar. Choose **Apply**.
4. To set a **Relative to selected event** time range, do the following.
   a. Choose a preset time period relative to the start time of Insights events. Preset values are available in minutes, hours, days, or weeks. The maximum relative time period is 12 weeks.

   b. If needed, customize the preset value in the boxes below the presets. Choose **Clear** to reset your changes if needed. When you have set the relative time that you want, choose **Apply**.
5. In **To**, choose the day and specify the time that you want to be the end of the time range. Choose **Apply**.
6. To remove a time range filter, choose the calendar icon on the right of the **Time range** box, and then choose **Remove**.

Viewing Insights events details

1. Choose an Insights event in the results list to show its details. The details page for an Insights event shows a graph of the unusual activity timeline.



⊟ **Insights** Info

Filter: **DeleteNetworkInterface**

Graph time range:

Start time: **2021-08-07 (01:01:00)**

End time: **2021-08-07 (04:01:00)**

< **1** >

Error: Client.InvalidParameterValue ●

ID: 41619a1c

Start: 2021-08-07T09:29:00Z

Duration: 4 minutes

⊟ **Insights graph** Info

2021-08-07 (01:01:00) > 2021-08-07 (04:01:00)

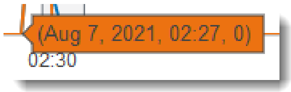
Feedback
Number of API calls per minute
8
6
4
2
0
01:30
Aug 7, 2021
02:00
02:30
03:00
03:30
04:00
API: ec2.amazonaws.com_DeleteNetworkInterface
Error: Client.InvalidParameterValue

▼ **Additional information**

Insight type	Event source	Trigger	API calls per minute
API error rate	ec2.amazon[redacted].com	approximately 7 events	1.7500 - Insight average 0.0003 - Baseline average

Start event ID	End event ID	Shared event ID
[redacted]-b074-40ad-98fe-[redacted]1773	[redacted]-492f-[redacted]5400	[redacted]0a1a-4ce4-8d0e-[redacted]5af

2. Hover over the highlighted bands to show the start time and duration of each Insights event in the graph.



The following information is shown in the **Additional information** area of the graph:

- **Insight type**. This can be API call rate or API error rate.
- **Trigger**. This is a link to the **Cloudtrail events** tab, which lists the management events that were analyzed to determine that unusual activity occurred.
- **API calls per minute**
  - **Baseline average** - The typical rate of occurrences per minute on the API on which the Insights event was logged, as measured within approximately the preceding seven days, in a specific Region in your account.
  - **Insights average** - The rate of occurrences per minute on this API that triggered the Insights event. The CloudTrail Insights average for the start event is the rate of calls or errors per minute on the API that triggered the Insights event. Typically, this is the first minute of unusual activity. The Insights average for the end event is the rate of API calls

or errors per minute over the duration of the unusual activity, between the start Insights event and the end Insights event.

- **Event source**. The AWS service endpoint on which the unusual number of API calls or errors were logged. In the preceding image, the source is `ec2.amazonaws.com`, which is the service endpoint for Amazon EC2.
- **Event IDs**.
  - **Start event ID** - The ID of the Insights event that was logged at the start of unusual activity.
  - **End event ID** - The ID of the Insights event that was logged at the end of unusual activity.
  - **Shared event ID** - In Insights events, the **Shared event ID** is a GUID that is generated by CloudTrail Insights to uniquely identify a start and end pair of Insights events. **Shared event ID** is common between the start and the end Insights event, and helps to create a correlation between both events to uniquely identify unusual activity.

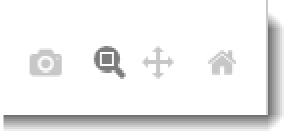
3. Choose the **Attributions** tab to view information about the user identities, user agents, and on API call rate Insights events, error codes correlated with unusual and baseline activity. A maximum of five user identities, five user agents, and five error codes are shown in tables on the **Attributions** tab, sorted by an average of the count of activity, in descending order from highest to lowest.
4. On the **CloudTrail events** tab, view related events that CloudTrail analyzed to determine that unusual activity occurred. By default, a filter is already applied for the Insights event name, which is also the name of the related API. The **CloudTrail events** tab shows CloudTrail management events related to the subject API that occurred between the start time (minus one minute) and end time (plus one minute) of the Insights event.

   As you select other Insights events in the graph, the events shown in the **CloudTrail events** table change. These events help you perform deeper analysis to determine the probable cause of an Insights event and reasons for unusual API activity.

   To show all CloudTrail events that were logged during the Insights event duration, and not only those for the related API, turn off the filter.
5. Choose the **Insights event record** tab to view the Insights start and end events in JSON format.
6. Choosing the linked **Event source** returns you to the **Insights** page, filtered by that event source.

Zoom, pan, and download graph

You can zoom, pan, and reset the axes of the graph on the Insights event details page by using a toolbar in the upper right corner.



From left to right, the command buttons on the graph toolbar do the following:

- **Download plot as a PNG** - Download the graph image shown on the details page, and save it in PNG format.
- **Zoom** - Drag to select an area on the graph that you want to enlarge and see in greater detail.
- **Pan** - Shift the graph to see adjacent dates or times.
- **Reset axes** - Change graph axes back to the original, clearing zoom and pan settings.

Change graph time span settings

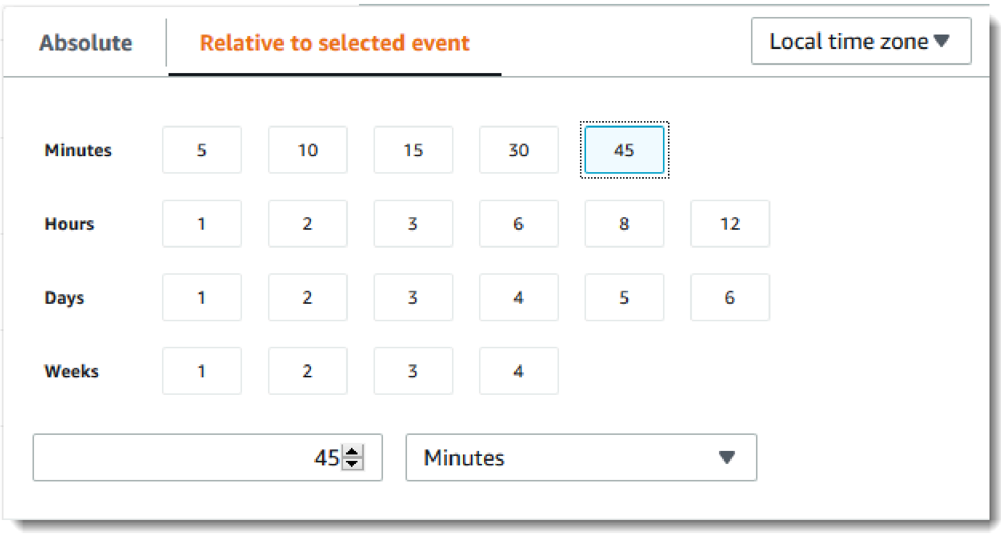
You can change the time span—the selected duration of the events shown on the *x* axis—that is shown in the graph by choosing a setting in the graph's upper right corner.

2020-08-05 (09:50:30) > 2020-08-05 (12:50:30)

The default time span that is shown in the graph depends on the duration of the selected Insights event.

Duration of Insights event	Default time span
Less than 4 hours	**3h** (three hours)
Between 4 and 12 hours	**12h**(12 hours)
Between 12 and 24 hours	**1d** (one day)
Between 24 and 72 hours	**3d** (three days)
Longer than 72 hours	**1w** (one week)

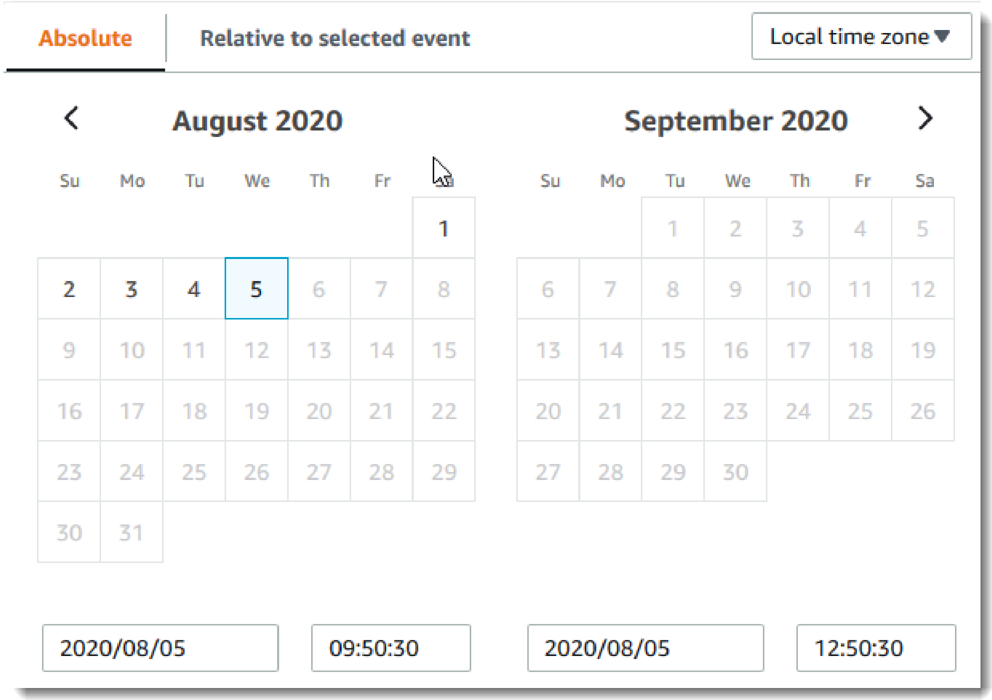
You can choose presets of five minutes, 30 minutes, one hour, three hours, 12 hours, or **Custom**. The following image shows **Relative to selected event** time periods you can choose in **Custom** settings. Relative time periods are approximate time periods surrounding the start and end of the selected Insights event that is displayed on an Insights event details page.



45 Minutes

To customize a selected preset, specify a number and time unit in the boxes below the presets.

To specify an exact date and time range, choose the **Absolute** tab. If you set an absolute date and time range, start and end times are required. For information about how to set the time, see [the section called "Filtering Insights events"]() in this topic.



2020/08/05 09:50:30 2020/08/05 12:50:30

Downloading Insights events

You can download recorded Insights event history as a file in CSV or JSON format. Use filters and time ranges to reduce the size of the file you download.

> **Note**
>
> CloudTrail event history files are data files that contain information (such as resource names) that can be configured by individual users. Some data can potentially be interpreted as commands in programs used to read and analyze this data (CSV injection). For example, when CloudTrail events are exported to CSV and imported to a spreadsheet program, that program might warn you about security concerns. As a security best practice, disable links or macros from downloaded event history files.

1. Specify the filter and time range for events you want to download. For example, you can specify the event name, `StartInstances`, and specify a time range for the last three days of activity.
2. Choose **Download events**, and then choose **Download CSV** or **Download JSON**. You are prompted to choose a location to save the file.

   > **Note**
   >
   > Your download might take some time to finish. For faster results, before you start the download process, use a more specific filter or a shorter time range to narrow the results.
3. After your download is complete, open the file to view the events that you specified.
4. To cancel your download, choose **Cancel download**. If you cancel a download before it is finished, a CSV or JSON file on your local computer might contain only part of your events.

Viewing CloudTrail Insights events for trails with the AWS CLI

You can look up CloudTrail Insights events for the last 90 days by running the `aws cloudtrail lookup-events` command. The `lookup-events` command has the following options:

- `--end-time`

- `--event-category`
- `--max-results`
- `--start-time`
- `--lookup-attributes`
- `--next-token`
- `--generate-cli-skeleton`
- `--cli-input-json`

For general information about using the AWS Command Line Interface, see the AWS Command Line Interface User Guide.

Contents

Prerequisites

- To run AWS CLI commands, you must install the AWS CLI. For more information, see Get started with the AWS CLI.
- Make sure your AWS CLI version is greater than 1.6.6. To verify the CLI version, run **aws --version** on the command line.
- To set the account, Region, and default output format for an AWS CLI session, use the **aws configure** command. For more information, see Configuring the AWS Command Line Interface.

- To log Insights events on API call volume, the trail must log `write` management events. To log Insights events on API error rate, the trail must log `read` or `write` management events.

> **Note**
>
> The CloudTrail AWS CLI commands are case-sensitive.

Getting command line help

To see the command line help for `lookup-events`, type the following command.

```
aws cloudtrail lookup-events help
```

Looking up Insights events

To see the ten latest Insights events, type the following command.

```
aws cloudtrail lookup-events --event-category insight
```

A returned event looks similar to the following example,

```
{
    "NextToken": "kbOt5LlZe+
+mErCebpy2TgaMgmDvF1kYGFcH64JSjIbZFjsuvrSqg66b5YGssKutDYIyII4lrP4IDbeQdiObkp9YAlju3oXd12juEXAMP
    "Events": [
        {
            "eventVersion": "1.07",
            "eventTime": "2019-10-15T21:13:00Z",
            "awsRegion": "us-east-1",
            "eventID": "EXAMPLE-9b6f-45f8-bc6b-9b41c052ebc7",
            "eventType": "AwsCloudTrailInsight",
            "recipientAccountId": "123456789012",
            "sharedEventID": "EXAMPLE8-02b2-4e93-9aab-08ed47ea5fd3",
            "insightDetails": {
            "state": "Start",
            "eventSource": "autoscaling.amazonaws.com",
            "eventName": "CompleteLifecycleAction",
            "insightType": "ApiCallRateInsight",
            "insightContext": {
              "statistics": {
                "baseline": {
                  "average": 0.0000882145
                },
                "insight": {
                  "average": 0.6
                },
                "insightDuration": 5,
                "baselineDuration": 11336
              },
              "attributions": [
                {
                  "attribute": "userIdentityArn",
                  "insight": [
                    {
                      "value": "arn:aws:sts::012345678901:assumed-role/
CodeDeployRole1",
                      "average": 0.2
                    },
                    {
                      "value": "arn:aws:sts::012345678901:assumed-role/
CodeDeployRole2",
                      "average": 0.2
                    },
                    {
                      "value": "arn:aws:sts::012345678901:assumed-role/
CodeDeployRole3",
                      "average": 0.2
                    }
                  ],
                  "baseline": [
                    {
                      "value": "arn:aws:sts::012345678901:assumed-role/
CodeDeployRole1",
                      "average": 0.0000882145
                    }
                  ]
                },
                {
                  "attribute": "userAgent",
                  "insight": [
                    {
                      "value": "codedeploy.amazonaws.com",
                      "average": 0.6
                    }
                  ],
                  "baseline": [
                    {
                      "value": "codedeploy.amazonaws.com",
                      "average": 0.0000882145
                    }
                  ]
                },
                {
                  "attribute": "errorCode",
                  "insight": [
                    {
                      "value": "null",
                      "average": 0.6
                    }
                  ],
                  "baseline": [
                    {
                      "value": "null",
                      "average": 0.0000882145
                    }
                  ]
                }
              ]
            }
          },
            "eventCategory": "Insight"
        },
        {
            "eventVersion": "1.07",
            "eventTime": "2019-10-15T21:14:00Z",
            "awsRegion": "us-east-1",
            "eventID": "EXAMPLEc-9eac-4af6-8e07-26a5ae8786a5",
            "eventType": "AwsCloudTrailInsight",
            "recipientAccountId": "123456789012",
            "sharedEventID": "EXAMPLE8-02b2-4e93-9aab-08ed47ea5fd3",
            "insightDetails": {
            "state": "End",
            "eventSource": "autoscaling.amazonaws.com",
            "eventName": "CompleteLifecycleAction",
            "insightType": "ApiCallRateInsight",
            "insightContext": {
              "statistics": {
                "baseline": {
```

```
         "average": 0.0000882145
        },
        "insight": {
         "average": 0.6
        },
        "insightDuration": 5,
        "baselineDuration": 11336
       },
       "attributions": [
        {
         "attribute": "userIdentityArn",
         "insight": [
          {
           "value": "arn:aws:sts::012345678901:assumed-role/
CodeDeployRole1",
           "average": 0.2
          },
          {
           "value": "arn:aws:sts::012345678901:assumed-role/
CodeDeployRole2",
           "average": 0.2
          },
          {
           "value": "arn:aws:sts::012345678901:assumed-role/
CodeDeployRole3",
           "average": 0.2
          }
         ],
         "baseline": [
          {
           "value": "arn:aws:sts::012345678901:assumed-role/
CodeDeployRole1",
           "average": 0.0000882145
          }
         ]
        },
        {
         "attribute": "userAgent",
         "insight": [
          {
           "value": "codedeploy.amazonaws.com",
           "average": 0.6
          }
         ],
                  "baseline": [
                    {
                      "value": "codedeploy.amazonaws.com",
                      "average": 0.0000882145
                    }
                  ]
                },
                {
                  "attribute": "errorCode",
                  "insight": [
                    {
                      "value": "null",
                      "average": 0.6
                    }
                  ],
                  "baseline": [
                    {
                      "value": "null",
                      "average": 0.0000882145
                    }
                  ]
                }
              ]
            }
          },
            "eventCategory": "Insight"
        }
    ]
}
```

For an explanation of the lookup-related fields in the output, see Lookup output fields in this topic. For an explanation of fields in the Insights event, see CloudTrail record contents.

Specifying the number of Insights events to return

To specify the number of events to return, type the following command.

```
aws cloudtrail lookup-events --event-category insight --max-results <integer>
```

The default value for *<integer>*, if it is not specified, is 10. Possible values are 1 through 50. The following example returns one result.

```
aws cloudtrail lookup-events --event-category insight --max-results 1
```

Looking up Insights events by time range

Insights events from the past 90 days are available for lookup. To specify a time range, type the following command.

```
aws cloudtrail lookup-events --event-category insight --start-time <timestamp> --end-
time <timestamp>
```

`--start-time <timestamp>` specifies, in UTC, that only Insights events that occur after or at the specified time are returned. If the specified start time is after the specified end time, an error is returned.

`--end-time <timestamp>` specifies, in UTC, that only Insights events that occur before or at the specified time are returned. If the specified end time is before the specified start time, an error is returned.

The default start time is the earliest date that data is available within the last 90 days. The default end time is the time of the event that occurred closest to the current time.

All timestamps are shown in UTC.

Looking up Insights events by attribute

To filter by an attribute, type the following command.

```
aws cloudtrail lookup-events --event-category insight --lookup-attributes
 AttributeKey=<attribute>,AttributeValue=<string>
```

You can specify only one attribute key-value pair for each **lookup-events** command. The following are valid Insights event values for `AttributeKey`. Value names are case sensitive.

- `EventId`
- `EventName`
- `EventSource`

The maximum length for the `AttributeValue` is 2000 characters. The following characters ('_', ' ', ',', '\\n') count as two characters towards the 2000 character limit.

Attribute lookup examples

The following example command returns Insights events in which the value of `EventName` is `PutRule`.

```
aws cloudtrail lookup-events --event-category insight --lookup-attributes
 AttributeKey=EventName, AttributeValue=PutRule
```

The following example command returns Insights events in which the value of `EventId` is `b5cc8c40-12ba-4d08-a8d9-2bceb9a3e002`.

```
aws cloudtrail lookup-events --event-category insight --lookup-attributes
 AttributeKey=EventId, AttributeValue=b5cc8c40-12ba-4d08-a8d9-2bceb9a3e002
```

The following example command returns Insights events in which the value of `EventSource` is `iam.amazonaws.com`.

```
aws cloudtrail lookup-events --event-category insight --lookup-attributes
 AttributeKey=EventSource, AttributeValue=iam.amazonaws.com
```

Specifying the next page of results

To get the next page of results from a `lookup-events` command, type the following command.

```
aws cloudtrail lookup-events --event-category insight <same parameters as previous
 command> --next-token=<token>
```

In this command, the value for *<token>* is taken from the first field of the output of the previous command.

When you use `--next-token` in a command, you must use the same parameters as in the previous command. For example, suppose you run the following command.

```
aws cloudtrail lookup-events --event-category insight --lookup-attributes
 AttributeKey=EventName, AttributeValue=PutRule
```

To get the next page of results, your next command would look like the following.

```
aws cloudtrail lookup-events --event-category insight --lookup-attributes
 AttributeKey=EventName,AttributeValue=PutRule --next-token=EXAMPLEZe+
+mErCebpy2TgaMgmDvF1kYGFcH64JSjIbZFjsuvrSqg66b5YGssKutDYIyII4lrP4IDbeQdiObkp9YAlju3oXd12juEXAMP
```

Getting JSON input from a file

The AWS CLI for some AWS services has two parameters, `--generate-cli-skeleton` and `--cli-input-json`, that you can use to generate a JSON template, which you can modify and use as input to the `--cli-input-json` parameter. This section describes how to use these parameters with `aws cloudtrail lookup-events`. For more information, see AWS CLI skeletons and input files.

To look up Insights events by getting JSON input from a file

1. Create an input template for use with `lookup-events` by redirecting the `--generate-cli-skeleton` output to a file, as in the following example.

   ```
   aws cloudtrail lookup-events --event-category insight --generate-cli-skeleton >
    LookupEvents.txt
   ```

   The template file generated (in this case, LookupEvents.txt) looks like the following.

   ```

   {
       "LookupAttributes": [
           {
               "AttributeKey": "",
               "AttributeValue": ""
           }
       ],
       "StartTime": null,
       "EndTime": null,
       "MaxResults": 0,
       "NextToken": ""
   }
   ```

2. Use a text editor to modify the JSON as needed. The JSON input must contain only values that are specified.

⚠ **Important**

All empty or null values must be removed from the template before you can use it.

The following example specifies a time range and maximum number of results to return.

```
{
    "StartTime": "2023-11-01",
    "EndTime": "2023-12-12",
    "MaxResults": 10
}
```

3. To use the edited file as input, use the syntax `--cli-input-json file://<filename>`, as in the following example.

```
aws cloudtrail lookup-events --event-category insight --cli-input-json file://
LookupEvents.txt
```

ⓘ **Note**

You can use other arguments on the same command line as `--cli-input-json`.

Lookup output fields

Events

A list of lookup events based on the lookup attribute and time range that were specified. The events list is sorted by time, with the latest event listed first. Each entry contains information about the lookup request and includes a string representation of the CloudTrail event that was retrieved.

The following entries describe the fields in each lookup event.

CloudTrailEvent

A JSON string that contains an object representation of the event returned. For information about each of the elements returned, see Record Body Contents.

EventId

A string that contains the GUID of the event returned.

EventName

A string that contains the name of the event returned.

EventSource

The AWS service that the request was made to.

EventTime

The date and time, in UNIX time format, of the event.

Resources

A list of resources referenced by the event that was returned. Each resource entry specifies a resource type and a resource name.

ResourceName

A string that contains the name of the resource referenced by the event.

ResourceType

A string that contains the type of a resource referenced by the event. When the resource type cannot be determined, null is returned.

Username

A string that contains the user name of the account for the event returned.

NextToken

A string to get the next page of results from a previous `lookup-events` command. To use the token, the parameters must be the same as those in the original command. If no `NextToken` entry appears in the output, there are no more results to return.

For more information about CloudTrail Insights events, see Logging Insights events in this guide.

Copying trail events to CloudTrail Lake

You can copy existing trail events to a CloudTrail Lake event data store to create a point-in-time snapshot of events logged to the trail. Copying trail events does not interfere with the trail's ability to log events and does not modify the trail in any way.

You can copy trail events to an existing event data store configured for CloudTrail events, or you can create a new CloudTrail event data store and choose the **Copy trail events** option as part of event data store creation. For more information about copying trail events to an existing event data store, see Copy trail events to an existing event data store using the CloudTrail console. For more information about creating a new event data store, see Create an event data store for CloudTrail events with the console.

Copying trail events to a CloudTrail Lake event data store, allows you to run queries on the copied events. CloudTrail Lake queries offer a deeper and more customizable view of events than simple key and value lookups in Event history, or running `LookupEvents`. For more information on CloudTrail Lake, see Working with AWS CloudTrail Lake.

If you are copying trail events to an organization event data store, you must use the management account for the organization. You cannot copy trail events using the delegated administrator account for an organization.

CloudTrail Lake event data stores incur charges. When you create an event data store, you choose the pricing option you want to use for the event data store. The pricing option determines the cost for ingesting and storing events, and the default and maximum retention period for the event data store. For information about CloudTrail pricing and managing Lake costs, see AWS CloudTrail Pricing and Managing CloudTrail Lake costs.

When you copy trail events to a CloudTrail Lake event data store, you incur charges based on the amount of uncompressed data the event data store ingests.

When you copy trail events to CloudTrail Lake, CloudTrail unzips the logs that are stored in gzip (compressed) format and then copies the events contained in the logs to your event data store. The size of the uncompressed data could be greater than the actual S3 storage size. To get a general estimate of the size of the uncompressed data, you can multiply the size of the logs in the S3 bucket by 10.

You can reduce costs by specifying a narrower time range for the copied events. If you are planning to only use the event data store to query your copied events, you can turn off event ingestion to avoid incurring charges on future events. For more information, see AWS CloudTrail Pricing and Managing CloudTrail Lake costs.

Scenarios

The following table describes some common scenarios for copying trail events and how you accomplish each scenario using the console.

Scenario	How do I accomplish this in the console?
Analyze and query historical trail events in CloudTrail Lake without ingesting new events	Create a new event data store and choose the **Copy trail events** option as part of event data store creation. When creating the event data store, deselect **Ingest events** (step 15 of the procedure) to ensure the event data store contains only the historical events for your trail and no future events.
Replace your existing trail with a CloudTrail Lake event data store	Create an event data store with the same event selectors as your trail to ensure that the event data store has the same coverage as your trail. To avoid duplicating events between the source trail and destination event data store, choose a date range for the copied events that is earlier than the creation of the event data store. After your event data store is created, you can turn off logging for the trail to avoid additional charges.

Topics

Considerations for copying trail events

Consider the following factors when copying trail events.

- When copying trail events, CloudTrail uses the S3 `GetObject` API operation to retrieve the trail events in the source S3 bucket. There are some S3 archived storage classes, such as S3 Glacier Flexible Retrieval, S3 Glacier Deep Archive, S3 Outposts, and S3 Intelligent-Tiering Deep Archive tiers that are not accessible by using `GetObject`. To copy trail events stored in these archived storage classes, you must first restore a copy using the S3 `RestoreObject` operation. For information about restoring archived objects, see Restoring Archived Objects in the *Amazon S3 User Guide*.

- When you copy trail events to an event data store, CloudTrail copies all trail events regardless of the configuration of the destination event data store's event types, advanced event selectors, or AWS Region.
- Before copying trail events to an existing event data store, be sure the event data store's pricing option and retention period are configured appropriately for your use case.
  - **Pricing option:** The pricing option determines the cost for ingesting and storing events. For more information about pricing options, see AWS CloudTrail Pricing and Event data store pricing options.
  - **Retention period:** The retention period determines how long event data is kept in the event data store. CloudTrail only copies trail events that have an `eventTime` within the event data store’s retention period. To determine the appropriate retention period, take the sum of the oldest event you want to copy in days and the number of days you want to retain the events in the event data store (**retention period** = *`oldest-event-in-days`* + *`number-days-to-retain`*). For example, if the oldest event you're copying is 45 days old and you want to keep the events in the event data store for a further 45 days, you would set the retention period to 90 days.
- If you are copying trail events to an event data store for investigation and do not want to ingest any future events, you can stop ingestion on the event data store. When creating the event data store, deselect the **Ingest events** option (step 15 of the procedure) to ensure the event data store contains only the historical events for your trail and no future events.
- Before copying trail events, disable any access control lists (ACLs) attached to the source S3 bucket, and update the S3 bucket policy for the destination event data store. For more information about updating the S3 bucket policy, see Amazon S3 bucket policy for copying trail events. For more information about disabling ACLs, see Controlling ownership of objects and disabling ACLs for your bucket.
- CloudTrail only copies trail events from Gzip compressed log files that are in the source S3 bucket. CloudTrail does not copy trail events from uncompressed log files, or log files that were compressed using a format other than Gzip.
- To avoid duplicating events between the source trail and destination event data store, choose a time range for the copied events that is earlier than the creation of the event data store.
- By default, CloudTrail only copies CloudTrail events contained in the S3 bucket's `CloudTrail` prefix and the prefixes inside the `CloudTrail` prefix, and does not check prefixes for other AWS services. If you want to copy CloudTrail events contained in another prefix, you must choose the prefix when you copy trail events.

- To copy trail events to an organization event data store, you must use the management account for the organization. You cannot use the delegated administrator account to copy trail events to an organization event data store.

Required permissions for copying trail events

Before copying trail events, ensure you have all the required permissions for your IAM role. You only need to update the IAM role permissions if you choose an existing IAM role to copy trail events. If you choose to create a new IAM role, CloudTrail provides all necessary permissions for the role.

If the source S3 bucket uses a KMS key for data encryption, ensure that the KMS key policy allows CloudTrail to decrypt data in the bucket. If the source S3 bucket uses multiple KMS keys, you must update each key's policy to allow CloudTrail to decrypt the data in the bucket.

Topics

IAM permissions for copying trail events

When copying trail events, you have the option to create a new IAM role, or use an existing IAM role. When you choose a new IAM role, CloudTrail creates an IAM role with the required permissions and no further action is required on your part.

If you choose an existing role, ensure the IAM role's policies allow CloudTrail to copy trail events from the source S3 bucket. This section provides examples of the required IAM role permission and trust policies.

The following example provides the permissions policy, which allows CloudTrail to copy trail events from the source S3 bucket. Replace *DOC-EXAMPLE-BUCKET*, *myAccountID*, *region*, *prefix*, and *eventDataStoreId* with the appropriate values for your configuration. The *myAccountID* is the AWS account ID used for CloudTrail Lake, which may not be the same as the AWS account ID for the S3 bucket.

Replace *key-region*, *keyAccountID*, and *keyID* with the values for the KMS key used to encrypt the source S3 bucket. You can omit the AWSCloudTrailImportKeyAccess statement if the source S3 bucket does not use a KMS key for encryption.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AWSCloudTrailImportBucketAccess",
      "Effect": "Allow",
      "Action": ["s3:ListBucket", "s3:GetBucketAcl"],
      "Resource": [
        "arn:aws:s3:::DOC-EXAMPLE-BUCKET"
      ],
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "myAccountID",
          "aws:SourceArn":
 "arn:aws:cloudtrail:region:myAccountID:eventdataStore/eventDataStoreId"
         }
       }
    },
    {
      "Sid": "AWSCloudTrailImportObjectAccess",
      "Effect": "Allow",
      "Action": ["s3:GetObject"],
      "Resource": [
        "arn:aws:s3:::DOC-EXAMPLE-BUCKET/prefix",
        "arn:aws:s3:::DOC-EXAMPLE-BUCKET/prefix/*"
      ],
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "myAccountID",
          "aws:SourceArn":
 "arn:aws:cloudtrail:region:myAccountID:eventdataStore/eventDataStoreId"
         }
       }
    },
    {
      "Sid": "AWSCloudTrailImportKeyAccess",
      "Effect": "Allow",
      "Action": ["kms:GenerateDataKey","kms:Decrypt"],
      "Resource": [
        "arn:aws:kms:key-region:keyAccountID:key/keyID"
      ]
    }
  ]
}
```

The following example provides the IAM trust policy, which allows CloudTrail to assume an IAM role to copy trail events from the source S3 bucket. Replace *myAccountID*, *region*, and *eventDataStoreArn* with the appropriate values for your configuration. The *myAccountID* is the AWS account ID used for CloudTrail Lake, which may not be the same as the AWS account ID for the S3 bucket.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "cloudtrail.amazonaws.com"
      },
      "Action": "sts:AssumeRole",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "myAccountID",
          "aws:SourceArn":
 "arn:aws:cloudtrail:region:myAccountID:eventdataStore/eventDataStoreId"
        }
      }
    }
  ]
}
```

Amazon S3 bucket policy for copying trail events

By default, Amazon S3 buckets and objects are private. Only the resource owner (the AWS account that created the bucket) can access the bucket and objects it contains. The resource owner can grant access permissions to other resources and users by writing an access policy.

Before you copy trail events, you must update the S3 bucket policy to allow CloudTrail to copy trail events from the source S3 bucket.

You can add the following statement to the S3 bucket policy to grant these permissions. Replace *roleArn* and *DOC-EXAMPLE-BUCKET* with the appropriate values for your configuration.

```
{
  "Sid": "AWSCloudTrailImportBucketAccess",
  "Effect": "Allow",
  "Action": [
    "s3:ListBucket",
    "s3:GetBucketAcl",
    "s3:GetObject"
  ],
  "Principal": {
    "AWS": "roleArn"
  },
  "Resource": [
    "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
    "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
  ]
},
```

KMS key policy for decrypting data in the source S3 bucket

If the source S3 bucket uses a KMS key for data encryption, ensure the KMS key policy provides CloudTrail with the `kms:Decrypt` and `kms:GenerateDataKey` permissions required to copy trail events from an S3 bucket with SSE-KMS encryption enabled. If your source S3 bucket uses multiple KMS keys, you must update each key's policy. Updating the KMS key policy allows CloudTrail to decrypt data in the source S3 bucket, run validation checks to ensure that events conform to CloudTrail standards, and copy events into the CloudTrail Lake event data store.

The following example provides the KMS key policy, which allows CloudTrail to decrypt the data in the source S3 bucket. Replace *roleArn*, *DOC-EXAMPLE-BUCKET*, *myAccountID*, *region*, and *eventDataStoreId* with the appropriate values for your configuration. The *myAccountID* is the AWS account ID used for CloudTrail Lake, which may not be the same as the AWS account ID for the S3 bucket.

```
{
  "Sid": "AWSCloudTrailImportDecrypt",
  "Effect": "Allow",
  "Action": [
          "kms:Decrypt",
          "kms:GenerateDataKey"
  ],
  "Principal": {
    "AWS": "roleArn"
  },
  "Resource": "*",
  "Condition": {
    "StringLike": {
      "kms:EncryptionContext:aws:s3:arn": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
    },
    "StringEquals": {
      "aws:SourceAccount": "myAccountID",
      "aws:SourceArn":
 "arn:aws:cloudtrail:region:myAccountID:eventdataStore/eventDataStoreId"
    }
  }
}
```

Copy trail events to an existing event data store using the CloudTrail console

Use the following procedure to copy trail events to an existing event data store. For information about how to create a new event data store, see Create an event data store for CloudTrail events with the console.

> **Note**
>
> Before copying trail events to an existing event data store, be sure the event data store's pricing option and retention period are configured appropriately for your use case.
>
> - **Pricing option:** The pricing option determines the cost for ingesting and storing events. For more information about pricing options, see AWS CloudTrail Pricing and Event data store pricing options.
> - **Retention period:** The retention period determines how long event data is kept in the event data store. CloudTrail only copies trail events that have an `eventTime` within the event data store's retention period. To determine the appropriate retention period, take the sum of the oldest event you want to copy in days and the number of days you want to retain the events in the event data store (**retention period** = *`oldest-event-in-`*

> *days* + *number-days-to-retain*). For example, if the oldest event you're copying is 45 days old and you want to keep the events in the event data store for a further 45 days, you would set the retention period to 90 days.

To copy trail events to an event data store

1. Sign in to the AWS Management Console and open the CloudTrail console at https://console.aws.amazon.com/cloudtrail/.
2. Choose **Trails** in the left navigation pane of the CloudTrail console.
3. On the **Trails** page, choose the trail, and then choose **Copy events to Lake**. If the source S3 bucket for the trail uses a KMS key for data encryption, ensure that the KMS key policy allows CloudTrail to decrypt data in the bucket. If the source S3 bucket uses multiple KMS keys, you must update each key's policy to allow CloudTrail to decrypt data in the bucket. For more information about updating the KMS key policy, see KMS key policy for decrypting data in the source S3 bucket.
4. (Optional) By default, CloudTrail only copies CloudTrail events contained in the S3 bucket's `CloudTrail` prefix and the prefixes inside the `CloudTrail` prefix, and does not check prefixes for other AWS services. If you want to copy CloudTrail events contained in another prefix, choose **Enter S3 URI**, and then choose **Browse S3** to browse to the prefix.

   The S3 bucket policy must grant CloudTrail access to copy trail events. For more information about updating the S3 bucket policy, see Amazon S3 bucket policy for copying trail events.
5. For **Specify a time range of events**, choose the time range for copying the events. CloudTrail checks the prefix and log file name to verify the name contains a date between the chosen start and end date before attempting to copy trail events. You can choose a **Relative range** or an **Absolute range**. To avoid duplicating events between the source trail and destination event data store, choose a time range that is earlier than the creation of the event data store.

   > **Note**
   >
   > CloudTrail only copies trail events that have an `eventTime` within the event data store’s retention period. For example, if an event data store’s retention period is 90 days, then CloudTrail will not copy any trail events with an `eventTime` older than 90 days.

- If you choose **Relative range**, you can choose to copy events logged in the last 6 months, 1 year, 2 years, 7 years, or a custom range. CloudTrail copies the events logged within the chosen time period.
- If you choose **Absolute range**, you can choose a specific start and end date. CloudTrail copies the events that occurred between the chosen start and end dates.

6. For **Delivery location**, choose the destination event data store from the drop-down list.
7. For **Permissions**, choose from the following IAM role options. If you choose an existing IAM role, verify that the IAM role policy provides the necessary permissions. For more information about updating the IAM role permissions, see IAM permissions for copying trail events.

   - Choose **Create a new role (recommended)** to create a new IAM role. For **Enter IAM role name**, enter a name for the role. CloudTrail automatically creates the necessary permissions for this new role.
   - Choose **Use a custom IAM role ARN** to use a custom IAM role that is not listed. For **Enter IAM role ARN**, enter the IAM ARN.
   - Choose an existing IAM role from the drop-down list.
8. Choose **Copy events**.
9. You are prompted to confirm the copy. When you are ready to confirm, choose **Copy trail events to Lake**, and then choose **Copy events**.
10. On the **Copy details** page, you can see the copy status and review any failures. When a trail event copy completes, its **Copy status** is set to either **Completed** if there were no errors, or **Failed** if errors occurred.

    > **Note**
    >
    > Details shown on the event copy details page are not in real-time. The actual values for details such as **Prefixes copied** may be higher than what is shown on the page. CloudTrail updates the details incrementally over the course of the event copy.
11. If the **Copy status** is **Failed**, fix any errors shown in **Copy failures**, and then choose **Retry copy**. When you retry a copy, CloudTrail resumes the copy at the location where the failure occurred.

For more information about viewing the details of a trail event copy, see View event copy details with the CloudTrail console.

Getting and viewing your CloudTrail log files

After you create a trail and configure it to capture the log files you want, you need to be able to find the log files and interpret the information they contain.

CloudTrail delivers your log files to an Amazon S3 bucket that you specify when you create the trail. CloudTrail typically delivers logs within an average of about 5 minutes of an API call. This time is not guaranteed. Review the AWS CloudTrail Service Level Agreement for more information. Insights events are typically delivered to your bucket within 30 minutes of unusual activity. After you enable Insights events for the first time, allow up to 36 hours to see the first Insights events, if unusual activity is detected.

> **Note**
>
> If you misconfigure your trail (for example, the S3 bucket is unreachable), CloudTrail will attempt to redeliver the log files to your S3 bucket for 30 days, and these attempted-to-deliver events will be subject to standard CloudTrail charges. To avoid charges on a misconfigured trail, you need to delete the trail.

Topics

- Finding your CloudTrail log files
- Downloading your CloudTrail log files

Finding your CloudTrail log files

CloudTrail publishes log files to your S3 bucket in a gzip archive. In the S3 bucket, the log file has a formatted name that includes the following elements:

- The bucket name that you specified when you created trail (found on the Trails page of the CloudTrail console)
- The (optional) prefix you specified when you created your trail
- The string "AWSLogs"
- The account number
- The string "CloudTrail"

- A Region identifier such as us-west-1
- The year the log file was published in YYYY format
- The month the log file was published in MM format
- The day the log file was published in DD format
- An alphanumeric string that disambiguates the file from others that cover the same time period

The following example shows a complete log file object name:

```
DOC-EXAMPLE-BUCKET/prefix_name/AWSLogs/Account ID/
CloudTrail/region/YYYY/MM/DD/file_name.json.gz
```

Note

For organization trails, the log file object name in the S3 bucket includes the organization unit ID in the path, as follows:

```
DOC-EXAMPLE-BUCKET/prefix_name/AWSLogs/O-ID/Account ID/
CloudTrail/Region/YYYY/MM/DD/file_name.json.gz
```

To retrieve a log file, you can use the Amazon S3 console, the Amazon S3 command line interface (CLI), or the API.

To find your log files with the Amazon S3 console

1. Open the Amazon S3 console.
2. Choose the bucket you specified.
3. Navigate through the object hierarchy until you find the log file you want.

   All log files have a .gz extension.

You will navigate through an object hierarchy that is similar to the following example, but with a different bucket name, account ID, Region, and date.

```

All Buckets
```

```
    DOC-EXAMPLE-BUCKET
        AWSLogs
            123456789012
                CloudTrail
                    us-west-1
                        2014
                            06
                                20
```

A log file for the preceding object hierarchy will look like the following:

```

123456789012_CloudTrail_us-west-1_20140620T1255ZHdkvFTXOA3Vnhbc.json.gz
```

Note

Although uncommon, you may receive log files that contain one or more duplicate events. In most cases, duplicate events will have the same `eventID`. For more information about the `eventID` field, see CloudTrail record contents.

Downloading your CloudTrail log files

Log files are in JSON format. If you have a JSON viewer add-on installed, you can view the files directly in your browser. Double-click the log file name in the bucket to open a new browser window or tab. The JSON displays in a readable format.

CloudTrail log files are Amazon S3 objects. You can use the Amazon S3 console, the AWS Command Line Interface (CLI), or the Amazon S3 API to retrieve log files.

For more information, see Amazon S3 objects overview in the *Amazon Simple Storage Service User Guide*.

The following procedure describes how to download a log file with the AWS Management Console.

To download and read a log file

1. Open the Amazon S3 console at https://console.aws.amazon.com/s3/.
2. Choose the bucket and choose the log file that you want to download.

3. Choose **Download** or **Download as** and follow the prompts to save the file. This saves the file in compressed format.

   > **Note**
   >
   > Some browsers, such as Chrome, automatically extract the log file for you. If your browser does this for you, skip to step 5.

4. Use a product such as 7-Zip to extract the log file.
5. Open the log file in a text editor such as Notepad++.

For more information about the event fields that can appear in a log file entry, see CloudTrail record contents.

AWS partners with third-party specialists in logging and analysis to provide solutions that use CloudTrail output. For more information, see AWS CloudTrail partners.

> **Note**
>
> You can also use the **Event history** feature to look up events for create, update, and delete API activity during the last 90 days.
> For more information, see Working with CloudTrail Event history.

Configuring Amazon SNS notifications for CloudTrail

You can be notified when CloudTrail publishes new log files to your Amazon S3 bucket. You manage notifications using Amazon Simple Notification Service (Amazon SNS).

Notifications are optional. If you want notifications, you configure CloudTrail to send update information to an Amazon SNS topic whenever a new log file has been sent. To receive these notifications, you can use Amazon SNS to subscribe to the topic. As a subscriber you can get updates sent to a Amazon Simple Queue Service (Amazon SQS) queue, which enables you to handle these notifications programmatically.

Topics

Configuring CloudTrail to send notifications

You can configure a trail to use an Amazon SNS topic. You can use the CloudTrail console or the **aws cloudtrail create-trail** CLI command to create the topic. CloudTrail creates the Amazon SNS topic for you and attaches an appropriate policy, so that CloudTrail has permission to publish to that topic.

When you create an SNS topic name, the name must meet the following requirements:

- Between 1 and 256 characters long
- Contain uppercase and lowercase ASCII letters, numbers, underscores, or hyphens

When you configure notifications for a trail that applies to all Regions, notifications from all Regions are sent to the Amazon SNS topic that you specify. If you have one or more Region-specific trails, you must create a separate topic for each Region and subscribe to each individually.

To receive notifications, subscribe to the Amazon SNS topic or topics that CloudTrail uses. You do this with the Amazon SNS console or Amazon SNS CLI commands. For more information, see Subscribing to an Amazon SNS topic in the *Amazon Simple Notification Service Developer Guide*.

Note

CloudTrail sends a notification when log files are written to the Amazon S3 bucket. An active account can generate a large number of notifications. If you subscribe with email or SMS, you can receive a large volume of messages. We recommend that you subscribe using Amazon Simple Queue Service (Amazon SQS), which lets you handle notifications programmatically. For more information, see Subscribing an Amazon SQS queue to an Amazon SNS topic (console) in the *Amazon Simple Queue Service Developer Guide*.

The Amazon SNS notification consists of a JSON object that includes a `Message` field. The `Message` field lists the full path to the log file, as shown in the following example:

```
{
    "s3Bucket": "DOC-EXAMPLE-BUCKET","s3ObjectKey": ["AWSLogs/123456789012/
CloudTrail/us-east-2/2013/12/13/123456789012_CloudTrail_us-
west-2_20131213T1920Z_LnPgDQnpkSKEsppV.json.gz"]
}
```

If multiple log files are delivered to your Amazon S3 bucket, a notification may contain multiple logs, as shown in the following example:

```
{
    "s3Bucket": "DOC-EXAMPLE-BUCKET",
    "s3ObjectKey": [
        "AWSLogs/123456789012/CloudTrail/us-
east-2/2016/08/11/123456789012_CloudTrail_us-
east-2_20160811T2215Z_kpaMYavMQA9Ahp7L.json.gz",
        "AWSLogs/123456789012/CloudTrail/us-
east-2/2016/08/11/123456789012_CloudTrail_us-
east-2_20160811T2210Z_zqDkyQv3TK8ZdLr0.json.gz",
        "AWSLogs/123456789012/CloudTrail/us-
east-2/2016/08/11/123456789012_CloudTrail_us-
east-2_20160811T2205Z_jaMVRa6JfdLCJYHP.json.gz"
    ]
}
```

If you choose to receive notifications by email, the body of the email consists of the content of the `Message` field. For information about the JSON structure, see Fanout to Amazon SQS queues in the *Amazon Simple Notification Service Developer Guide*. Only the `Message` field shows CloudTrail information. The other fields contain information from the Amazon SNS service.

If you create a trail with the CloudTrail API, you can specify an existing Amazon SNS topic that you want CloudTrail to send notifications to with the `CreateTrail` or `UpdateTrail` operations. You must make sure that the topic exists and that it has permissions that allow CloudTrail to send notifications to it. See Amazon SNS topic policy for CloudTrail.

Additional resources

For more information about Amazon SNS topics and about subscribing to them, see the *Amazon Simple Notification Service Developer Guide*.

Using AWS CloudTrail with interface VPC endpoints

If you use Amazon Virtual Private Cloud (Amazon VPC) to host your AWS resources, you can establish a private connection between your VPC and AWS CloudTrail. You can use this connection to enable CloudTrail to communicate with your resources on your VPC without going through the public internet.

Amazon VPC is an AWS service that you can use to launch AWS resources in a virtual network that you define. With a VPC, you have control over your network settings, such the IP address range, subnets, route tables, and network gateways. With VPC endpoints, the routing between the VPC and AWS services is handled by the AWS network, and you can use IAM policies to control access to service resources.

To connect your VPC to CloudTrail, you define an *interface VPC endpoint* for CloudTrail. An interface endpoint is an elastic network interface with a private IP address that serves as an entry point for traffic destined to a supported AWS service. The endpoint provides reliable, scalable connectivity to CloudTrail without requiring an internet gateway, network address translation (NAT) instance, or VPN connection. For more information, see What is Amazon VPC in the *Amazon VPC User Guide*.

Interface VPC endpoints are powered by AWS PrivateLink, an AWS technology that enables private communication between AWS services using an elastic network interface with private IP addresses. For more information, see AWS PrivateLink.

The following steps are for users of Amazon VPC. For more information, see Get started with Amazon VPC in the *Amazon VPC User Guide*.

Availability

CloudTrail currently supports VPC endpoints in the following AWS Regions:

- US East (Ohio)
- US East (N. Virginia)
- US West (N. California)
- US West (Oregon)
- Africa (Cape Town)
- Asia Pacific (Hong Kong)
- Asia Pacific (Hyderabad)
- Asia Pacific (Jakarta)
- Asia Pacific (Melbourne)
- Asia Pacific (Mumbai)
- Asia Pacific (Osaka)
- Asia Pacific (Seoul)

- Asia Pacific (Singapore)
- Asia Pacific (Sydney)
- Asia Pacific (Tokyo)
- Canada (Central)
- Canada West (Calgary)
- Europe (Frankfurt)
- Europe (Ireland)
- Europe (London)
- Europe (Milan)
- Europe (Paris)
- Europe (Spain)
- Europe (Stockholm)
- Europe (Zurich)
- Israel (Tel Aviv)
- Middle East (Bahrain)
- Middle East (UAE)
- South America (São Paulo)
- AWS GovCloud (US-East)
- AWS GovCloud (US-West)

Create a VPC endpoint for CloudTrail

To start using CloudTrail with your VPC, create an interface VPC endpoint for CloudTrail. For more information, see Access an AWS service using an interface VPC endpoint in the *Amazon VPC User Guide*.

You don't need to change the settings for CloudTrail. CloudTrail calls other AWS services using either public endpoints or private interface VPC endpoints, whichever are in use.

Shared subnets

A CloudTrail VPC endpoint, like any other VPC endpoint, can only be created by an owner account in the shared subnet. However, a participant account can use CloudTrail VPC endpoints in subnets that are shared with the participant account. For more information about Amazon VPC sharing, see Share your VPC with other accounts in the *Amazon VPC User Guide*.

Naming requirements for CloudTrail resources, S3 buckets, and KMS keys

This section provides information about the naming requirements for CloudTrail resources, Amazon S3 buckets, and KMS keys.

Topics

- CloudTrail resource naming requirements
- Amazon S3 bucket naming requirements
- AWS KMS alias naming requirements

CloudTrail resource naming requirements

CloudTrail resource names must meet the following requirements:

- Contain only ASCII letters (a-z, A-Z), numbers (0-9), periods (.), underscores (_), or dashes (-).
- Start with a letter or number, and end with a letter or number.
- Be between 3 and 128 characters.
- Have no adjacent periods, underscores or dashes. Names like my-_namespace and my-\-namespace are invalid.
- Not be in IP address format (for example, `192.168.5.4`).

Amazon S3 bucket naming requirements

The Amazon S3 bucket that you use to store CloudTrail log files must have a name that conforms with naming requirements for non-US Standard regions. Amazon S3 defines a bucket name as a series of one or more labels, separated by periods. For a complete list of naming rules, see Bucket naming rules in the *Amazon Simple Storage Service User Guide*.

The following are some of the rules:

- The bucket name can be between 3 and 63 characters long, and can contain only lower-case characters, numbers, periods, and dashes.

- Each label in the bucket name must start with a lowercase letter or number.
- The bucket name cannot contain underscores, end with a dash, have consecutive periods, or use dashes adjacent to periods.
- The bucket name cannot be formatted as an IP address (198.51.100.24).

⚠ **Warning**

Because S3 allows your bucket to be used as a URL that can be accessed publicly, the bucket name that you choose must be globally unique. If some other account has already created a bucket with the name that you chose, you must use another name. For more information, see Bucket restrictions and limitations in the *Amazon Simple Storage Service User Guide*.

AWS KMS alias naming requirements

When you create an AWS KMS key, you can choose an alias to identify it. For example, you might choose the alias "KMS-CloudTrail-us-west-2" to encrypt the logs for a specific trail.

The alias must meet the following requirements:

- Between 1 and 256 characters, inclusive
- Contain alphanumeric characters (A-Z, a-z, 0-9), hyphens (-), forward slashes (/), and underscores (_)
- Cannot begin with **aws**

For more information, see Creating Keys in the *AWS Key Management Service Developer Guide*.

AWS account closure and trails

AWS CloudTrail continuously monitors and records events for account activity generated by any user, role, or AWS service for an AWS account. Users can create a CloudTrail trail to receive a copy of these events in a S3 bucket that they own.

CloudTrail is a foundational security service, therefore, trails created by users continue to exist and deliver events even after an AWS account is closed, unless a user explicitly deletes the trails in their AWS account prior to closing it. This behavior also applies to the organization trails that are created by the management account or the delegated administrator, and to multi-Region organization trails that are then created in the organization's member accounts. This ensures that if a user reopens a closed account that user has an unbroken record of account activity. It also provides users with visibility into any final account activity, including the deletion and termination of remaining account resources and services.

Users have the option to delete trails prior to closing their AWS account, or to contact AWS Support to request trail deletion after their AWS account has been closed.

For more information about closing an AWS account, see Close an AWS account.

> **Note**
>
> If CloudTrail log file validation is enabled, users will continue to receive hourly digest files which indicate if any CloudTrail logs were created or not.
> CloudTrail Lake event data stores, CloudTrail Lake channels for integrations, CloudTrail service-linked channels, and resources created for trails (for example, Amazon CloudWatch Logs log groups and Amazon S3 buckets existing in the closed account), follow standard AWS behavior for account closure and are permanently deleted after the post-closure period (typically 90 days).

Configure CloudTrail settings

You can use the **Settings** page on the CloudTrail console to configure and review CloudTrail settings, such as managing delegated administrators for an AWS Organizations organization and viewing any service-linked channels created for your account.

To access the Settings page

1. Sign in to the AWS Management Console and open the CloudTrail console at https://console.aws.amazon.com/cloudtrail/.
2. Choose **Settings** in the left navigation pane of the CloudTrail console.
3. Review and update your settings as needed.

   The following settings are available:

   - Organization delegated administrators – If you have an AWS Organizations organization, you can view CloudTrail delegated administrators, add delegated administrators (up to three maximum), and remove delegated administrators. Only the organization's management account can add or remove delegated administrators.

     The organization's management account can assign any account within the organization to act as a CloudTrail delegated administrator to manage the organization's trails and event data stores on behalf of the organization.
   - Viewing service-linked channels – You can view any service-linked channels created for your account.

     AWS services can create a service-linked channel to receive CloudTrail events on your behalf. The AWS service creating the service-linked channel configures advanced event selectors for the channel and specifies whether the channel applies to all AWS Regions, or a single AWS Region.

Organization delegated administrator

When you use CloudTrail with an AWS Organizations organization, you can assign any account within the organization to act as a CloudTrail delegated administrator to manage the organization's trails and event data stores on behalf of the organization. A delegated administrator is a member account in an organization that can perform the same administrative tasks (except as noted) in CloudTrail as the management account.

If you choose a delegated administrator, this member account has administrative permissions on all organization trails and event data stores in the organization. Adding a delegated administrator does not alter the management or operation of the organization's trails or event data stores.

The first time you add a delegated administrator in the CloudTrail console, or by using the AWS CLI or CloudTrail API, CloudTrail checks whether the organization’s management account has a service-linked role. If the management account does not have a service-linked role, CloudTrail creates the service-linked role for the management account. For more information about service-linked roles, see Using service-linked roles for AWS CloudTrail.

> **Note**
>
> When you add a delegated administrator using the AWS Organizations CLI or API operation, the service-linked role doesn't get created if it does not exist. The service-linked role is only created when you make a call from the management account directly to the CloudTrail service, such as when you add a delegated administrator or create an organization trail or event data store using the CloudTrail console, AWS CLI or CloudTrail API.

Take note of the following factors that define how the delegated administrator operates in CloudTrail.

The management account remains the owner of any CloudTrail organization resources the delegated administrator creates.

The organization's management account remains the owner of any CloudTrail organization resources the delegated administrator creates, such as trails and event data stores. This provides continuity for the organization in the event the delegated administrator changes.

Removing a delegated administrator account does not delete any CloudTrail organization resources they created.

Organization trails and event data stores created by the delegated administrator are not deleted when you remove the delegated administrator, because the management account always serves as the owner of the CloudTrail organization resources regardless of whether they are created by the delegated administrator or the management account.

An organization can have a maximum of three CloudTrail delegated administrators.

> You can have a maximum of three CloudTrail delegated administrators per organization. For more information about removing a delegated administrator, see Remove a CloudTrail delegated administrator.

The following table shows the capabilities of the management account, delegated administrator accounts, and accounts that are members within the AWS Organizations organization.

Capabilities	Management account	Delegated administrator account	Member accounts
Add or remove delegated administrator accounts.	Yes	No	No
Create an organization trail.	Yes	Yes[1]	No
View a list of organization trails.	Yes	Yes	Yes
Update an organization trail.	Yes	Yes[1, 2]	No
Delete an organization trail.	Yes	Yes	No
Create an organization event data store for CloudTrail events or AWS Config configuration items.	Yes	Yes	No
Enable Insights on an organization event data store.	Yes	No	No
Update an organization event data store.	Yes	Yes[2]	No
Enable Lake query federation on an organization event data store[3].	Yes	Yes	No
Disable Lake query federation on an organization event data store.	Yes	Yes	No

Capabilities	Management account	Delegated administrator account	Member accounts
Delete an organization event data store.	Yes	Yes	No
Copy trail events to an organization event data store.	Yes	No	No
Run queries on organization event data stores.	Yes	Yes	No
View the Lake dashboard for an organization event data store.	Yes	Yes	No

[1] The delegated administrator can only configure a CloudWatch Logs log group using the AWS CLI or CloudTrail `CreateTrail` or `UpdateTrail` API operations. Both the CloudWatch Logs log group and log role must exist in the calling account.

[2] Only the management account can convert an organization trail or event data store to an account-level trail or event data store, or convert an account-level trail or event data store to an organization trail or event data store. These actions are not allowed for the delegated administrator because organization trails and event data stores only exist in the management account. When an organization trail or event data store is converted to an account-level trail or event data store, only the management account has access to the trail or event data store.

[3] Only a single delegated administrator account or the management account can enable federation on an organization event data store. Other delegated administrator accounts can query and share information using the Lake Formation data sharing feature. Any delegated administrator account as well as the organization's management account can disable federation.

Topics

- Required permissions to assign a delegated administrator
- Add a CloudTrail delegated administrator
- Remove a CloudTrail delegated administrator

Required permissions to assign a delegated administrator

When assigning a CloudTrail delegated administrator, you must have the permissions to add and remove the delegated administrator in CloudTrail, as well as certain AWS Organizations API actions and IAM permissions listed in the following policy statement.

You can add the following statement to the end of an IAM policy to grant these permissions:

```
{
    "Sid": "Permissions",
    "Effect": "Allow",
    "Action": [
        "cloudtrail:RegisterOrganizationDelegatedAdmin",
        "cloudtrail:DeregisterOrganizationDelegatedAdmin",
        "organizations:RegisterDelegatedAdministrator",
        "organizations:DeregisterDelegatedAdministrator",
        "organizations:ListAWSServiceAccessForOrganization",
        "iam:CreateServiceLinkedRole",
        "iam:GetRole"
    ],
    "Resource": "*"
}
```

Add a CloudTrail delegated administrator

You can add a delegated administrator to manage an organization's CloudTrail resources, such as trails and event data stores.

You can add a CloudTrail delegated administrator for your AWS organization using the CloudTrail console or the AWS CLI.

Before you add a delegated administrator, be sure they have an account in your organization and you are signed in with the management account for your organization. For information about how to create a new AWS account for your organization, see Creating an AWS account in your organization. For information about how to invite an existing AWS account to your organization, see Inviting an AWS account to join your organization.

CloudTrail console

The following procedure shows you how to add a CloudTrail delegated administrator using the CloudTrail console.

1. Sign in to the AWS Management Console and open the CloudTrail console at https://console.aws.amazon.com/cloudtrail/.
2. Choose **Settings** in the left navigation pane of the CloudTrail console.
3. In the **Organization delegated administrators** section, choose **Register administrator**.
4. Enter the twelve-digit AWS account ID of the account that you want to assign as the CloudTrail delegated administrator for the organization's trails and event data stores.
5. Choose **Register administrator**.

AWS CLI

The following example adds a CloudTrail delegated administrator.

```
aws cloudtrail register-organization-delegated-admin
  --member-account-id="memberAccountId"
```

This command produces no output if it's successful.

Remove a CloudTrail delegated administrator

You can remove a CloudTrail delegated administrator using the CloudTrail console or the AWS CLI.

CloudTrail console

The following procedure shows you how to remove a CloudTrail delegated administrator using the CloudTrail console.

1. Sign in to the AWS Management Console and open the CloudTrail console at https://console.aws.amazon.com/cloudtrail/.
2. Choose **Settings** in the left navigation pane of the CloudTrail console.
3. In the **Organization delegated administrators** section, choose the delegated administrator that you want to remove.
4. Choose **Remove administrator**.
5. Confirm you want to remove the delegated administrator and then choose **Remove administrator**.

AWS CLI

The following command removes a CloudTrail delegated administrator.

```
aws cloudtrail deregister-organization-delegated-admin
  --delegated-admin-account-id="delegatedAdminAccountId"
```

This command produces no output if it's successful.

Viewing service-linked channels

AWS services can create a service-linked channel to receive CloudTrail events on your behalf. The AWS service creating the service-linked channel configures advanced event selectors for the channel and specifies whether the channel applies to all AWS Regions, or a single AWS Region.

Topics

Viewing service-linked channels by using the console

Using the CloudTrail console, you can view information about any CloudTrail service-linked channels created by AWS services. The table is empty if your account does not have any service-linked channels.

Use the following procedure to view information about a service-linked channel.

1. Choose **Settings** in the left navigation pane of the CloudTrail console.
2. From **Service-linked channels**, choose a service-linked channel to view its details.
3. On the details page, review the configured settings for the service-linked channel.

   You can view the following information on the details page.

   - **Channel name** - The full name of the channel. The channel name format is `aws-service-channel/`*`AWS_service_name`*`/slc` where *`AWS_service_name`* represents the name of the AWS service that manages the channel.
   - **Channel ARN** - The ARN of the channel, which you can use in a API request to get details about the channel.

- **All regions** - The value is Yes if the channel is configured for all AWS Regions.
- **AWS service** - The name of the AWS service managing the channel.
- **Management events** - Shows any management events configured for the channel.
- **Data events** - Shows any data events configured for the channel.

Viewing service-linked channels by using the AWS CLI

Using the AWS CLI, you can view information about any CloudTrail service-linked channels created by AWS services.

Topics

- Get a CloudTrail service-linked channel
- List all CloudTrail service-linked channels
- AWS service events on service-linked channels

Get a CloudTrail service-linked channel

The following example AWS CLI command returns information about a specific CloudTrail service-linked channel, including the name of the destination AWS service, any advanced selectors configured for the channel, and whether the channel applies to all Regions or a single Region.

You must specify an ARN or the ID suffix of an ARN for --channel.

```
aws cloudtrail get-channel --channel EXAMPLE-ee54-4813-92d5-999aeEXAMPLE
```

The following is an example response. In this example, AWS_service_name represents the name of the AWS service that created the channel.

```
{
    "ChannelArn": "arn:aws:cloudtrail:us-east-1:111122223333:channel/EXAMPLE-
ee54-4813-92d5-999aeEXAMPLE",
    "Name": "aws-service-channel/AWS_service_name/slc",
    "Source": "CloudTrail",
    "SourceConfig": {
        "ApplyToAllRegions": false,
        "AdvancedEventSelectors": [
            {
                "Name": "Management Events Only",
                "FieldSelectors": [
                    {
                        "Field": "eventCategory",
                        "Equals": [
                            "Management"
                        ]
                    }
                ]
            }
        ]
    },
    "Destinations": [
        {
            "Type": "AWS_SERVICE",
            "Location": "AWS_service_name"
        }
    ]
}
```

List all CloudTrail service-linked channels

The following example AWS CLI command returns information about all CloudTrail service-linked channels that were created on your behalf. Optional parameters include `--max-results`, to specify a maximum number of results that you want the command to return on a single page. If there are more results than your specified `--max-results` value, run the command again adding the returned `NextToken` value to get the next page of results.

```
aws cloudtrail list-channels
```

The following is an example response. In this example, `AWS_service_name` represents the name of the AWS service that created the channel.

```
{
    "Channels": [
        {
            "ChannelArn": "arn:aws:cloudtrail:us-east-1:111122223333:channel/EXAMPLE-
ee54-4813-92d5-999aeEXAMPLE",
            "Name": "aws-service-channel/AWS_service_name/slc"
        }
    ]
}
```

AWS service events on service-linked channels

The AWS service managing the service-linked channel can initiate actions on the service-linked channel (for example, creating or updating a service-linked channel). CloudTrail logs these actions as AWS service events, and delivers these events to the **Event history**, and any active trails and event data stores configured for management events. For these events, the `eventType` field is `AwsServiceEvent`.

The following is an example log file entry of an AWS service event for creation of a service-linked channel.

```
{
   "eventVersion":"1.08",
   "userIdentity":{
      "accountId":"111122223333",
      "invokedBy":"AWS Internal"
   },
   "eventTime":"2022-08-18T17:11:22Z",
   "eventSource":"cloudtrail.amazonaws.com",
   "eventName":"CreateServiceLinkedChannel",
   "awsRegion":"us-east-1",
   "sourceIPAddress":"AWS Internal",
   "userAgent":"AWS Internal",
   "requestParameters":null,
   "responseElements":null,
   "requestID":"564f004c-EXAMPLE",
   "eventID":"234f004b-EXAMPLE",
   "readOnly":false,
   "resources":[
      {
         "accountId":"184434908391",
         "type":"AWS::CloudTrail::Channel",
         "ARN":"arn:aws:cloudtrail:us-east-1:111122223333:channel/7944f0ec-EXAMPLE"
      }
   ],
   "eventType":"AwsServiceEvent",
   "managementEvent":true,
   "recipientAccountId":"111122223333",
   "eventCategory":"Management"
}
```

Understanding CloudTrail events

An event in CloudTrail is the record of an activity in an AWS account. This activity can be an action taken by an IAM identity, or service that is monitorable by CloudTrail. CloudTrail events provide a history of both API and non-API account activity made through the AWS Management Console, AWS SDKs, command line tools, and other AWS services.

CloudTrail log files aren't an ordered stack trace of the public API calls, so events don't appear in any specific order.

There are three types of CloudTrail events:

- Management events
- Data events
- Insights events

By default, trails and event data stores log management events, but not data or Insights events.

All event types use a CloudTrail JSON log format. The log contains information about requests for resources in your account, such as who made the request, the services used, the actions performed, and parameters for the action. The event data is enclosed in a `Records` array.

For information about CloudTrail event record fields, see CloudTrail record contents.

Management events

Management events provide information about management operations that are performed on resources in your AWS account. These are also known as *control plane operations*.

Example management events include:

- Configuring security (for example, AWS Identity and Access Management `AttachRolePolicy` API operations).
- Registering devices (for example, Amazon EC2 `CreateDefaultVpc` API operations).
- Configuring rules for routing data (for example, Amazon EC2 `CreateSubnet` API operations).
- Setting up logging (for example, AWS CloudTrail `CreateTrail` API operations).

Management events can also include non-API events that occur in your account. For example, when a user signs in to your account, CloudTrail logs the `ConsoleLogin` event. For more information, see Non-API events captured by CloudTrail.

By default, CloudTrail trails and CloudTrail Lake event data stores log management events. For more information about logging management events, see Logging management events.

The following example shows a single log record of a management event. In this event, an IAM user named `Mary_Major` ran the **aws cloudtrail start-logging** command to call the CloudTrail `StartLogging` action to start the logging process on a trail named `myTrail`.

```
{
    "eventVersion": "1.09",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EXAMPLE6E4XEGITWATV6R",
        "arn": "arn:aws:iam::123456789012:user/Mary_Major",
        "accountId": "123456789012",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "Mary_Major",
        "sessionContext": {
            "attributes": {
                "creationDate": "2023-07-19T21:11:57Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2023-07-19T21:33:41Z",
    "eventSource": "cloudtrail.amazonaws.com",
    "eventName": "StartLogging",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "aws-cli/2.13.5 Python/3.11.4 Linux/4.14.255-314-253.539.amzn2.x86_64
 exec-env/CloudShell exe/x86_64.amzn.2 prompt/off command/cloudtrail.start-logging",
    "requestParameters": {
        "name": "myTrail"
    },
    "responseElements": null,
    "requestID": "9d478fc1-4f10-490f-a26b-EXAMPLE0e932",
    "eventID": "eae87c48-d421-4626-94f5-EXAMPLEac994",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.2",
        "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
        "clientProvidedHostHeader": "cloudtrail.us-east-1.amazonaws.com"
    },
    "sessionCredentialFromConsole": "true"
}
```

In this next example, an IAM user user named `Paulo_Santos` ran the **aws cloudtrail start-event-data-store-ingestion** command to call the `StartEventDataStoreIngestion` action to start ingestion on an event data store.

```
{
    "eventVersion": "1.09",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EXAMPLEPHCNW5EQV7NA54",
        "arn": "arn:aws:iam::123456789012:user/Paulo_Santos",
        "accountId": "123456789012",
        "accessKeyId": "(AKIAIOSFODNN7EXAMPLE",
        "userName": "Paulo_Santos",
        "sessionContext": {
            "attributes": {
                "creationDate": "2023-07-21T21:55:30Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2023-07-21T21:57:28Z",
    "eventSource": "cloudtrail.amazonaws.com",
    "eventName": "StartEventDataStoreIngestion",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "aws-cli/2.13.1 Python/3.11.4 Linux/4.14.255-314-253.539.amzn2.x86_64
 exec-env/CloudShell exe/x86_64.amzn.2 prompt/off command/cloudtrail.start-event-data-
store-ingestion",
    "requestParameters": {
        "eventDataStore": "arn:aws:cloudtrail:us-
east-1:123456789012:eventdatastore/2a8f2138-0caa-46c8-a194-EXAMPLE87d41"
    },
    "responseElements": null,
    "requestID": "f62a3494-ba4e-49ee-8e27-EXAMPLE4253f",
    "eventID": "d97ca7e2-04fe-45b4-882d-EXAMPLEa9b2c",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.2",
        "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
        "clientProvidedHostHeader": "cloudtrail.us-east-1.amazonaws.com"
    },
    "sessionCredentialFromConsole": "true"
}
```

Data events

Data events provide information about the resource operations performed on or in a resource. These are also known as *data plane operations*. Data events are often high-volume activities.

Example data events include:

- Amazon S3 object-level API activity (for example, `GetObject`, `DeleteObject`, and `PutObject` API operations) on objects in S3 buckets.
- AWS Lambda function execution activity (the `Invoke` API).
- CloudTrail `PutAuditEvents` activity on a CloudTrail Lake channel that is used to log events from outside AWS.
- Amazon SNS `Publish` and `PublishBatch` API operations on topics.

The following table shows the data event types available for trails and event data stores. The **Data event type (console)** column shows the appropriate selection in the console. The **resources.type value** column shows the `resources.type` value that you would specify to include data events of that type in your trail or event data store using the AWS CLI or CloudTrail APIs.

For trails, you can use basic or advanced event selectors to log data events for Amazon S3 objects in general purpose buckets, Lambda functions, and DynamoDB tables (shown in the first three rows of the table). You can use only advanced event selectors to log the data event types shown in the remaining rows.

For event data stores, you can use only advanced event selectors to include data events.

AWS service	Description	Data event type (console)	resources.type value
Amazon DynamoDB	Amazon DynamoDB item-level API activity on tables (for example, PutItem, DeleteItem, and UpdateItem API operations). **Note** For tables with streams enabled, the resources field in the data event contains both AWS::DynamoDB::Stream and AWS::DynamoDB::Table. If you specify AWS::DynamoDB::Table for the resources.type, it will log both DynamoDB	**DynamoDB**	AWS::DynamoDB::Table

AWS service	Description	Data event type (console)	resources.type value
	table and DynamoDB streams events by default. To exclude streams events, add a filter on the `eventName` field.		
AWS Lambda	AWS Lambda function execution activity (the `Invoke` API).	**Lambda**	`AWS::Lambda::Function`
Amazon S3	Amazon S3 object-level API activity (for example, `GetObject` , `DeleteObject` , and `PutObject` API operations) on objects in general purpose buckets.	**S3**	`AWS::S3::Object`

AWS service	Description	Data event type (console)	resources.type value
AWS AppConfig	AWS AppConfig API activity for configuration operations such as calls to `StartConfigurationSession` and `GetLatestConfiguration`.	**AWS AppConfig**	`AWS::AppConfig::Configuration`
AWS B2B Data Interchange	B2B Data Interchange API activity for Transformer operations such as calls to `GetTransformerJob` and `StartTransformerJob`.	**B2B Data Interchange**	`AWS::B2BI::Transformer`
Amazon Bedrock	Amazon Bedrock API activity on an agent alias.	**Bedrock agent alias**	`AWS::Bedrock::AgentAlias`
Amazon Bedrock	Amazon Bedrock API activity on a flow alias.	**Bedrock flow alias**	`AWS::Bedrock::FlowAlias`
Amazon Bedrock	Amazon Bedrock API activity on a guardrail.	**Bedrock guardrail**	`AWS::Bedrock::Guardrail`

AWS service	Description	Data event type (console)	resources.type value
Amazon Bedrock	Amazon Bedrock API activity on a knowledge base.	**Bedrock knowledge base**	`AWS::Bedrock::KnowledgeBase`
Amazon CloudFront	CloudFront API activity on a KeyValueStore.	**CloudFront KeyValueStore**	`AWS::CloudFront::KeyValueStore`
AWS Cloud Map	AWS Cloud Map API activity on a namespace.	**AWS Cloud Map namespace**	`AWS::ServiceDiscovery::Namespace`
AWS Cloud Map	AWS Cloud Map API activity on a service.	**AWS Cloud Map service**	`AWS::ServiceDiscovery::Service`
AWS CloudTrail	CloudTrail `PutAuditEvents` activity on a CloudTrail Lake channel that is used to log events from outside AWS.	**CloudTrail channel**	`AWS::CloudTrail::Channel`
Amazon CloudWatch	Amazon CloudWatch API activity on metrics.	**CloudWatch metric**	`AWS::CloudWatch::Metric`
Amazon CodeWhisperer	Amazon CodeWhisperer API activity on a customization.	**CodeWhisperer customization**	`AWS::CodeWhisperer::Customization`

AWS service	Description	Data event type (console)	resources.type value
Amazon CodeWhisperer	Amazon CodeWhisperer API activity on a profile.	**CodeWhisperer**	`AWS::CodeWhisperer::Profile`
Amazon Cognito	Amazon Cognito API activity on Amazon Cognito identity pools.	**Cognito Identity Pools**	`AWS::Cognito::IdentityPool`
Amazon DynamoDB	Amazon DynamoDB API activity on streams.	**DynamoDB Streams**	`AWS::DynamoDB::Stream`
Amazon Elastic Block Store	Amazon Elastic Block Store (EBS) direct APIs, such as `PutSnapshotBlock` , `GetSnapshotBlock` , and `ListChangedBlocks` on Amazon EBS snapshots.	**Amazon EBS direct APIs**	`AWS::EC2::Snapshot`
Amazon EMR	Amazon EMR API activity on a write-ahead log workspace.	**EMR write-ahead log workspace**	`AWS::EMRWAL::Workspace`
Amazon FinSpace	Amazon FinSpace API activity on environments.	**FinSpace**	`AWS::FinSpace::Environment`

AWS service	Description	Data event type (console)	resources.type value
AWS Glue	AWS Glue API activity on tables that were created by Lake Formation.	**Lake Formation**	`AWS::Glue::Table`
Amazon GuardDuty	Amazon GuardDuty API activity for a detector.	**GuardDuty detector**	`AWS::GuardDuty::Detector`
AWS HealthImaging	AWS HealthImaging API activity on data stores.	**MedicalImaging data store**	`AWS::MedicalImaging::Datastore`
AWS IoT	AWS IoT API activity on certificates.	**IoT certificate**	`AWS::IoT::Certificate`
AWS IoT	AWS IoT API activity on things.	**IoT thing**	`AWS::IoT::Thing`
AWS IoT Greengrass Version 2	Greengrass API activity from a Greengrass core device on a component version. **Note** Greengrass doesn't log access denied events.	**IoT Greengrass component version**	`AWS::GreengrassV2::ComponentVersion`

AWS service	Description	Data event type (console)	resources.type value
AWS IoT Greengrass Version 2	Greengrass API activity from a Greengrass core device on a deployment. **Note** Greengrass doesn't log access denied events.	**IoT Greengrass deployment**	`AWS::GreengrassV2::Deployment`
AWS IoT SiteWise	IoT SiteWise API activity on assets.	**IoT SiteWise asset**	`AWS::IoTSiteWise::Asset`
AWS IoT SiteWise	IoT SiteWise API activity on time series.	**IoT SiteWise time series**	`AWS::IoTSiteWise::TimeSeries`
AWS IoT TwinMaker	IoT TwinMaker API activity on an entity.	**IoT TwinMaker entity**	`AWS::IoTTwinMaker::Entity`
AWS IoT TwinMaker	IoT TwinMaker API activity on a workspace.	**IoT TwinMaker workspace**	`AWS::IoTTwinMaker::Workspace`
Amazon Kendra Intelligent Ranking	Amazon Kendra Intelligent Ranking API activity on rescore execution plans.	**Kendra Ranking**	`AWS::KendraRanking::ExecutionPlan`

AWS service	Description	Data event type (console)	resources.type value
Amazon Keyspaces (for Apache Cassandra)	Amazon Keyspaces API activity on a table.	**Cassandra table**	`AWS::Cassandra::Table`
Amazon Kinesis Data Streams	Kinesis Data Streams API activity on streams.	**Kinesis stream**	`AWS::Kinesis::Stream`
Amazon Kinesis Data Streams	Kinesis Data Streams API activity on stream consumers.	**Kinesis stream consumer**	`AWS::Kinesis::StreamConsumer`
Amazon Kinesis Video Streams	Kinesis Video Streams API activity on video streams, such as calls to `GetMedia` and `PutMedia`.	**Kinesis video stream**	`AWS::KinesisVideo::Stream`
Amazon Machine Learning	Machine Learning API activity on ML models.	**Maching Learning MlModel**	`AWS::MachineLearning::MlModel`
Amazon Managed Blockchain	Amazon Managed Blockchain API activity on a network.	**Managed Blockchain network**	`AWS::ManagedBlockchain::Network`
Amazon Managed Blockchain	Amazon Managed Blockchain JSON-RPC calls on Ethereum nodes, such as `eth_getBalance` or `eth_getBlockByNumber` .	**Managed Blockchain**	`AWS::ManagedBlockchain::Node`

AWS service	Description	Data event type (console)	resources.type value
Amazon Neptune Graph	Data API activities, for example queries, algorithms, or vector search, on a Neptune Graph.	**Neptune Graph**	`AWS::NeptuneGraph::Graph`
AWS Payment Cryptography	AWS Payment Cryptography API activity on aliases.	**Payment Cryptography Alias**	`AWS::PaymentCryptography::Alias`
AWS Payment Cryptography	AWS Payment Cryptography API activity on keys.	**Payment Cryptography Key**	`AWS::PaymentCryptography::Key`
AWS Private CA	AWS Private CA Connector for Active Directory API activity.	**AWS Private CA Connector for Active Directory**	`AWS::PCAConnectorAD::Connector`
AWS Private CA	AWS Private CA Connector for SCEP API activity.	**AWS Private CA Connector for SCEP**	`AWS::PCAConnectorSCEP::Connector`
Amazon Q Apps	Data API activity on Amazon Q Apps.	**Amazon Q Apps**	`AWS::QApps:QApp`
Amazon Q Business	Amazon Q Business API activity on an application.	**Amazon Q Business application**	`AWS::QBusiness::Application`

AWS service	Description	Data event type (console)	resources.type value
Amazon Q Business	Amazon Q Business API activity on a data source.	**Amazon Q Business data source**	`AWS::QBusiness::DataSource`
Amazon Q Business	Amazon Q Business API activity on an index.	**Amazon Q Business index**	`AWS::QBusiness::Index`
Amazon Q Business	Amazon Q Business API activity on a web experience.	**Amazon Q Business web experience**	`AWS::QBusiness::WebExperience`
Amazon RDS	Amazon RDS API activity on a DB Cluster.	**RDS Data API - DB Cluster**	`AWS::RDS::DBCluster`
Amazon S3	Amazon S3 API activity on access points.	**S3 Access Point**	`AWS::S3::AccessPoint`
Amazon S3	Amazon S3 object-level API activity (for example, `GetObject` , `DeleteObject` , and `PutObject` API operations) on objects in directory buckets.	**S3 Express**	`AWS::S3Express::Object`

AWS service	Description	Data event type (console)	resources.type value
Amazon S3	Amazon S3 Object Lambda access points API activity, such as calls to `CompleteMultipartUpload` and `GetObject` .	**S3 Object Lambda**	`AWS::S3ObjectLambda::AccessPoint`
Amazon S3 on Outposts	Amazon S3 on Outposts object-level API activity.	**S3 Outposts**	`AWS::S3Outposts::Object`
Amazon SageMaker	Amazon SageMaker `InvokeEndpointWithResponseStream` activity on endpoints.	**SageMaker endpoint**	`AWS::SageMaker::Endpoint`
Amazon SageMaker	Amazon SageMaker API activity on feature stores.	**SageMaker feature store**	`AWS::SageMaker::FeatureGroup`
Amazon SageMaker	Amazon SageMaker API activity on experiment trial components.	**SageMaker metrics experiment trial component**	`AWS::SageMaker::ExperimentTrialComponent`
Amazon SNS	Amazon SNS `Publish` API operations on platform endpoints.	**SNS platform endpoint**	`AWS::SNS::PlatformEndpoint`

AWS service	Description	Data event type (console)	resources.type value
Amazon SNS	Amazon SNS `Publish` and `PublishBatch` API operations on topics.	**SNS topic**	`AWS::SNS::Topic`
Amazon SQS	Amazon SQS API activity on messages.	**SQS**	`AWS::SQS::Queue`
AWS Step Functions	Step Functions API activity on a state machine.	**Step Functions state machine**	`AWS::StepFunctions::StateMachine`
AWS Supply Chain	AWS Supply Chain API activity on an instance.	**Supply Chain**	`AWS::SCN::Instance`
Amazon SWF	Amazon SWF API activity on domains.	**SWF domain**	`AWS::SWF::Domain`
AWS Systems Manager	Systems Manager API activity on control channels.	**Systems Manager**	`AWS::SSMMessages::ControlChannel`
AWS Systems Manager	Systems Manager API activity on managed nodes.	**Systems Manager managed node**	`AWS::SSM::ManagedNode`
Amazon Timestream	Amazon Timestream `Query` API activity on databases.	**Timestream database**	`AWS::Timestream::Database`

AWS service	Description	Data event type (console)	resources.type value
Amazon Timestream	Amazon Timestream `Query` API activity on tables.	**Timestream table**	`AWS::Timestream::Table`
Amazon Verified Permissions	Amazon Verified Permissions API activity on a policy store.	**Amazon Verified Permissions**	`AWS::VerifiedPermissions::PolicyStore`
Amazon WorkSpaces Thin Client	WorkSpaces Thin Client API activity on a Device.	**Thin Client Device**	`AWS::ThinClient::Device`
Amazon WorkSpaces Thin Client	WorkSpaces Thin Client API activity on an Environment.	**Thin Client Environment**	`AWS::ThinClient::Environment`
AWS X-Ray	X-Ray API activity on traces.	**X-Ray trace**	`AWS::XRay::Trace`

Data events are not logged by default when you create a trail or event data store. To record CloudTrail data events, you must explicitly add the supported resources or resource types for which you want to collect activity. For more information, see Creating a trail with the CloudTrail console and Create an event data store for CloudTrail events with the console.

Additional charges apply for logging data events. For CloudTrail pricing, see AWS CloudTrail Pricing.

The following example shows a single log record of a data event for the Amazon SNS `Publish` action.

```
{
    "eventVersion": "1.09",
    "userIdentity": {
          "type": "AssumedRole",
        "principalId": "EX_PRINCIPAL_ID",
        "arn": "arn:aws:iam::123456789012:user/Bob",
        "accountId": "123456789012",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
        "sessionIssuer": {
            "type": "Role",
            "principalId": "AKIAIOSFODNN7EXAMPLE",
            "arn": "arn:aws:iam::123456789012:role/Admin",
            "accountId": "123456789012",
            "userName": "ExampleUser"
            },
            "attributes": {
                "creationDate": "2023-08-21T16:44:05Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2023-08-21T16:48:37Z",
    "eventSource": "sns.amazonaws.com",
    "eventName": "Publish",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "aws-cli/1.29.16 md/Botocore#1.31.16 ua/2.0 os/
linux#5.4.250-173.369.amzn2int.x86_64 md/arch#x86_64 lang/python#3.8.17 md/
pyimpl#CPython cfg/retry-mode#legacy botocore/1.31.16",
    "requestParameters": {
        "topicArn": "arn:aws:sns:us-east-1:123456789012:ExampleSNSTopic",
        "message": "HIDDEN_DUE_TO_SECURITY_REASONS",
        "subject": "HIDDEN_DUE_TO_SECURITY_REASONS",
        "messageStructure": "json",
        "messageAttributes": "HIDDEN_DUE_TO_SECURITY_REASONS"
    },
    "responseElements": {
        "messageId": "0787cd1e-d92b-521c-a8b4-90434e8ef840"
    },
    "requestID": "0a8ab208-11bf-5e01-bd2d-ef55861b545d",
    "eventID": "bb3496d4-5252-4660-9c28-3c6aebdb21c0",
    "readOnly": false,
    "resources": [{
        "accountId": "123456789012",
        "type": "AWS::SNS::Topic",
                "ARN": "arn:aws:sns:us-east-1:123456789012:ExampleSNSTopic"
    }],
    "eventType": "AwsApiCall",
    "managementEvent": false,
    "recipientAccountId": "123456789012",
    "eventCategory": "Data",
    "tlsDetails": {
        "tlsVersion": "TLSv1.2",
        "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
        "clientProvidedHostHeader": "sns.us-east-1.amazonaws.com"
    }
}
```

The next example shows a single log record of a data event for the Amazon Cognito `GetCredentialsForIdentity` action.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "Unknown"
    },
    "eventTime": "2023-01-19T16:55:08Z",
    "eventSource": "cognito-identity.amazonaws.com",
    "eventName": "GetCredentialsForIdentity",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.4",
    "userAgent": "aws-cli/2.7.25 Python/3.9.11 Darwin/21.6.0 exe/x86_64 prompt/off
 command/cognito-identity.get-credentials-for-identity",
    "requestParameters": {
        "logins": {
            "cognito-idp.us-east-1.amazonaws.com/us-east-1_aaaaaaaaa":
 "HIDDEN_DUE_TO_SECURITY_REASONS"
        },
        "identityId": "us-east-1:1cf667a2-49a6-454b-9e45-23199EXAMPLE"
    },
    "responseElements": {
        "credentials": {
            "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
            "sessionToken": "aAaAaAaAaAaAab1111111111EXAMPLE",
            "expiration": "Jan 19, 2023 5:55:08 PM"
        },
        "identityId": "us-east-1:1cf667a2-49a6-454b-9e45-23199EXAMPLE"
    },
    "requestID": "659dfc23-7c4e-4e7c-858a-1abce884d645",
    "eventID": "6ad1c766-5a41-4b28-b5ca-e223ccb00f0d",
    "readOnly": false,
    "resources": [{
        "accountId": "111122223333",
        "type": "AWS::Cognito::IdentityPool",
        "ARN": "arn:aws:cognito-identity:us-east-1:111122223333:identitypool/us-
east-1:2dg778b3-50b7-565c-0f56-34200EXAMPLE"
    }],
    "eventType": "AwsApiCall",
    "managementEvent": false,
    "recipientAccountId": "111122223333",
    "eventCategory": "Data"
}
```

Insights events

CloudTrail Insights events capture unusual API call rate or error rate activity in your AWS account by analyzing CloudTrail management activity. Insights events provide relevant information, such as the associated API, error code, incident time, and statistics, that help you understand and act on unusual activity. Unlike other types of events captured in a CloudTrail trail or event data store, Insights events are logged only when CloudTrail detects changes in your account's API usage or error rate logging that differ significantly from the account's typical usage patterns.

Examples of activity that might generate Insights events include:

- Your account typically logs no more than 20 Amazon S3 `deleteBucket` API calls per minute, but your account starts to log an average of 100 `deleteBucket` API calls per minute. An Insights event is logged at the start of the unusual activity, and another Insights event is logged to mark the end of the unusual activity.
- Your account typically logs 20 calls per minute to the Amazon EC2 `AuthorizeSecurityGroupIngress` API, but your account starts to log zero calls to `AuthorizeSecurityGroupIngress`. An Insights event is logged at the start of the unusual activity, and ten minutes later, when the unusual activity ends, another Insights event is logged to mark the end of the unusual activity.
- Your account typically logs less than one `AccessDeniedException` error in a seven-day period on the AWS Identity and Access Management API, `DeleteInstanceProfile`. Your account starts to log an average of 12 `AccessDeniedException` errors per minute on the `DeleteInstanceProfile` API call. An Insights event is logged at the start of the unusual error rate activity, and another Insights event is logged to mark the end of the unusual activity.

These examples are provided for illustration purposes only. Your results may vary depending on your use case.

To log CloudTrail Insights events, you must explicitly enable Insights events on a new or existing trail or event data store. For more information about creating a trail, see Creating a trail with the CloudTrail console. For more information about creating an event data store, see Create an event data store for Insights events with the console.

Additional charges apply for Insights events. You will be charged separately if you enable Insights for both trails and event data stores. For more information, see AWS CloudTrail Pricing.

There are two events logged to show unusual activity in CloudTrail Insights: a start event and an end event. The following example shows a single log record of a starting Insights event that occurred when the Application Auto Scaling API `CompleteLifecycleAction` was called an unusual number of times. For Insights events, the value of `eventCategory` is `Insight`. An `insightDetails` block identifies the event state, source, name, Insights type, and context, including statistics and attributions. For more information about the `insightDetails` block, see CloudTrail Insights `insightDetails` element.

```
{
        "eventVersion": "1.08",
        "eventTime": "2023-07-10T01:42:00Z",
        "awsRegion": "us-east-1",
        "eventID": "55ed45c5-0b0c-4228-9fe5-EXAMPLEc3f4d",
        "eventType": "AwsCloudTrailInsight",
        "recipientAccountId": "123456789012",
        "sharedEventID": "979c82fe-14d4-4e4c-aa01-EXAMPLE3acee",
        "insightDetails": {
            "state": "Start",
            "eventSource": "autoscaling.amazonaws.com",
            "eventName": "CompleteLifecycleAction",
            "insightType": "ApiCallRateInsight",
            "insightContext": {
                "statistics": {
                    "baseline": {
                        "average": 9.82222E-5
                    },
                    "insight": {
                        "average": 5.0
                    },
                    "insightDuration": 1,
                    "baselineDuration": 10181
                },
                "attributions": [{
                    "attribute": "userIdentityArn",
                    "insight": [{
                        "value": "arn:aws:sts::123456789012:assumed-role/
CodeDeployRole1",
                        "average": 5.0
                    }, {
                        "value": "arn:aws:sts::123456789012:assumed-role/
CodeDeployRole2",
                        "average": 5.0
                    }, {
                        "value": "arn:aws:sts::123456789012:assumed-role/
CodeDeployRole3",
                        "average": 5.0
                    }],
                    "baseline": [{
                        "value": "arn:aws:sts::123456789012:assumed-role/
CodeDeployRole1",
                        "average": 9.82222E-5
                    }]
                }, {
                    "attribute": "userAgent",
                    "insight": [{
                        "value": "codedeploy.amazonaws.com",
                        "average": 5.0
                    }],
                    "baseline": [{
                        "value": "codedeploy.amazonaws.com",
                        "average": 9.82222E-5
                    }]
                }, {
                    "attribute": "errorCode",
                    "insight": [{
                        "value": "null",
                        "average": 5.0
                    }],
                    "baseline": [{
                        "value": "null",
                        "average": 9.82222E-5
                    }]
                }]
            }
        },
        "eventCategory": "Insight"
    }
```

Logging management events

By default, trails and event data stores log management events and don't include data or Insights events.

Additional charges apply for data or Insights events. For more information, see AWS CloudTrail Pricing.

Contents

Management events

Management events provide visibility into management operations that are performed on resources in your AWS account. These are also known as control plane operations. Example management events include:

- Configuring security (for example, IAM `AttachRolePolicy` API operations)
- Registering devices (for example, Amazon EC2 `CreateDefaultVpc` API operations)
- Configuring rules for routing data (for example, Amazon EC2 `CreateSubnet` API operations)
- Setting up logging (for example, AWS CloudTrail `CreateTrail` API operations)

Management events can also include non-API events that occur in your account. For example, when a user logs in to your account, CloudTrail logs the `ConsoleLogin` event. For more information, see Non-API events captured by CloudTrail.

By default, trails and event data stores are configured to log management events.

> **Note**
>
> The CloudTrail **Event history** feature supports only management events. You cannot exclude AWS KMS or Amazon RDS Data API events from **Event history**; settings that you apply to a trail or event data store do not apply to **Event history**. For more information, see Working with CloudTrail Event history.

Read and write events

When you configure your trail or event data store to log management events, you can specify whether you want read-only events, write-only events, or both.

- **Read**

  Read-only events include API operations that read your resources, but don't make changes. For example, read-only events include the Amazon EC2 `DescribeSecurityGroups` and `DescribeSubnets` API operations. These operations return only information about your Amazon EC2 resources and don't change your configurations.
- **Write**

  Write-only events include API operations that modify (or might modify) your resources. For example, the Amazon EC2 `RunInstances` and `TerminateInstances` API operations modify your instances.

Example: Logging read and write events for separate trails

The following example shows how you can configure trails to split log activity for an account into separate S3 buckets: one bucket receives read-only events and a second bucket receives write-only events.

1. You create a trail and choose an S3 bucket named `DOC-EXAMPLE-BUCKET1` to receive log files. You then update the trail to specify that you want **Read** management events.

2. You create a second trail and choose an S3 bucket named `DOC-EXAMPLE-BUCKET2` to receive log files. You then update the trail to specify that you want **Write** management events.
3. The Amazon EC2 `DescribeInstances` and `TerminateInstances` API operations occur in your account.
4. The `DescribeInstances` API operation is a read-only event and it matches the settings for the first trail. The trail logs and delivers the event to `DOC-EXAMPLE-BUCKET1`.
5. The `TerminateInstances` API operation is a write-only event and it matches the settings for the second trail. The trail logs and delivers the event to `DOC-EXAMPLE-BUCKET2`.

Logging management events with the AWS Management Console

1. Sign in to the AWS Management Console and open the CloudTrail console at https://console.aws.amazon.com/cloudtrail/.
2. To update a trail, open the **Trails** page of the CloudTrail console and choose the trail name.

   To update an event data store, open the **Event data stores** page of the CloudTrail console and choose the event data store name.
3. For **Management events**, choose **Edit**.
   - Choose if you want your trail or event data store to log **Read** events, **Write** events, or both.
   - Choose **Exclude AWS KMS events** to filter AWS Key Management Service (AWS KMS) events out of your trail or event data store. The default setting is to include all AWS KMS events.

     The option to log or exclude AWS KMS events is available only if you log management events on your trail or event data store. If you choose not to log management events, AWS KMS events are not logged, and you cannot change AWS KMS event logging settings.

     AWS KMS actions such as `Encrypt`, `Decrypt`, and `GenerateDataKey` typically generate a large volume (more than 99%) of events. These actions are now logged as **Read** events. Low-volume, relevant AWS KMS actions such as `Disable`, `Delete`, and `ScheduleKey` (which typically account for less than 0.5% of AWS KMS event volume) are logged as **Write** events.

     To exclude high-volume events like `Encrypt`, `Decrypt`, and `GenerateDataKey`, but still log relevant events such as `Disable`, `Delete` and `ScheduleKey`, choose to log **Write** management events, and clear the check box for **Exclude AWS KMS events**.

  - Choose **Exclude Amazon RDS Data API events** to filter Amazon Relational Database Service Data API events out of your trail or event data store. The default setting is to include all Amazon RDS Data API events. For more information about Amazon RDS Data API events, see Logging Data API calls with AWS CloudTrail in the *Amazon RDS User Guide for Aurora*.

4. Choose **Save changes** when you are finished.

Logging management events with the AWS CLI

You can configure your trails or event data stores to log management events using the AWS CLI.

Topics

- Examples: Logging management events for trails
- Examples: Logging management events for event data stores

Examples: Logging management events for trails

To view whether your trail is logging management events, run the `get-event-selectors` command.

```
aws cloudtrail get-event-selectors --trail-name TrailName
```

The following example returns the default settings for a trail. By default, trails log all management events, log events from all event sources, and don't log data events.

```
{
    "TrailARN": "arn:aws:cloudtrail:us-east-1:111122223333:trail/TrailName",
    "AdvancedEventSelectors": [
        {
            "Name": "Management events selector",
            "FieldSelectors": [
                {
                    "Field": "eventCategory",
                    "Equals": [
                        "Management"
                    ]
                }
            ]
        }
    ]
}
```

You can use either basic or advanced event selectors to log management events. You cannot apply both event selectors and advanced event selectors to a trail. If you apply advanced event selectors to a trail, any existing basic event selectors are overwritten. The following sections provide examples of how to log management events using advanced event selectors and basic event selectors.

Topics

- Examples: Logging management events for trails using advanced event selectors
- Examples: Logging management events for trails using basic event selectors

Examples: Logging management events for trails using advanced event selectors

The following example creates an advanced event selector for a trail named *TrailName* to include read-only and write-only management events (by omitting the `readOnly` selector), but to exclude AWS Key Management Service (AWS KMS) events. Because AWS KMS events are treated as management events, and there can be a high volume of them, they can have a substantial impact on your CloudTrail bill if you have more than one trail that captures management events.

If you choose not to log management events, AWS KMS events are not logged, and you cannot change AWS KMS event logging settings.

To start logging AWS KMS events to a trail again, remove the `eventSource` selector, and run the command again.

```
aws cloudtrail put-event-selectors --trail-name TrailName \
--advanced-event-selectors '
[
  {
    "Name": "Log all management events except KMS events",
    "FieldSelectors": [
      { "Field": "eventCategory", "Equals": ["Management"] },
      { "Field": "eventSource", "NotEquals": ["kms.amazonaws.com"] }
    ]
  }
]'
```

The example returns the advanced event selectors that are configured for the trail.

```
{
  "AdvancedEventSelectors": [
    {
      "Name": "Log all management events except KMS events",
      "FieldSelectors": [
        {
          "Field": "eventCategory",
          "Equals": [ "Management" ]
        },
        {
          "Field": "eventSource",
          "NotEquals": [ "kms.amazonaws.com" ]
        }
      ]
    }
  ],
  "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/TrailName"
}
```

To start logging excluded events to a trail again, remove the `eventSource` selector, as shown in the following command.

```
aws cloudtrail put-event-selectors --trail-name TrailName \
--advanced-event-selectors '
[
  {
    "Name": "Log all management events",
    "FieldSelectors": [
      { "Field": "eventCategory", "Equals": ["Management"] }
    ]
  }
]'
```

The next example creates an advanced event selector for a trail named *`TrailName`* to include read-only and write-only management events (by omitting the `readOnly` selector), but to exclude Amazon RDS Data API management events. To exclude Amazon RDS Data API management events, specify the Amazon RDS Data API event source in the string value for the `eventSource` field: `rdsdata.amazonaws.com`.

If you choose not to log management events, Amazon RDS Data API management events are not logged, and you cannot change Amazon RDS Data API event logging settings.

To start logging Amazon RDS Data API management events to a trail again, remove the `eventSource` selector, and run the command again.

```
aws cloudtrail put-event-selectors --trail-name TrailName \
--advanced-event-selectors '
[
  {
    "Name": "Log all management events except Amazon RDS Data API management events",
    "FieldSelectors": [
      { "Field": "eventCategory", "Equals": ["Management"] },
      { "Field": "eventSource", "NotEquals": ["rdsdata.amazonaws.com"] }
    ]
  }
]'
```

The example returns the advanced event selectors that are configured for the trail.

```
{
  "AdvancedEventSelectors": [
    {
      "Name": "Log all management events except Amazon RDS Data API management events",
      "FieldSelectors": [
        {
          "Field": "eventCategory",
          "Equals": [ "Management" ]
        },
        {
          "Field": "eventSource",
          "NotEquals": [ "rdsdata.amazonaws.com" ]
        }
      ]
    }
  ],
  "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/TrailName"
}
```

To start logging excluded events to a trail again, remove the `eventSource` selector, as shown in the following command.

```
aws cloudtrail put-event-selectors --trail-name TrailName \
--advanced-event-selectors '
[
  {
    "Name": "Log all management events",
    "FieldSelectors": [
      { "Field": "eventCategory", "Equals": ["Management"] }
    ]
  }
]'
```

Examples: Logging management events for trails using basic event selectors

To configure your trail to log management events, run the `put-event-selectors` command. The following example shows how to configure your trail to include all management events for two S3 objects. You can specify from 1 to 5 event selectors for a trail. You can specify from 1 to 250 data resources for a trail.

Note

The maximum number of S3 data resources is 250, regardless of the number of event selectors.

```
aws cloudtrail put-event-selectors --trail-name TrailName --event-selectors
 '[{ "ReadWriteType": "All", "IncludeManagementEvents":true, "DataResources":
 [{ "Type": "AWS::S3::Object", "Values": ["arn:aws:s3:::DOC-EXAMPLE-BUCKET/prefix",
 "arn:aws:s3:::DOC-EXAMPLE-BUCKET2/prefix2"] }] }]'
```

The following example returns the event selector configured for the trail.

```
{
    "TrailARN": "arn:aws:cloudtrail:us-east-1:111122223333:trail/TrailName",
    "EventSelectors": [
        {
            "ReadWriteType": "All",
            "IncludeManagementEvents": true,
            "DataResources": [
                {
                    "Type": "AWS::S3::Object",
                    "Values": [
                        "arn:aws:s3:::DOC-EXAMPLE-BUCKET/prefix",
                        "arn:aws:s3:::DOC-EXAMPLE-BUCKET2/prefix2",
                    ]
                }
            ],
            "ExcludeManagementEventSources": []
        }
    ]
}
```

To exclude AWS Key Management Service (AWS KMS) events from a trail's logs, run the `put-event-selectors` command and add the attribute `ExcludeManagementEventSources` with a value of `kms.amazonaws.com`. The following example creates an event selector for a trail named *`TrailName`* to include read-only and write-only management events, but exclude AWS KMS events. Because AWS KMS can generate a high volume of events, the user in this example might want to limit events to manage the cost of a trail.

```
aws cloudtrail put-event-selectors --trail-name TrailName --event-
selectors '[{"ReadWriteType": "All","ExcludeManagementEventSources":
 ["kms.amazonaws.com"],"IncludeManagementEvents": true}]'
```

The example returns the event selector configured for the trail.

```
{
    "TrailARN": "arn:aws:cloudtrail:us-east-1:111122223333:trail/TrailName",
    "EventSelectors": [
        {
            "ReadWriteType": "All",
            "IncludeManagementEvents": true,
            "DataResources": [],
            "ExcludeManagementEventSources": [
                "kms.amazonaws.com"
            ]
        }
    ]
}
```

To exclude Amazon RDS Data API management events from a trail's logs, run the `put-event-selectors` command and add the attribute `ExcludeManagementEventSources` with a value of `rdsdata.amazonaws.com`. The following example creates an event selector for a trail named *`TrailName`* to include read-only and write-only management events, but exclude Amazon RDS Data API management events. Because Amazon RDS Data API can generate a high volume of management events, the user in this example might want to limit events to manage the cost of a trail.

```
{
    "TrailARN": "arn:aws:cloudtrail:us-east-1:111122223333:trail/TrailName",
    "EventSelectors": [
        {
            "ReadWriteType": "All",
            "IncludeManagementEvents": true,
            "DataResources": [],
            "ExcludeManagementEventSources": [
                "rdsdata.amazonaws.com"
            ]
        }
    ]
}
```

To start logging AWS KMS or Amazon RDS Data API management events to a trail again, pass an empty string as the value of `ExcludeManagementEventSources`, as shown in the following command.

```
aws cloudtrail put-event-selectors --trail-name TrailName --event-
selectors '[{"ReadWriteType": "All","ExcludeManagementEventSources":
 [],"IncludeManagementEvents": true}]'
```

To log relevant AWS KMS events to a trail like `Disable`, `Delete` and `ScheduleKey`, but exclude high-volume AWS KMS events like `Encrypt`, `Decrypt`, and `GenerateDataKey`, log write-only management events, and keep the default setting to log AWS KMS events, as shown in the following example.

```
aws cloudtrail put-event-selectors --trail-name TrailName --event-
selectors '[{"ReadWriteType": "WriteOnly","ExcludeManagementEventSources":
 [],"IncludeManagementEvents": true}]'
```

Examples: Logging management events for event data stores

To view whether your event data store includes management events, run the **get-event-data-store** command.

```
aws cloudtrail get-event-data-store
```

```
--event-data-store arn:aws:cloudtrail:us-east-1:12345678910:eventdatastore/EXAMPLE-
f852-4e8f-8bd1-bcf6cEXAMPLE
```

The following is an example response. Creation and last updated times are in `timestamp` format.

```
{
    "EventDataStoreArn": "arn:aws:cloudtrail:us-east-1:12345678910:eventdatastore/
EXAMPLE-f852-4e8f-8bd1-bcf6cEXAMPLE",
    "Name": "myManagementEvents",
    "Status": "ENABLED",
    "AdvancedEventSelectors": [
        {
            "Name": "Management events selector",
            "FieldSelectors": [
                {
                    "Field": "eventCategory",
                    "Equals": [
                        "Management"
                    ]
                }
            ]
        }
    ],
    "MultiRegionEnabled": true,
    "OrganizationEnabled": false,
    "BillingMode": "FIXED_RETENTION_PRICING",
    "RetentionPeriod": 2557,
    "TerminationProtectionEnabled": true,
    "CreatedTimestamp": "2023-02-04T15:56:27.418000+00:00",
    "UpdatedTimestamp": "2023-02-04T15:56:27.544000+00:00"
}
```

To create an event data store that includes all management events, you run the **create-event-data-store** command. You do not need to specify any advanced event selectors to include all management events.

```
aws cloudtrail create-event-data-store
--name my-event-data-store
--retention-period 90\
```

The following is an example response.

```
{
    "EventDataStoreArn": "arn:aws:cloudtrail:us-east-1:12345678910:eventdatastore/
EXAMPLE-f852-4e8f-8bd1-bcf6cEXAMPLE",
    "Name": "my-event-data-store",
    "Status": "CREATED",
    "AdvancedEventSelectors": [
        {
            "Name": "Default management events",
            "FieldSelectors": [
                {
                    "Field": "eventCategory",
                    "Equals": [
                        "Management"
                    ]
                }
            ]
        }
    ],
    "MultiRegionEnabled": true,
    "OrganizationEnabled": false,
    "BillingMode": "EXTENDABLE_RETENTION_PRICING",
    "RetentionPeriod": 90,
    "TerminationProtectionEnabled": true,
    "CreatedTimestamp": "2023-11-13T16:41:57.224000+00:00",
    "UpdatedTimestamp": "2023-11-13T16:41:57.357000+00:00"
}
```

To create an event data store that excludes AWS Key Management Service (AWS KMS) events, run the `create-event-data-store` command and specify that `eventSource` does not equal `kms.amazonaws.com`. The following example creates an event data store that includes read-only and write-only management events, but excludes AWS KMS events.

```
aws cloudtrail create-event-data-store --name event-data-store-name --retention-period
 90 --advanced-event-selectors '[
    {
        "Name": "Management events selector",
        "FieldSelectors": [
            {"Field": "eventCategory","Equals": ["Management"]},
            {"Field": "eventSource","NotEquals": ["kms.amazonaws.com"]}
        ]
    }
]'
```

The following is an example response.

```
{
    "EventDataStoreArn": "arn:aws:cloudtrail:us-east-1:12345678910:eventdatastore/
EXAMPLE-f852-4e8f-8bd1-bcf6cEXAMPLE",
    "Name": "event-data-store-name",
    "Status": "CREATED",
    "AdvancedEventSelectors": [
        {
            "Name": "Management events selector",
            "FieldSelectors": [
                {
                    "Field": "eventCategory",
                    "Equals": [
                        "Management"
                    ]
                },
                {
                    "Field": "eventSource",
                    "NotEquals": [
                        "kms.amazonaws.com"
                    ]
                }
            ]
        }
    ],
    "MultiRegionEnabled": true,
    "OrganizationEnabled": false,
    "BillingMode": "EXTENDABLE_RETENTION_PRICING",
    "RetentionPeriod": 90,
    "TerminationProtectionEnabled": true,
    "CreatedTimestamp": "2023-11-13T17:02:02.067000+00:00",
    "UpdatedTimestamp": "2023-11-13T17:02:02.241000+00:00"
}
```

To create an event data store that excludes Amazon RDS Data API management events, run the `create-event-data-store` command and specify that `eventSource` does not equal `rdsdata.amazonaws.com`. The following example creates an event data store that includes read-only and write-only management events, but excludes Amazon RDS Data API events.

```
aws cloudtrail create-event-data-store --name event-data-store-name --retention-period
 90 --advanced-event-selectors '[
    {
        "Name": "Management events selector",
        "FieldSelectors": [
            {"Field": "eventCategory","Equals": ["Management"]},
            {"Field": "eventSource","NotEquals": ["rdsdata.amazonaws.com"]}
        ]
    }
]'
```

The following is an example response.

```
{
    "EventDataStoreArn": "arn:aws:cloudtrail:us-east-1:12345678910:eventdatastore/
EXAMPLE-f852-4e8f-8bd1-bcf6cEXAMPLE",
    "Name": "my-event-data-store",
    "Status": "CREATED",
    "AdvancedEventSelectors": [
        {
            "Name": "Management events selector",
            "FieldSelectors": [
                {
                    "Field": "eventCategory",
                    "Equals": [
                        "Management"
                    ]
                },
                {
                    "Field": "eventSource",
                    "NotEquals": [
                        "rdsdata.amazonaws.com"
                    ]
                }
            ]
        }
    ],
    "MultiRegionEnabled": true,
    "OrganizationEnabled": false,
    "BillingMode": "EXTENDABLE_RETENTION_PRICING",
    "RetentionPeriod": 90,
    "TerminationProtectionEnabled": true,
    "CreatedTimestamp": "2023-11-13T17:02:02.067000+00:00",
    "UpdatedTimestamp": "2023-11-13T17:02:02.241000+00:00"
}
```

Logging management events with the AWS SDKs

Use the GetEventSelectors operation to see whether your trail is logging management events for a trail. You can configure your trails to log management events with the PutEventSelectors operation. For more information, see the AWS CloudTrail API Reference.

Run the GetEventDataStore operation to see whether your event data store includes management events. You can configure your event data stores to include management events by running the CreateEventDataStore or UpdateEventDataStore operations. For more information, see Create, update, and manage event data stores with the AWS CLI and the AWS CloudTrail API Reference.

Logging data events

This section describes how to log data events using the CloudTrail console and AWS CLI.

By default, trails and event data stores do not log data events. Additional charges apply for data events. For more information, see AWS CloudTrail Pricing.

Data events provide information about the resource operations performed on or in a resource. These are also known as *data plane operations*. Data events are often high-volume activities.

Example data events include:

- Amazon S3 object-level API activity (for example, `GetObject`, `DeleteObject`, and `PutObject` API operations) on objects in S3 buckets.
- AWS Lambda function execution activity (the `Invoke` API).
- CloudTrail `PutAuditEvents` activity on a CloudTrail Lake channel that is used to log events from outside AWS.
- Amazon SNS `Publish` and `PublishBatch` API operations on topics.

You can use advanced event selectors to create fine-grained selectors, which help you control costs by only logging the specific events of interest for your use cases. For example, you can use advanced event selectors to log specific API calls by adding a filter on the `eventName` field. For more information, see Filtering data events by using advanced event selectors.

Note

The events that are logged by your trails are available in Amazon EventBridge. For example, if you choose to log data events for S3 objects but not management events, your trail processes and logs only data events for the specified S3 objects. The data events for these S3 objects are available in Amazon EventBridge. For more information, see Events from AWS services in the *Amazon EventBridge User Guide*.

Contents

  - Filtering data events by readOnly value using the AWS Management Console
  - Filtering data events by readOnly value using the AWS CLI
- Logging data events for AWS Config compliance
- Logging data events with the AWS SDKs

Data events

The following table shows the data event types available for trails and event data stores. The **Data event type (console)** column shows the appropriate selection in the console. The **resources.type value** column shows the `resources.type` value that you would specify to include data events of that type in your trail or event data store using the AWS CLI or CloudTrail APIs.

For trails, you can use basic or advanced event selectors to log data events for Amazon S3 objects in general purpose buckets, Lambda functions, and DynamoDB tables (shown in the first three rows of the table). You can use only advanced event selectors to log the data event types shown in the remaining rows.

For event data stores, you can use only advanced event selectors to include data events.

AWS service	Description	Data event type (console)	resources.type value
Amazon DynamoDB	Amazon DynamoDB item-level API activity on tables (for example, `PutItem`, `DeleteItem` , and `UpdateItem` API operations). **Note** For tables with streams enabled, the `resources`	**DynamoDB**	`AWS::DynamoDB::Table`

AWS service	Description	Data event type (console)	resources.type value
	field in the data event contains both `AWS::DynamoDB::Stream` and `AWS::DynamoDB::Table`. If you specify `AWS::DynamoDB::Table` for the `resources.type`, it will log both DynamoDB table and DynamoDB streams events by default. To exclude streams events, add a filter on the `eventName` field.		

AWS service	Description	Data event type (console)	resources.type value
AWS Lambda	AWS Lambda function execution activity (the `Invoke` API).	**Lambda**	`AWS::Lambda::Function`
Amazon S3	Amazon S3 object-level API activity (for example, `GetObject`, `DeleteObject`, and `PutObject` API operations) on objects in general purpose buckets.	**S3**	`AWS::S3::Object`
AWS AppConfig	AWS AppConfig API activity for configuration operations such as calls to `StartConfigurationSession` and `GetLatestConfiguration`.	**AWS AppConfig**	`AWS::AppConfig::Configuration`
AWS B2B Data Interchange	B2B Data Interchange API activity for Transformer operations such as calls to `GetTransformerJob` and `StartTransformerJob`.	**B2B Data Interchange**	`AWS::B2BI::Transformer`

AWS service	Description	Data event type (console)	resources.type value
Amazon Bedrock	Amazon Bedrock API activity on an agent alias.	**Bedrock agent alias**	`AWS::Bedrock::AgentAlias`
Amazon Bedrock	Amazon Bedrock API activity on a flow alias.	**Bedrock flow alias**	`AWS::Bedrock::FlowAlias`
Amazon Bedrock	Amazon Bedrock API activity on a guardrail .	**Bedrock guardrail**	`AWS::Bedrock::Guardrail`
Amazon Bedrock	Amazon Bedrock API activity on a knowledge base.	**Bedrock knowledge base**	`AWS::Bedrock::KnowledgeBase`
Amazon CloudFront	CloudFront API activity on a KeyValueStore.	**CloudFront KeyValueStore**	`AWS::CloudFront::KeyValueStore`
AWS Cloud Map	AWS Cloud Map API activity on a namespace.	**AWS Cloud Map namespace**	`AWS::ServiceDiscovery::Namespace`
AWS Cloud Map	AWS Cloud Map API activity on a service.	**AWS Cloud Map service**	`AWS::ServiceDiscovery::Service`
AWS CloudTrail	CloudTrail `PutAuditEvents` activity on a CloudTrail Lake channel that is used to log events from outside AWS.	**CloudTrail channel**	`AWS::CloudTrail::Channel`

AWS service	Description	Data event type (console)	resources.type value
Amazon CloudWatch	Amazon CloudWatch API activity on metrics.	**CloudWatch metric**	`AWS::CloudWatch::Metric`
Amazon CodeWhisperer	Amazon CodeWhisperer API activity on a customization.	**CodeWhisperer customization**	`AWS::CodeWhisperer::Customization`
Amazon CodeWhisperer	Amazon CodeWhisperer API activity on a profile.	**CodeWhisperer**	`AWS::CodeWhisperer::Profile`
Amazon Cognito	Amazon Cognito API activity on Amazon Cognito identity pools.	**Cognito Identity Pools**	`AWS::Cognito::IdentityPool`
Amazon DynamoDB	Amazon DynamoDB API activity on streams.	**DynamoDB Streams**	`AWS::DynamoDB::Stream`
Amazon Elastic Block Store	Amazon Elastic Block Store (EBS) direct APIs, such as `PutSnapshotBlock` , `GetSnapshotBlock` , and `ListChangedBlocks` on Amazon EBS snapshots.	**Amazon EBS direct APIs**	`AWS::EC2::Snapshot`

AWS service	Description	Data event type (console)	resources.type value
Amazon EMR	Amazon EMR API activity on a write-ahead log workspace.	**EMR write-ahead log workspace**	`AWS::EMRWAL::Workspace`
Amazon FinSpace	Amazon FinSpace API activity on environments.	**FinSpace**	`AWS::FinSpace::Environment`
AWS Glue	AWS Glue API activity on tables that were created by Lake Formation.	**Lake Formation**	`AWS::Glue::Table`
Amazon GuardDuty	Amazon GuardDuty API activity for a detector.	**GuardDuty detector**	`AWS::GuardDuty::Detector`
AWS HealthImaging	AWS HealthImaging API activity on data stores.	**MedicalImaging data store**	`AWS::MedicalImaging::Datastore`
AWS IoT	AWS IoT API activity on certificates.	**IoT certificate**	`AWS::IoT::Certificate`
AWS IoT	AWS IoT API activity on things.	**IoT thing**	`AWS::IoT::Thing`

AWS service	Description	Data event type (console)	resources.type value
AWS IoT Greengrass Version 2	Greengrass API activity from a Greengrass core device on a component version. **Note** Greengrass doesn't log access denied events.	**IoT Greengrass component version**	`AWS::GreengrassV2::ComponentVersion`
AWS IoT Greengrass Version 2	Greengrass API activity from a Greengrass core device on a deployment. **Note** Greengrass doesn't log access denied events.	**IoT Greengrass deployment**	`AWS::GreengrassV2::Deployment`
AWS IoT SiteWise	IoT SiteWise API activity on assets.	**IoT SiteWise asset**	`AWS::IoTSiteWise::Asset`
AWS IoT SiteWise	IoT SiteWise API activity on time series.	**IoT SiteWise time series**	`AWS::IoTSiteWise::TimeSeries`

AWS service	Description	Data event type (console)	resources.type value
AWS IoT TwinMaker	IoT TwinMaker API activity on an entity.	**IoT TwinMaker entity**	`AWS::IoTTwinMaker::Entity`
AWS IoT TwinMaker	IoT TwinMaker API activity on a workspace.	**IoT TwinMaker workspace**	`AWS::IoTTwinMaker::Workspace`
Amazon Kendra Intelligent Ranking	Amazon Kendra Intelligent Ranking API activity on rescore execution plans.	**Kendra Ranking**	`AWS::KendraRanking::ExecutionPlan`
Amazon Keyspaces (for Apache Cassandra)	Amazon Keyspaces API activity on a table.	**Cassandra table**	`AWS::Cassandra::Table`
Amazon Kinesis Data Streams	Kinesis Data Streams API activity on streams.	**Kinesis stream**	`AWS::Kinesis::Stream`
Amazon Kinesis Data Streams	Kinesis Data Streams API activity on stream consumers.	**Kinesis stream consumer**	`AWS::Kinesis::StreamConsumer`
Amazon Kinesis Video Streams	Kinesis Video Streams API activity on video streams, such as calls to `GetMedia` and `PutMedia`.	**Kinesis video stream**	`AWS::KinesisVideo::Stream`

AWS service	Description	Data event type (console)	resources.type value
Amazon Machine Learning	Machine Learning API activity on ML models.	**Maching Learning MlModel**	`AWS::MachineLearning::MlModel`
Amazon Managed Blockchain	Amazon Managed Blockchain API activity on a network.	**Managed Blockchain network**	`AWS::ManagedBlockchain::Network`
Amazon Managed Blockchain	Amazon Managed Blockchain JSON-RPC calls on Ethereum nodes, such as `eth_getBalance` or `eth_getBlockByNumber` .	**Managed Blockchain**	`AWS::ManagedBlockchain::Node`
Amazon Neptune Graph	Data API activities, for example queries, algorithms, or vector search, on a Neptune Graph.	**Neptune Graph**	`AWS::NeptuneGraph::Graph`
AWS Payment Cryptography	AWS Payment Cryptography API activity on aliases.	**Payment Cryptography Alias**	`AWS::PaymentCryptography::Alias`
AWS Payment Cryptography	AWS Payment Cryptography API activity on keys.	**Payment Cryptography Key**	`AWS::PaymentCryptography::Key`

AWS service	Description	Data event type (console)	resources.type value
AWS Private CA	AWS Private CA Connector for Active Directory API activity.	**AWS Private CA Connector for Active Directory**	`AWS::PCAConnectorAD::Connector`
AWS Private CA	AWS Private CA Connector for SCEP API activity.	**AWS Private CA Connector for SCEP**	`AWS::PCAConnectorSCEP::Connector`
Amazon Q Apps	Data API activity on Amazon Q Apps.	**Amazon Q Apps**	`AWS::QApps:QApp`
Amazon Q Business	Amazon Q Business API activity on an application.	**Amazon Q Business application**	`AWS::QBusiness::Application`
Amazon Q Business	Amazon Q Business API activity on a data source.	**Amazon Q Business data source**	`AWS::QBusiness::DataSource`
Amazon Q Business	Amazon Q Business API activity on an index.	**Amazon Q Business index**	`AWS::QBusiness::Index`
Amazon Q Business	Amazon Q Business API activity on a web experience.	**Amazon Q Business web experience**	`AWS::QBusiness::WebExperience`
Amazon RDS	Amazon RDS API activity on a DB Cluster.	**RDS Data API - DB Cluster**	`AWS::RDS::DBCluster`

AWS service	Description	Data event type (console)	resources.type value
Amazon S3	Amazon S3 API activity on access points.	**S3 Access Point**	`AWS::S3::AccessPoint`
Amazon S3	Amazon S3 object-level API activity (for example, `GetObject` , `DeleteObject` , and `PutObject` API operations) on objects in directory buckets.	**S3 Express**	`AWS::S3Express::Object`
Amazon S3	Amazon S3 Object Lambda access points API activity, such as calls to `CompleteMultipartUpload` and `GetObject` .	**S3 Object Lambda**	`AWS::S3ObjectLambda::AccessPoint`
Amazon S3 on Outposts	Amazon S3 on Outposts object-level API activity.	**S3 Outposts**	`AWS::S3Outposts::Object`
Amazon SageMaker	Amazon SageMaker `InvokeEndpointWithResponseStream` activity on endpoints.	**SageMaker endpoint**	`AWS::SageMaker::Endpoint`
Amazon SageMaker	Amazon SageMaker API activity on feature stores.	**SageMaker feature store**	`AWS::SageMaker::FeatureGroup`

AWS service	Description	Data event type (console)	resources.type value
Amazon SageMaker	Amazon SageMaker API activity on experiment trial components.	**SageMaker metrics experiment trial component**	`AWS::SageMaker::ExperimentTrialComponent`
Amazon SNS	Amazon SNS `Publish` API operations on platform endpoints.	**SNS platform endpoint**	`AWS::SNS::PlatformEndpoint`
Amazon SNS	Amazon SNS `Publish` and `PublishBatch` API operations on topics.	**SNS topic**	`AWS::SNS::Topic`
Amazon SQS	Amazon SQS API activity on messages.	**SQS**	`AWS::SQS::Queue`
AWS Step Functions	Step Functions API activity on a state machine.	**Step Functions state machine**	`AWS::StepFunctions::StateMachine`
AWS Supply Chain	AWS Supply Chain API activity on an instance.	**Supply Chain**	`AWS::SCN::Instance`
Amazon SWF	Amazon SWF API activity on domains.	**SWF domain**	`AWS::SWF::Domain`
AWS Systems Manager	Systems Manager API activity on control channels.	**Systems Manager**	`AWS::SSMMessages::ControlChannel`

AWS service	Description	Data event type (console)	resources.type value
AWS Systems Manager	Systems Manager API activity on managed nodes.	**Systems Manager managed node**	`AWS::SSM::ManagedNode`
Amazon Timestream	Amazon Timestream `Query` API activity on databases.	**Timestream database**	`AWS::Timestream::Database`
Amazon Timestream	Amazon Timestream `Query` API activity on tables.	**Timestream table**	`AWS::Timestream::Table`
Amazon Verified Permissions	Amazon Verified Permissions API activity on a policy store.	**Amazon Verified Permissions**	`AWS::VerifiedPermissions::PolicyStore`
Amazon WorkSpaces Thin Client	WorkSpaces Thin Client API activity on a Device.	**Thin Client Device**	`AWS::ThinClient::Device`
Amazon WorkSpaces Thin Client	WorkSpaces Thin Client API activity on an Environment.	**Thin Client Environment**	`AWS::ThinClient::Environment`
AWS X-Ray	X-Ray API activity on traces.	**X-Ray trace**	`AWS::XRay::Trace`

To record CloudTrail data events, you must explicitly add each resource type for which you want to collect activity. For more information, see Creating a trail with the CloudTrail console and Create an event data store for CloudTrail events with the console.

On a single-Region trail or event data store, you can log data events only for resources that you can access in that Region. Though S3 buckets are global, AWS Lambda functions and DynamoDB tables are regional.

Additional charges apply for logging data events. For CloudTrail pricing, see AWS CloudTrail Pricing.

Examples: Logging data events for Amazon S3 objects

Logging data events for all S3 objects in an S3 bucket

The following example demonstrates how logging works when you configure logging of all data events for an S3 bucket named `DOC-EXAMPLE-BUCKET`. In this example, the CloudTrail user specified an empty prefix, and the option to log both **Read** and **Write** data events.

1. A user uploads an object to `DOC-EXAMPLE-BUCKET`.
2. The `PutObject` API operation is an Amazon S3 object-level API. It is recorded as a data event in CloudTrail. Because the CloudTrail user specified an S3 bucket with an empty prefix, events that occur on any object in that bucket are logged. The trail or event data store processes and logs the event.
3. Another user uploads an object to `DOC-EXAMPLE-BUCKET2`.
4. The `PutObject` API operation occurred on an object in an S3 bucket that wasn't specified for the trail or event data store. The trail or event data store doesn't log the event.

Logging data events for specific S3 objects

The following example demonstrates how logging works when you configure a trail or event data store to log events for specific S3 objects. In this example, the CloudTrail user specified an S3 bucket named `DOC-EXAMPLE-BUCKET3`, with the prefix *`my-images`*, and the option to log only **Write** data events.

1. A user deletes an object that begins with the `my-images` prefix in the bucket, such as `arn:aws:s3:::DOC-EXAMPLE-BUCKET3/my-images/example.jpg`.
2. The `DeleteObject` API operation is an Amazon S3 object-level API. It is recorded as a **Write** data event in CloudTrail. The event occurred on an object that matches the S3 bucket and prefix specified in the trail or event data store. The trail or event data store processes and logs the event.

3. Another user deletes an object with a different prefix in the S3 bucket, such as `arn:aws:s3:::DOC-EXAMPLE-BUCKET3/my-videos/example.avi`.
4. The event occurred on an object that doesn't match the prefix specified in your trail or event data store. The trail or event data store doesn't log the event.
5. A user calls the `GetObject` API operation for the object, `arn:aws:s3:::DOC-EXAMPLE-BUCKET3/my-images/example.jpg`.
6. The event occurred on a bucket and prefix that are specified in the trail or event data store, but `GetObject` is a read-type Amazon S3 object-level API. It is recorded as a **Read** data event in CloudTrail, and the trail or event data store is not configured to log **Read** events. The trail or event data store doesn't log the event.

> **Note**
>
> For trails, if you are logging data events for specific Amazon S3 buckets, we recommend you do not use an Amazon S3 bucket for which you are logging data events to receive log files that you have specified in the data events section for your trail. Using the same Amazon S3 bucket causes your trail to log a data event each time log files are delivered to your Amazon S3 bucket. Log files are aggregated events delivered at intervals, so this is not a 1:1 ratio of event to log file; the event is logged in the next log file. For example, when CloudTrail delivers logs, the `PutObject` event occurs on the S3 bucket. If the S3 bucket is also specified in the data events section, the trail processes and logs the `PutObject` event as a data event. That action is another `PutObject` event, and the trail processes and logs the event again.
>
> To avoid logging data events for the Amazon S3 bucket where you receive log files if you configure a trail to log all Amazon S3 data events in your AWS account, consider configuring delivery of log files to an Amazon S3 bucket that belongs to another AWS account. For more information, see Receiving CloudTrail log files from multiple accounts.

Logging data events for S3 objects in other AWS accounts

When you configure your trail to log data events, you can also specify S3 objects that belong to other AWS accounts. When an event occurs on a specified object, CloudTrail evaluates whether the event matches any trails in each account. If the event matches the settings for a trail, the trail processes and logs the event for that account. Generally, both API callers and resource owners can receive events.

If you own an S3 object and you specify it in your trail, your trail logs events that occur on the object in your account. Because you own the object, your trail also logs events when other accounts call the object.

If you specify an S3 object in your trail, and another account owns the object, your trail only logs events that occur on that object in your account. Your trail doesn't log events that occur in other accounts.

Example: Logging data events for an Amazon S3 object for two AWS accounts

The following example shows how two AWS accounts configure CloudTrail to log events for the same S3 object.

1. In your account, you want your trail to log data events for all objects in your S3 bucket named `DOC-EXAMPLE-BUCKET`. You configure the trail by specifying the S3 bucket with an empty object prefix.
2. Bob has a separate account that has been granted access to the S3 bucket. Bob also wants to log data events for all objects in the same S3 bucket. For his trail, he configures his trail and specifies the same S3 bucket with an empty object prefix.
3. Bob uploads an object to the S3 bucket with the `PutObject` API operation.
4. This event occurred in his account and it matches the settings for his trail. Bob's trail processes and logs the event.
5. Because you own the S3 bucket and the event matches the settings for your trail, your trail also processes and logs the same event. Because there are now two copies of the event (one logged in Bob's trail, and one logged in yours), CloudTrail charges for two copies of the data event.
6. You upload an object to the S3 bucket.
7. This event occurs in your account and it matches the settings for your trail. Your trail processes and logs the event.
8. Because the event didn't occur in Bob's account, and he doesn't own the S3 bucket, Bob's trail doesn't log the event. CloudTrail charges for only one copy of this data event.

Example: Logging data events for all buckets, including an S3 bucket used by two AWS accounts

The following example shows the logging behavior when **Select all S3 buckets in your account** is enabled for trails that collect data events in an AWS account.

1. In your account, you want your trail to log data events for all S3 buckets. You configure the trail by choosing **Read** events, **Write** events, or both for **All current and future S3 buckets** in **Data events**.
2. Bob has a separate account that has been granted access to an S3 bucket in your account. He wants to log data events for the bucket to which he has access. He configures his trail to get data events for all S3 buckets.
3. Bob uploads an object to the S3 bucket with the `PutObject` API operation.
4. This event occurred in his account and it matches the settings for his trail. Bob's trail processes and logs the event.
5. Because you own the S3 bucket and the event matches the settings for your trail, your trail also processes and logs the event. Because there are now two copies of the event (one logged in Bob's trail, and one logged in yours), CloudTrail charges each account for a copy of the data event.
6. You upload an object to the S3 bucket.
7. This event occurs in your account and it matches the settings for your trail. Your trail processes and logs the event.
8. Because the event didn't occur in Bob's account, and he doesn't own the S3 bucket, Bob's trail doesn't log the event. CloudTrail charges for only one copy of this data event in your account.
9. A third user, Mary, has access to the S3 bucket, and runs a `GetObject` operation on the bucket. She has a trail configured to log data events on all S3 buckets in her account. Because she is the API caller, CloudTrail logs a data event in her trail. Though Bob has access to the bucket, he is not the resource owner, so no event is logged in his trail this time. As the resource owner, you receive an event in your trail about the `GetObject` operation that Mary called. CloudTrail charges your account and Mary's account for each copy of the data event: one in Mary's trail, and one in yours.

Read-only and write-only events

When you configure your trail or event data store to log data and management events, you can specify whether you want read-only events, write-only events, or both.

- **Read**

  Read events include API operations that read your resources, but don't make changes. For example, read-only events include the Amazon EC2 `DescribeSecurityGroups` and

`DescribeSubnets` API operations. These operations return only information about your Amazon EC2 resources and don't change your configurations.

- **Write**

  Write events include API operations that modify (or might modify) your resources. For example, the Amazon EC2 `RunInstances` and `TerminateInstances` API operations modify your instances.

Example: Logging read and write events for separate trails

The following example shows how you can configure trails to split log activity for an account into separate S3 buckets: one bucket named DOC-EXAMPLE-BUCKET1 receives read-only events and a second DOC-EXAMPLE-BUCKET2 receives write-only events.

1. You create a trail and choose the S3 bucket named `DOC-EXAMPLE-BUCKET1` to receive log files. You then update the trail to specify that you want **Read** management events and data events.
2. You create a second trail and choose the S3 bucket the `DOC-EXAMPLE-BUCKET2` to receive log files. You then update the trail to specify that you want **Write** management events and data events.
3. The Amazon EC2 `DescribeInstances` and `TerminateInstances` API operations occur in your account.
4. The `DescribeInstances` API operation is a read-only event and it matches the settings for the first trail. The trail logs and delivers the event to the `DOC-EXAMPLE-BUCKET1`.
5. The `TerminateInstances` API operation is a write-only event and it matches the settings for the second trail. The trail logs and delivers the event to the `DOC-EXAMPLE-BUCKET2` .

Logging data events with the AWS Management Console

The following procedures describe how to an update existing event data store or trail to log data events by using the AWS Management Console. For information about how to create an event data store to log data events, see Create an event data store for CloudTrail events with the console. For information about how to create a trail to log data events, see Creating a trail in the console.

For trails, the steps for logging data events differ based on whether you're using advanced event selectors or basic event selectors. You can log data events for all data event types using advanced event selectors, but if you use basic event selectors you're limited to logging data events for Amazon S3 buckets and bucket objects, AWS Lambda functions, and Amazon DynamoDB tables.

Updating an existing event data store to log data events in the AWS Management Console

Use the following procedure to update an existing event data store to log data events. For more information about using advanced event selectors, see Filtering data events by using advanced event selectors in this topic.

1. Sign in to the AWS Management Console and open the CloudTrail console at https://console.aws.amazon.com/cloudtrail/.
2. From the navigation pane, under **Lake**, choose **Event data stores**.
3. On the **Event data stores** page, choose the event data store you want to update.

   > **Note**
   >
   > You can only enable data events on event data stores that contain CloudTrail events. You cannot enable data events on CloudTrail event data stores for AWS Config configuration items, CloudTrail Insights events, or non-AWS events.

4. On the details page, in **Data events**, choose **Edit**.
5. If you are not already logging data events, choose the **Data events** check box.
6. For **Data event type**, choose the resource type on which you want to log data events.
7. Choose a log selector template. CloudTrail includes predefined templates that log all data events for the resource type. To build a custom log selector template, choose **Custom**.
8. (Optional) In **Selector name**, enter a name to identify your selector. The selector name is a descriptive name for an advanced event selector, such as "Log data events for only two S3 buckets". The selector name is listed as `Name` in the advanced event selector and is viewable if you expand the **JSON view**.
9. In **Advanced event selectors**, build an expression for the specific resources on which you want to log data events. You can skip this step if you are using a predefined log template.

   a. Choose from the following fields.

      - **`readOnly`** - `readOnly` can be set to **equals** a value of `true` or `false`. Read-only data events are events that do not change the state of a resource, such as `Get*` or `Describe*` events. Write events add, change, or delete resources, attributes, or artifacts, such as `Put*`, `Delete*`, or `Write*` events. To log both `read` and `write` events, don't add a `readOnly` selector.

- **`eventName`** - `eventName` can use any operator. You can use it to include or exclude any data event logged to CloudTrail, such as `PutBucket`, `GetItem`, or `GetSnapshotBlock`.
- **`resources.ARN`** - You can use any operator with `resources.ARN`, but if you use **equals** or **does not equal**, the value must exactly match the ARN of a valid resource of the type you've specified in the template as the value of `resources.type`.

  The following table shows the valid ARN format for each `resources.type`.

  > **Note**
  >
  > You can't use the `resources.ARN` field to filter resource types that do not have ARNs.

resources.type	resources.ARN
`AWS::DynamoDB::Table` [1]	`arn:partition:dynamodb:region:account_ID:table/table_name`
`AWS::Lambda::Function`	`arn:partition:lambda:region:account_ID:function:function_name`
`AWS::S3::Object` [2]	`arn:partition:s3:::DOC-EXAMPLE-BUCKET/` `arn:partition:s3:::DOC-EXAMPLE-BUCKET/object_or_file_name/`
`AWS::AppConfig::Configuration`	`arn:partition:appconfig:region:account_ID:application/application_ID/environment/environment_ID/configuration/configuration_profile_ID`

resources.type	resources.ARN
`AWS::B2BI::Transformer`	`arn:partition :b2bi:region:account_ID :transformer/ transformer_ID`
`AWS::Bedrock::AgentAlias`	`arn:partition :bedrock: region:account_ID :agent-alias/ agent_ID/alias_ID`
`AWS::Bedrock::FlowAlias`	`arn:partition :bedrock: region:account_ID :flow/flow_ID/alias/alias_ID`
`AWS::Bedrock::Guardrail`	`arn:partition :bedrock: region:account_ID :guardrail/ guardrail_ID`
`AWS::Bedrock::KnowledgeBase`	`arn:partition :bedrock: region:account_ID :knowledge-base/knowledge_base_ID`
`AWS::Cassandra::Table`	`arn:partition :cassandra: region:account_ID :keyspace/ keyspace_name /table/table_name`
`AWS::CloudFront::KeyValueStore`	`arn:partition :cloudfront: region:account_ID :key-value-store/KVS_name`
`AWS::CloudTrail::Channel`	`arn:partition :cloudtrail: region:account_ID :channel/ channel_UUID`

resources.type	resources.ARN
AWS::CodeWhisperer::Customization	arn:*partition* :codewhisperer: *region*:*account_ID* :customization/ *customization_ID*
AWS::CodeWhisperer::Profile	arn:*partition* :codewhisperer: *region*:*account_ID* :profile/ *profile_ID*
AWS::Cognito::IdentityPool	arn:*partition* :cognito-identity: *region*:*account_ID* :identitypool/ *identity_pool_ID*
AWS::DynamoDB::Stream	arn:*partition* :dynamodb : *region*:*account_ID* :table/*table_name* /stream/*date_time*
AWS::EC2::Snapshot	arn:*partition* :ec2:*region*::snapshot/ *snapshot_ID*
AWS::EMRWAL::Workspace	arn:*partition* :emrwal:*region*:*account_ID* :workspace/ *workspace_name*
AWS::FinSpace::Environment	arn:*partition* :finspace : *region*:*account_ID* :environment/ *environment_ID*
AWS::Glue::Table	arn:*partition* :glue:*region*:*account_ID* :table/*database_name* /*table_name*
AWS::GreengrassV2::ComponentVersion	arn:*partition* :greengrass: *region*:*account_ID* :components/ *component_name*

resources.type	resources.ARN
`AWS::GreengrassV2::Deployment`	`arn:partition :greengrass: region:account_ID :deployments/ deployment_ID`
`AWS::GuardDuty::Detector`	`arn:partition :guardduty: region:account_ID :detector/ detector_ID`
`AWS::IoT::Certificate`	`arn:partition :iot:region:account_ID :cert/certificate_ID`
`AWS::IoT::Thing`	`arn:partition :iot:region:account_ID :thing/thing_ID`
`AWS::IoTSiteWise::Asset`	`arn:partition :iotsitewise: region:account_ID :asset/asset_ID`
`AWS::IoTSiteWise::TimeSeries`	`arn:partition :iotsitewise: region:account_ID :timeseries/ timeseries_ID`
`AWS::IoTTwinMaker::Entity`	`arn:partition :iottwinmaker: region:account_ID :workspace/ workspace_ID /entity/entity_ID`
`AWS::IoTTwinMaker::Workspace`	`arn:partition :iottwinmaker: region:account_ID :workspace/ workspace_ID`

resources.type	resources.ARN
`AWS::KendraRanking::ExecutionPlan`	`arn:partition:kendra-ranking:region:account_ID:rescore-execution-plan/rescore_execution_plan_ID`
`AWS::Kinesis::Stream`	`arn:partition:kinesis:region:account_ID:stream/stream_name`
`AWS::Kinesis::StreamConsumer`	`arn:partition:kinesis:region:account_ID:stream_type/stream_name/consumer/consumer_name:consumer_creation_timestamp`
`AWS::KinesisVideo::Stream`	`arn:partition:kinesisvideo:region:account_ID:stream/stream_name/creation_time`
`AWS::MachineLearning::MlModel`	`arn:partition:machinelearning:region:account_ID:mlmodel/model_ID`
`AWS::ManagedBlockchain::Network`	`arn:partition:managedblockchain:::networks/network_name`
`AWS::ManagedBlockchain::Node`	`arn:partition:managedblockchain:region:account_ID:nodes/node_ID`
`AWS::MedicalImaging::Datastore`	`arn:partition:medical-imaging:region:account_ID:datastore/data_store_ID`

resources.type	resources.ARN
`AWS::NeptuneGraph::Graph`	`arn:partition:neptune-graph:region:account_ID:graph/graph_ID`
`AWS::PaymentCryptography::Alias`	`arn:partition:payment-cryptography:region:account_ID:alias/alias`
`AWS::PaymentCryptography::Key`	`arn:partition:payment-cryptography:region:account_ID:key/key_ID`
`AWS::PCAConnectorAD::Connector`	`arn:partition:pca-connector-ad:region:account_ID:connector/connector_ID`
`AWS::PCAConnectorSCEP::Connector`	`arn:partition:pca-connector-scep:region:account_ID:connector/connector_ID`
`AWS::QApps:QApp`	`arn:partition:qapps:region:account_ID:application/application_UUID/qapp/qapp_UUID`
`AWS::QBusiness::Application`	`arn:partition:qbusiness:region:account_ID:application/application_ID`
`AWS::QBusiness::DataSource`	`arn:partition:qbusiness:region:account_ID:application/application_ID/index/index_ID/data-source/datasource_ID`

resources.type	resources.ARN
`AWS::QBusiness::Index`	`arn:partition:qbusiness:region:account_ID:application/application_ID/index/index_ID`
`AWS::QBusiness::WebExperience`	`arn:partition:qbusiness:region:account_ID:application/application_ID/web-experience/web_experienc_ID`
`AWS::RDS::DBCluster`	`arn:partition:rds:region:account_ID:cluster/cluster_name`
`AWS::S3::AccessPoint` [3]	`arn:partition:s3:region:account_ID:accesspoint/access_point_name`
`AWS::S3Express::Object`	`arn:partition:s3express:region:account_ID:bucket/bucket_name`
`AWS::S3ObjectLambda::AccessPoint`	`arn:partition:s3-object-lambda:region:account_ID:accesspoint/access_point_name`
`AWS::S3Outposts::Object`	`arn:partition:s3-outposts:region:account_ID:object_path`
`AWS::SageMaker::Endpoint`	`arn:partition:sagemaker:region:account_ID:endpoint/endpoint_name`

resources.type	resources.ARN
`AWS::SageMaker::ExperimentTrialComponent`	`arn:partition:sagemaker:region:account_ID:experiment-trial-component/experiment_trial_component_name`
`AWS::SageMaker::FeatureGroup`	`arn:partition:sagemaker:region:account_ID:feature-group/feature_group_name`
`AWS::SCN::Instance`	`arn:partition:scn:region:account_ID:instance/instance_ID`
`AWS::ServiceDiscovery::Namespace`	`arn:partition:servicediscovery:region:account_ID:namespace/namespace_ID`
`AWS::ServiceDiscovery::Service`	`arn:partition:servicediscovery:region:account_ID:service/service_ID`
`AWS::SNS::PlatformEndpoint`	`arn:partition:sns:region:account_ID:endpoint/endpoint_type/endpoint_name/endpoint_ID`
`AWS::SNS::Topic`	`arn:partition:sns:region:account_ID:topic_name`
`AWS::SQS::Queue`	`arn:partition:sqs:region:account_ID:queue_name`

resources.type	resources.ARN
`AWS::SSM::ManagedNode`	The ARN must be in one of the following formats: • `arn:partition:ssm:region:account_ID:managed-instance/instance_ID` • `arn:partition:ec2:region:account_ID:instance/instance_ID`
`AWS::SSMMessages::ControlChannel`	`arn:partition:ssmmessages:region:account_ID:control-channel/control_channel_ID`
`AWS::StepFunctions::StateMachine`	The ARN must be in one of the following formats: • `arn:partition:states:region:account_ID:stateMachine:stateMachine_name` • `arn:partition:states:region:account_ID:stateMachine:stateMachine_name/label_name`
`AWS::SWF::Domain`	`arn:partition:swf:region:account_ID:/domain/domain_name`
`AWS::ThinClient::Device`	`arn:partition:thinclient:region:account_ID:device/device_ID`

resources.type	resources.ARN
`AWS::ThinClient::Environment`	`arn:partition :thinclient: region:account_ID :environment/ environment_ID`
`AWS::Timestream::Database`	`arn:partition :timestream: region:account_ID :database/ database_name`
`AWS::Timestream::Table`	`arn:partition :timestream: region:account_ID :database/ database_name /table/table_name`
`AWS::VerifiedPermissions::PolicyStore`	`arn:partition :verifiedpermissions: region:account_ID :policy-store/ policy_store_ID`

[1] For tables with streams enabled, the `resources` field in the data event contains both `AWS::DynamoDB::Stream` and `AWS::DynamoDB::Table`. If you specify `AWS::DynamoDB::Table` for the `resources.type`, it will log both DynamoDB table and DynamoDB streams events by default. To exclude streams events, add a filter on the `eventName` field.

[2] To log all data events for all objects in a specific S3 bucket, use the `StartsWith` operator, and include only the bucket ARN as the matching value. The trailing slash is intentional; do not exclude it.

[3] To log events on all objects in an S3 access point, we recommend that you use only the access point ARN, don't include the object path, and use the `StartsWith` or `NotStartsWith` operators.

For more information about the ARN formats of data event resources, see Actions, resources, and condition keys in the *AWS Identity and Access Management User Guide*.

b. For each field, choose **+ Condition** to add as many conditions as you need, up to a maximum of 500 specified values for all conditions. For example, to exclude data events for two S3 buckets from data events that are logged on your event data store, you can set the field to **resources.ARN**, set the operator for **does not start with**, and then either paste in an S3 bucket ARN, or browse for the S3 buckets for which you do not want to log events.

   To add the second S3 bucket, choose **+ Condition**, and then repeat the preceding instruction, pasting in the ARN for or browsing for a different bucket.

   > **Note**
   >
   > You can have a maximum of 500 values for all selectors on an event data store. This includes arrays of multiple values for a selector such as `eventName`. If you have single values for all selectors, you can have a maximum of 500 conditions added to a selector.

c. Choose **+ Field** to add additional fields as required. To avoid errors, do not set conflicting or duplicate values for fields. For example, do not specify an ARN in one selector to be equal to a value, then specify that the ARN not equal the same value in another selector.

10. To add another data type on which to log data events, choose **Add data event type**. Repeat steps 6 through this step to configure advanced event selectors for the data event type.
11. After you've reviewed and verified your choices, choose **Save changes**.

Updating an existing trail to log data events with advanced event selectors in the AWS Management Console

In the AWS Management Console, if your trail is using advanced event selectors, you can choose from predefined templates that log all data events on a selected resource. After you choose a log selector template, you can customize the template to include only the data events you most want to see. For more information about using advanced event selectors, see Filtering data events by using advanced event selectors in this topic.

1. On the **Dashboard** or **Trails** pages of the CloudTrail console, choose the trail you want to update.
2. On the details page, in **Data events**, choose **Edit**.
3. If you are not already logging data events, choose the **Data events** check box.

4. For **Data event type**, choose the resource type on which you want to log data events.

5. Choose a log selector template. CloudTrail includes predefined templates that log all data events for the resource type. To build a custom log selector template, choose **Custom**.

> **Note**
>
> Choosing a predefined template for S3 buckets enables data event logging for all buckets currently in your AWS account and any buckets you create after you finish creating the trail. It also enables logging of data event activity performed by any user or role in your AWS account, even if that activity is performed on a bucket that belongs to another AWS account.
>
> If the trail applies only to one Region, choosing a predefined template that logs all S3 buckets enables data event logging for all buckets in the same Region as your trail and any buckets you create later in that Region. It will not log data events for Amazon S3 buckets in other Regions in your AWS account.
>
> If you are creating a trail for all Regions, choosing a predefined template for Lambda functions enables data event logging for all functions currently in your AWS account, and any Lambda functions you might create in any Region after you finish creating the trail. If you are creating a trail for a single Region (for trails, this only can be done by using the AWS CLI), this selection enables data event logging for all functions currently in that Region in your AWS account, and any Lambda functions you might create in that Region after you finish creating the trail. It does not enable data event logging for Lambda functions created in other Regions.
>
> Logging data events for all functions also enables logging of data event activity performed by any user or role in your AWS account, even if that activity is performed on a function that belongs to another AWS account.

6. (Optional) In **Selector name**, enter a name to identify your selector. The selector name is a descriptive name for an advanced event selector, such as "Log data events for only two S3 buckets". The selector name is listed as `Name` in the advanced event selector and is viewable if you expand the **JSON view**.

7. In **Advanced event selectors**, build an expression for the specific resources on which you want to log data events. You can skip this step if you are using a predefined log template.

   a. Choose from the following fields.

- **`readOnly`** - `readOnly` can be set to **equals** a value of `true` or `false`. Read-only data events are events that do not change the state of a resource, such as `Get*` or `Describe*` events. Write events add, change, or delete resources, attributes, or artifacts, such as `Put*`, `Delete*`, or `Write*` events. To log both `read` and `write` events, don't add a `readOnly` selector.
- **`eventName`** - `eventName` can use any operator. You can use it to include or exclude any data event logged to CloudTrail, such as `PutBucket`, `GetItem`, or `GetSnapshotBlock`.
- **`resources.ARN`** - You can use any operator with `resources.ARN`, but if you use **equals** or **does not equal**, the value must exactly match the ARN of a valid resource of the type you've specified in the template as the value of `resources.type`.

  The following table shows the valid ARN format for each `resources.type`.

  > **Note**
  >
  > You can't use the `resources.ARN` field to filter resource types that do not have ARNs.

resources.type	resources.ARN
`AWS::DynamoDB::Table` [1]	`arn:*partition* :dynamodb : *region*:*account_ID* :table/*table_name*`
`AWS::Lambda::Function`	`arn:*partition* :lambda:*region*:*account_ID* :function: *function_name*`

`AWS::S3::Object` [2]	`arn:*partition* :s3:::*DOC-EXAMPLE-BUCKET* /`
	`arn:*partition* :s3:::*DOC-EXAMPLE-BUCKET* /*object_or_file_name* /`

resources.type	resources.ARN
`AWS::AppConfig::Configuration`	`arn:partition:appconfig:region:account_ID:application/application_ID/environment/environment_ID/configuration/configuration_profile_ID`
`AWS::B2BI::Transformer`	`arn:partition:b2bi:region:account_ID:transformer/transformer_ID`
`AWS::Bedrock::AgentAlias`	`arn:partition:bedrock:region:account_ID:agent-alias/agent_ID/alias_ID`
`AWS::Bedrock::FlowAlias`	`arn:partition:bedrock:region:account_ID:flow/flow_ID/alias/alias_ID`
`AWS::Bedrock::Guardrail`	`arn:partition:bedrock:region:account_ID:guardrail/guardrail_ID`
`AWS::Bedrock::KnowledgeBase`	`arn:partition:bedrock:region:account_ID:knowledge-base/knowledge_base_ID`
`AWS::Cassandra::Table`	`arn:partition:cassandra:region:account_ID:keyspace/keyspace_name/table/table_name`
`AWS::CloudFront::KeyValueStore`	`arn:partition:cloudfront:region:account_ID:key-value-store/KVS_name`

resources.type	resources.ARN
`AWS::CloudTrail::Channel`	`arn:partition:cloudtrail:region:account_ID:channel/channel_UUID`
`AWS::CodeWhisperer::Customization`	`arn:partition:codewhisperer:region:account_ID:customization/customization_ID`
`AWS::CodeWhisperer::Profile`	`arn:partition:codewhisperer:region:account_ID:profile/profile_ID`
`AWS::Cognito::IdentityPool`	`arn:partition:cognito-identity:region:account_ID:identitypool/identity_pool_ID`
`AWS::DynamoDB::Stream`	`arn:partition:dynamodb:region:account_ID:table/table_name/stream/date_time`
`AWS::EC2::Snapshot`	`arn:partition:ec2:region::snapshot/snapshot_ID`
`AWS::EMRWAL::Workspace`	`arn:partition:emrwal:region:account_ID:workspace/workspace_name`
`AWS::FinSpace::Environment`	`arn:partition:finspace:region:account_ID:environment/environment_ID`
`AWS::Glue::Table`	`arn:partition:glue:region:account_ID:table/database_name/table_name`

resources.type	resources.ARN
AWS::GreengrassV2::ComponentVersion	`arn:partition :greengrass: region:account_ID :components/ component_name`
AWS::GreengrassV2::Deployment	`arn:partition :greengrass: region:account_ID :deployments/ deployment_ID`
AWS::GuardDuty::Detector	`arn:partition :guardduty: region:account_ID :detector/ detector_ID`
AWS::IoT::Certificate	`arn:partition :iot:region:account_ID :cert/certificate_ID`
AWS::IoT::Thing	`arn:partition :iot:region:account_ID :thing/thing_ID`
AWS::IoTSiteWise::Asset	`arn:partition :iotsitewise: region:account_ID :asset/asset_ID`
AWS::IoTSiteWise::TimeSeries	`arn:partition :iotsitewise: region:account_ID :timeseries/ timeseries_ID`
AWS::IoTTwinMaker::Entity	`arn:partition :iottwinmaker: region:account_ID :workspace/ workspace_ID /entity/entity_ID`
AWS::IoTTwinMaker::Workspace	`arn:partition :iottwinmaker: region:account_ID :workspace/ workspace_ID`

resources.type	resources.ARN
AWS::KendraRanking::ExecutionPlan	arn:*partition*:kendra-ranking:*region*:*account_ID*:rescore-execution-plan/*rescore_execution_plan_ID*
AWS::Kinesis::Stream	arn:*partition*:kinesis:*region*:*account_ID*:stream/*stream_name*
AWS::Kinesis::StreamConsumer	arn:*partition*:kinesis:*region*:*account_ID*:*stream_type*/*stream_name*/consumer/*consumer_name*:*consumer_creation_timestamp*
AWS::KinesisVideo::Stream	arn:*partition*:kinesisvideo:*region*:*account_ID*:stream/*stream_name*/*creation_time*
AWS::MachineLearning::MlModel	arn:*partition*:machinelearning:*region*:*account_ID*:mlmodel/*model_ID*
AWS::ManagedBlockchain::Network	arn:*partition*:managedblockchain:::networks/*network_name*
AWS::ManagedBlockchain::Node	arn:*partition*:managedblockchain:*region*:*account_ID*:nodes/*node_ID*
AWS::MedicalImaging::Datastore	arn:*partition*:medical-imaging:*region*:*account_ID*:datastore/*data_store_ID*

resources.type	resources.ARN
`AWS::NeptuneGraph::Graph`	`arn:partition:neptune-graph:region:account_ID:graph/graph_ID`
`AWS::PaymentCryptography::Alias`	`arn:partition:payment-cryptography:region:account_ID:alias/alias`
`AWS::PaymentCryptography::Key`	`arn:partition:payment-cryptography:region:account_ID:key/key_ID`
`AWS::PCAConnectorAD::Connector`	`arn:partition:pca-connector-ad:region:account_ID:connector/connector_ID`
`AWS::PCAConnectorSCEP::Connector`	`arn:partition:pca-connector-scep:region:account_ID:connector/connector_ID`
`AWS::QApps:QApp`	`arn:partition:qapps:region:account_ID:application/application_UUID/qapp/qapp_UUID`
`AWS::QBusiness::Application`	`arn:partition:qbusiness:region:account_ID:application/application_ID`
`AWS::QBusiness::DataSource`	`arn:partition:qbusiness:region:account_ID:application/application_ID/index/index_ID/data-source/datasource_ID`

resources.type	resources.ARN
`AWS::QBusiness::Index`	`arn:partition:qbusiness:region:account_ID:application/application_ID/index/index_ID`
`AWS::QBusiness::WebExperience`	`arn:partition:qbusiness:region:account_ID:application/application_ID/web-experience/web_experienc_ID`
`AWS::RDS::DBCluster`	`arn:partition:rds:region:account_ID:cluster/cluster_name`
`AWS::S3::AccessPoint` [3]	`arn:partition:s3:region:account_ID:accesspoint/access_point_name`
`AWS::S3Express::Object`	`arn:partition:s3express:region:account_ID:bucket/bucket_name`
`AWS::S3ObjectLambda::AccessPoint`	`arn:partition:s3-object-lambda:region:account_ID:accesspoint/access_point_name`
`AWS::S3Outposts::Object`	`arn:partition:s3-outposts:region:account_ID:object_path`
`AWS::SageMaker::Endpoint`	`arn:partition:sagemaker:region:account_ID:endpoint/endpoint_name`

resources.type	resources.ARN
`AWS::SageMaker::ExperimentTrialComponent`	`arn:partition :sagemaker: region:account_ID :experiment-trial-component/ experiment_trial_component_name`
`AWS::SageMaker::FeatureGroup`	`arn:partition :sagemaker: region:account_ID :feature-group/ feature_group_name`
`AWS::SCN::Instance`	`arn:partition :scn:region:account_ID :instance/ instance_ID`
`AWS::ServiceDiscovery::Namespace`	`arn:partition :servicediscovery: region:account_ID :namespace/ namespace_ID`
`AWS::ServiceDiscovery::Service`	`arn:partition :servicediscovery: region:account_ID :service/ service_ID`
`AWS::SNS::PlatformEndpoint`	`arn:partition :sns:region:account_ID :endpoint/ endpoint_type /endpoint_name /endpoint_ID`
`AWS::SNS::Topic`	`arn:partition :sns:region:account_ID :topic_name`
`AWS::SQS::Queue`	`arn:partition :sqs:region:account_ID :queue_name`

resources.type	resources.ARN
AWS::SSM::ManagedNode	The ARN must be in one of the following formats: • arn:*partition*:ssm:*region*:*account_ID*:managed-instance/*instance_ID* • arn:*partition*:ec2:*region*:*account_ID*:instance/*instance_ID*
AWS::SSMMessages::ControlChannel	arn:*partition*:ssmmessages:*region*:*account_ID*:control-channel/*control_channel_ID*
AWS::StepFunctions::StateMachine	The ARN must be in one of the following formats: • arn:*partition*:states:*region*:*account_ID*:stateMachine:*stateMachine_name* • arn:*partition*:states:*region*:*account_ID*:stateMachine:*stateMachine_name*/*label_name*
AWS::SWF::Domain	arn:*partition*:swf:*region*:*account_ID*:/domain/*domain_name*
AWS::ThinClient::Device	arn:*partition*:thinclient:*region*:*account_ID*:device/*device_ID*

resources.type	resources.ARN
`AWS::ThinClient::Environment`	`arn:partition :thinclient: region:account_ID :environment/ environment_ID`
`AWS::Timestream::Database`	`arn:partition :timestream: region:account_ID :database/ database_name`
`AWS::Timestream::Table`	`arn:partition :timestream: region:account_ID :database/ database_name /table/table_name`
`AWS::VerifiedPermissions::PolicyStore`	`arn:partition :verifiedpermissions: region:account_ID :policy-store/ policy_store_ID`

[1] For tables with streams enabled, the `resources` field in the data event contains both `AWS::DynamoDB::Stream` and `AWS::DynamoDB::Table`. If you specify `AWS::DynamoDB::Table` for the `resources.type`, it will log both DynamoDB table and DynamoDB streams events by default. To exclude streams events, add a filter on the `eventName` field.

[2] To log all data events for all objects in a specific S3 bucket, use the `StartsWith` operator, and include only the bucket ARN as the matching value. The trailing slash is intentional; do not exclude it.

[3] To log events on all objects in an S3 access point, we recommend that you use only the access point ARN, don't include the object path, and use the `StartsWith` or `NotStartsWith` operators.

For more information about the ARN formats of data event resources, see Actions, resources, and condition keys in the *AWS Identity and Access Management User Guide*.

   b. For each field, choose **+ Condition** to add as many conditions as you need, up to a maximum of 500 specified values for all conditions. For example, to exclude data events for two S3 buckets from data events that are logged on your trail, you can set the field to **resources.ARN**, set the operator for **does not start with**, and then either paste in an S3 bucket ARN, or browse for the S3 buckets for which you do not want to log events.

      To add the second S3 bucket, choose **+ Condition**, and then repeat the preceding instruction, pasting in the ARN for or browsing for a different bucket.

      > **Note**
      >
      > You can have a maximum of 500 values for all selectors on a trail. This includes arrays of multiple values for a selector such as `eventName`. If you have single values for all selectors, you can have a maximum of 500 conditions added to a selector.

   c. Choose **+ Field** to add additional fields as required. To avoid errors, do not set conflicting or duplicate values for fields. For example, do not specify an ARN in one selector to be equal to a value, then specify that the ARN not equal the same value in another selector.

8. To add another data type on which to log data events, choose **Add data event type**. Repeat steps 4 through this step to configure advanced event selectors for the data event type.
9. After you've reviewed and verified your choices, choose **Save changes**.

Update an existing trail to log data events with basic event selectors in the AWS Management Console

Use the following procedure to update an existing trail to log data events using basic event selectors.

1. Sign in to the AWS Management Console and open the CloudTrail console at https://console.aws.amazon.com/cloudtrail/.
2. Open the **Trails** page of the CloudTrail console and choose the trail name.

   > **Note**
   >
   > While you can edit an existing trail to log data events, as a best practice, consider creating a separate trail specifically for logging data events.

3. For **Data events**, choose **Edit**.
4. For Amazon S3 buckets:

   a. For **Data event source**, choose **S3**.

   b. You can choose to log **All current and future S3 buckets**, or you can specify individual buckets or functions. By default, data events are logged for all current and future S3 buckets.

   > **Note**
   >
   > Keeping the default **All current and future S3 buckets** option enables data event logging for all buckets currently in your AWS account and any buckets you create after you finish creating the trail. It also enables logging of data event activity performed by any user or role in your AWS account, even if that activity is performed on a bucket that belongs to another AWS account.
   > If you are creating a trail for a single Region (done by using the AWS CLI), selecting the **Select all S3 buckets in your account** option enables data event logging for all buckets in the same Region as your trail and any buckets you create later in that Region. It will not log data events for Amazon S3 buckets in other Regions in your AWS account.

   c. If you leave the default, **All current and future S3 buckets**, choose to log **Read** events, **Write** events, or both.

   d. To select individual buckets, empty the **Read** and **Write** check boxes for **All current and future S3 buckets**. In **Individual bucket selection**, browse for a bucket on which to log data events. To find specific buckets, type a bucket prefix for the bucket you want. You can select multiple buckets in this window. Choose **Add bucket** to log data events for more buckets. Choose to log **Read** events, such as `GetObject`, **Write** events, such as `PutObject`, or both.

   This setting takes precedence over individual settings you configure for individual buckets. For example, if you specify logging **Read** events for all S3 buckets, and then choose to add a specific bucket for data event logging, **Read** is already selected for the bucket you added. You cannot clear the selection. You can only configure the option for **Write**.

   To remove a bucket from logging, choose **X**.

5. To add another data type on which to log data events, choose **Add data event type**.

6. For Lambda functions:

   a. For **Data event source**, choose **Lambda**.

   b. In **Lambda function**, choose **All regions** to log all Lambda functions, or **Input function as ARN** to log data events on a specific function.

      To log data events for all Lambda functions in your AWS account, select **Log all current and future functions**. This setting takes precedence over individual settings you configure for individual functions. All functions are logged, even if all functions are not displayed.

      > **Note**
      >
      > If you are creating a trail for all Regions, this selection enables data event logging for all functions currently in your AWS account, and any Lambda functions you might create in any Region after you finish creating the trail. If you are creating a trail for a single Region (done by using the AWS CLI), this selection enables data event logging for all functions currently in that Region in your AWS account, and any Lambda functions you might create in that Region after you finish creating the trail. It does not enable data event logging for Lambda functions created in other Regions.
      > Logging data events for all functions also enables logging of data event activity performed by any user or role in your AWS account, even if that activity is performed on a function that belongs to another AWS account.

   c. If you choose **Input function as ARN**, enter the ARN of a Lambda function.

      > **Note**
      >
      > If you have more than 15,000 Lambda functions in your account, you cannot view or select all functions in the CloudTrail console when creating a trail. You can still select the option to log all functions, even if they are not displayed. If you want to log data events for specific functions, you can manually add a function if you know its ARN. You can also finish creating the trail in the console, and then use the AWS CLI and the **put-event-selectors** command to configure data event logging for specific Lambda functions. For more information, see Managing trails with the AWS CLI.

7. To add another data type on which to log data events, choose **Add data event type**.

8. For DynamoDB tables:

   a. For **Data event source**, choose **DynamoDB**.

   b. In **DynamoDB table selection**, choose **Browse** to select a table, or paste in the ARN of a DynamoDB table to which you have access. A DynamoDB table ARN uses the following format:

   ```
   arn:partition:dynamodb:region:account_ID:table/table_name
   ```

   To add another table, choose **Add row**, and browse for a table or paste in the ARN of a table to which you have access.

9. Choose **Save changes**.

Logging data events with the AWS Command Line Interface

You can configure your trails or event data stores to log data events using the AWS CLI.

Topics

- Logging data events for trails with the AWS CLI
- Logging data events for event data stores with the AWS CLI

Logging data events for trails with the AWS CLI

You can configure your trails to log management and data events using the AWS CLI.

Note

- Be aware that if your account is logging more than one copy of management events, you incur charges. There is always a charge for logging data events. For more information, see AWS CloudTrail Pricing.
- You can use either advanced event selectors or basic event selectors, but not both. If you apply advanced event selectors to a trail, any existing basic event selectors are overwritten.
- If your trail uses basic event selectors, you can only log the following resource types:
  - `AWS::DynamoDB::Table`

  - `AWS::Lambda::Function`
  - `AWS::S3::Object`

  To log additional resource types, you'll need to use advanced event selectors. To convert a trail to advanced event selectors, run the **get-event-selectors** command to confirm the current event selectors, and then configure the advanced event selectors to match the coverage of the previous event selectors, then add selectors for any resource types for which you want to log data events.
- You can use advanced event selectors to filter based on the value of the `eventName`, `resources.ARN`, and `readOnly` fields, giving you the ability to log only the data events of interest. For more information about configuring these fields, see AdvancedFieldSelector in the *AWS CloudTrail API Reference* and Filtering data events by using advanced event selectors in this topic.

To see whether your trail is logging management and data events, run the `get-event-selectors` command.

```
aws cloudtrail get-event-selectors --trail-name TrailName
```

The command returns the event selectors for the trail.

Topics

Log events by using advanced event selectors

> **Note**
>
> If you apply advanced event selectors to a trail, any existing basic event selectors are overwritten. Before configuring advanced event selectors, run the **get-event-selectors** command to confirm the current event selectors, and then configure the advanced event

selectors to match the coverage of the previous event selectors, then add selectors for any additional data events you want to log.

The following example creates custom advanced event selectors for a trail named *TrailName* to include read and write management events (by omitting the `readOnly` selector), `PutObject` and `DeleteObject` data events for all Amazon S3 bucket/prefix combinations except for a bucket named `DOC-EXAMPLE-BUCKET` and data events for an AWS Lambda function named `MyLambdaFunction`. Because these are custom advanced event selectors, each set of selectors has a descriptive name. Note that a trailing slash is part of the ARN value for S3 buckets.

```
aws cloudtrail put-event-selectors --trail-name TrailName --advanced-event-selectors
'[
  {
    "Name": "Log readOnly and writeOnly management events",
    "FieldSelectors": [
      { "Field": "eventCategory", "Equals": ["Management"] }
    ]
  },
  {
    "Name": "Log PutObject and DeleteObject events for all but one bucket",
    "FieldSelectors": [
      { "Field": "eventCategory", "Equals": ["Data"] },
      { "Field": "resources.type", "Equals": ["AWS::S3::Object"] },
      { "Field": "eventName", "Equals": ["PutObject","DeleteObject"] },
      { "Field": "resources.ARN", "NotStartsWith": ["arn:aws:s3:::DOC-EXAMPLE-
BUCKET/"] }
    ]
  },
  {
    "Name": "Log data plane actions on MyLambdaFunction",
    "FieldSelectors": [
      { "Field": "eventCategory", "Equals": ["Data"] },
      { "Field": "resources.type", "Equals": ["AWS::Lambda::Function"] },
      { "Field": "resources.ARN", "Equals": ["arn:aws:lambda:us-
east-2:111122223333:function/MyLambdaFunction"] }
    ]
  }
]'
```

The example returns the advanced event selectors that are configured for the trail.

```
{
  "AdvancedEventSelectors": [
    {
      "Name": "Log readOnly and writeOnly management events",
      "FieldSelectors": [
        {
          "Field": "eventCategory",
          "Equals": [ "Management" ]
        }
      ]
    },
    {
      "Name": "Log PutObject and DeleteObject events for all but one bucket",
      "FieldSelectors": [
        {
          "Field": "eventCategory",
          "Equals": [ "Data" ]
        },
        {
          "Field": "resources.type",
          "Equals": [ "AWS::S3::Object" ]
        },
        {
          "Field": "resources.ARN",
          "NotStartsWith": [ "arn:aws:s3:::DOC-EXAMPLE-BUCKET/" ]
        },
      ]
    },
{
      "Name": "Log data plane actions on MyLambdaFunction",
      "FieldSelectors": [
        {
          "Field": "eventCategory",
          "Equals": [ "Data" ]
        },
        {
          "Field": "resources.type",
          "Equals": [ "AWS::Lambda::Function" ]
        },
        {
          "Field": "eventName",
          "Equals": [ "Invoke" ]
        },
        {
          "Field": "resources.ARN",
          "Equals": [ "arn:aws:lambda:us-east-2:111122223333:function/
MyLambdaFunction" ]
        }
      ]
    }
  ],
  "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/TrailName"
}
```

Log all Amazon S3 events for an Amazon S3 bucket by using advanced event selectors

> **Note**
>
> If you apply advanced event selectors to a trail, any existing basic event selectors are overwritten.

The following example shows how to configure your trail to include all data events for all Amazon S3 objects in a specific S3 bucket. The value for S3 events for the `resources.type` field is `AWS::S3::Object`. Because the ARN values for S3 objects and S3 buckets are slightly different, you must add the `StartsWith` operator for `resources.ARN` to capture all events.

```
aws cloudtrail put-event-selectors --trail-name TrailName --region region \
--advanced-event-selectors \
'[
    {
            "Name": "S3EventSelector",
            "FieldSelectors": [
                { "Field": "eventCategory", "Equals": ["Data"] },
                { "Field": "resources.type", "Equals": ["AWS::S3::Object"] },
                { "Field": "resources.ARN", "StartsWith": ["arn:partition:s3:::DOC-
EXAMPLE-BUCKET/"] }
            ]
        }
]'
```

The command returns the following example output.

```
{
    "TrailARN": "arn:aws:cloudtrail:region:account_ID:trail/TrailName",
    "AdvancedEventSelectors": [
        {
            "Name": "S3EventSelector",
            "FieldSelectors": [
                {
                    "Field": "eventCategory",
                    "Equals": [
                        "Data"
                    ]
                },
                {
                    "Field": "resources.type",
                    "Equals": [
                        "AWS::S3::Object"
                    ]
                },
                {
                    "Field": "resources.ARN",
                    "StartsWith": [
                        "arn:partition:s3:::DOC-EXAMPLE-BUCKET/"
                    ]
                }
            ]
        }
    ]
}
```

Log Amazon S3 on AWS Outposts events by using advanced event selectors

Note

If you apply advanced event selectors to a trail, any existing basic event selectors are overwritten.

The following example shows how to configure your trail to include all data events for all Amazon S3 on Outposts objects in your outpost.

```
aws cloudtrail put-event-selectors --trail-name TrailName --region region \
--advanced-event-selectors \
'[
    {
            "Name": "OutpostsEventSelector",
            "FieldSelectors": [
                { "Field": "eventCategory", "Equals": ["Data"] },
                { "Field": "resources.type", "Equals": ["AWS::S3Outposts::Object"] }
            ]
    }
]'
```

The command returns the following example output.

```
{
    "TrailARN": "arn:aws:cloudtrail:region:account_ID:trail/TrailName",
    "AdvancedEventSelectors": [
        {
            "Name": "OutpostsEventSelector",
            "FieldSelectors": [
                {
                    "Field": "eventCategory",
                    "Equals": [
                        "Data"
                    ]
                },
                {
                    "Field": "resources.type",
                    "Equals": [
                        "AWS::S3Outposts::Object"
                    ]
                }
            ]
        }
    ]
}
```

Log events by using basic event selectors

The following is an example result of the **get-event-selectors** command showing basic event selectors. By default, when you create a trail by using the AWS CLI, a trail logs all management events. By default, trails do not log data events.

```
{
    "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/TrailName",
    "EventSelectors": [
        {
            "IncludeManagementEvents": true,
            "DataResources": [],
            "ReadWriteType": "All"
        }
    ]
}
```

To configure your trail to log management and data events, run the `put-event-selectors` command.

The following example shows how to use basic event selectors to configure your trail to include all management and data events for the S3 objects in two S3 bucket prefixes. You can specify from 1 to 5 event selectors for a trail. You can specify from 1 to 250 data resources for a trail.

> **Note**
>
> The maximum number of S3 data resources is 250, if you choose to limit data events by using basic event selectors.

```
aws cloudtrail put-event-selectors --trail-name TrailName --event-selectors
 '[{ "ReadWriteType": "All", "IncludeManagementEvents":true, "DataResources":
 [{ "Type": "AWS::S3::Object", "Values": ["arn:aws:s3:::DOC-EXAMPLE-BUCKET1/prefix",
 "arn:aws:s3:::DOC-EXAMPLE-BUCKET2;/prefix2"] }] }]'
```

The command returns the event selectors that are configured for the trail.

```
{
    "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/TrailName",
    "EventSelectors": [
        {
            "IncludeManagementEvents": true,
            "DataResources": [
                {
                    "Values": [
                        "arn:aws:s3:::DOC-EXAMPLE-BUCKET1/prefix",
                        "arn:aws:s3:::DOC-EXAMPLE-BUCKET2/prefix2",
                    ],
                    "Type": "AWS::S3::Object"
                }
            ],
            "ReadWriteType": "All"
        }
    ]
}
```

Logging data events for event data stores with the AWS CLI

You can configure your event data stores to include data events using the AWS CLI. Use the `create-event-data-store` command to create a new event data store to log data events. Use the `update-event-data-store` command to update the advanced event selectors for an existing event data store.

To see whether your event data store includes data events, run the `get-event-data-store` command.

```
aws cloudtrail get-event-data-store --event-data-store EventDataStoreARN
```

The command returns the settings for the event data store.

```
{
    "EventDataStoreArn": "arn:aws:cloudtrail:us-east-1:111122223333:eventdatastore/
EXAMPLE492-301f-4053-ac5e-EXAMPLE6441aa",
    "Name": "ebs-data-events",
    "Status": "ENABLED",
    "AdvancedEventSelectors": [
        {
            "Name": "Log all EBS direct APIs on EBS snapshots",
            "FieldSelectors": [
                {
                    "Field": "eventCategory",
                    "Equals": [
                        "Data"
                    ]
                },
                {
                    "Field": "resources.type",
                    "Equals": [
                        "AWS::EC2::Snapshot"
                    ]
                }
            ]
        }
    ],
    "MultiRegionEnabled": true,
    "OrganizationEnabled": false,
    "BillingMode": "EXTENDABLE_RETENTION_PRICING",
    "RetentionPeriod": 366,
    "TerminationProtectionEnabled": true,
    "CreatedTimestamp": "2023-11-04T15:57:33.701000+00:00",
    "UpdatedTimestamp": "2023-11-20T20:37:34.228000+00:00"
}
```

Topics

- Include all Amazon S3 events for a bucket
- Include Amazon S3 on AWS Outposts events

Include all Amazon S3 events for a bucket

The following example shows how to create an event data store to include all data events for all Amazon S3 objects in a specific S3 bucket. The value for S3 events for the `resources.type` field is `AWS::S3::Object`. Because the ARN values for S3 objects and S3 buckets are slightly different, you must add the `StartsWith` operator for `resources.ARN` to capture all events.

```
aws cloudtrail create-event-data-store --name "EventDataStoreName" --multi-region-
enabled \
--advanced-event-selectors \
'[
    {
            "Name": "S3EventSelector",
            "FieldSelectors": [
                { "Field": "eventCategory", "Equals": ["Data"] },
                { "Field": "resources.type", "Equals": ["AWS::S3::Object"] },
                { "Field": "resources.ARN", "StartsWith": ["arn:partition:s3:::DOC-
EXAMPLE-BUCKET/"] }
            ]
        }
]'
```

The command returns the following example output.

```
{
    "EventDataStoreArn": "arn:aws:cloudtrail:us-east-1:111122223333:eventdatastore/
EXAMPLE492-301f-4053-ac5e-EXAMPLE441aa",
    "Name": "EventDataStoreName",
    "Status": "ENABLED",
    "AdvancedEventSelectors": [
        {
            "Name": "S3EventSelector",
            "FieldSelectors": [
                {
                    "Field": "eventCategory",
                    "Equals": [
                        "Data"
                    ]
                },
                {
                    "Field": "resources.ARN",
                    "StartsWith": [
                        "arn:partition:s3:::DOC-EXAMPLE-BUCKET/"
                    ]
                },
                {
                    "Field": "resources.type",
                    "Equals": [
                        "AWS::S3::Object"
                    ]
                }
            ]
        }
    ],
    "MultiRegionEnabled": true,
    "OrganizationEnabled": false,
    "BillingMode": "EXTENDABLE_RETENTION_PRICING",
    "RetentionPeriod": 366,
    "TerminationProtectionEnabled": true,
    "CreatedTimestamp": "2023-11-04T15:57:33.701000+00:00",
    "UpdatedTimestamp": "2023-11-20T20:49:21.766000+00:00"
}
```

Include Amazon S3 on AWS Outposts events

The following example shows how to create an event data store that includes all data events for all Amazon S3 on Outposts objects in your outpost.

```
aws cloudtrail create-event-data-store --name EventDataStoreName \
--advanced-event-selectors \
'[
    {
            "Name": "OutpostsEventSelector",
            "FieldSelectors": [
                { "Field": "eventCategory", "Equals": ["Data"] },
                { "Field": "resources.type", "Equals": ["AWS::S3Outposts::Object"] }
            ]
        }
]'
```

The command returns the following example output.

```
{
    "EventDataStoreArn": "arn:aws:cloudtrail:us-east-1:111122223333:eventdatastore/
EXAMPLEb4a8-99b1-4ec2-9258-EXAMPLEc890",
    "Name": "EventDataStoreName",
    "Status": "CREATED",
    "AdvancedEventSelectors": [
        {
            "Name": "OutpostsEventSelector",
            "FieldSelectors": [
                {
                    "Field": "eventCategory",
                    "Equals": [
                        "Data"
                    ]
                },
                {
                    "Field": "resources.type",
                    "Equals": [
                        "AWS::S3Outposts::Object"
                    ]
                }
            ]
        }
    ],
    "MultiRegionEnabled": true,
    "OrganizationEnabled": false,
    "BillingMode": "EXTENDABLE_RETENTION_PRICING",
    "RetentionPeriod": 366,
    "TerminationProtectionEnabled": true,
    "CreatedTimestamp": "2023-02-20T21:00:17.673000+00:00",
    "UpdatedTimestamp": "2023-02-20T21:00:17.820000+00:00"
}
```

Filtering data events by using advanced event selectors

This section describes how you can use advanced event selectors to create fine-grained selectors, which help you control costs by only logging the specific data events of interest.

For example:

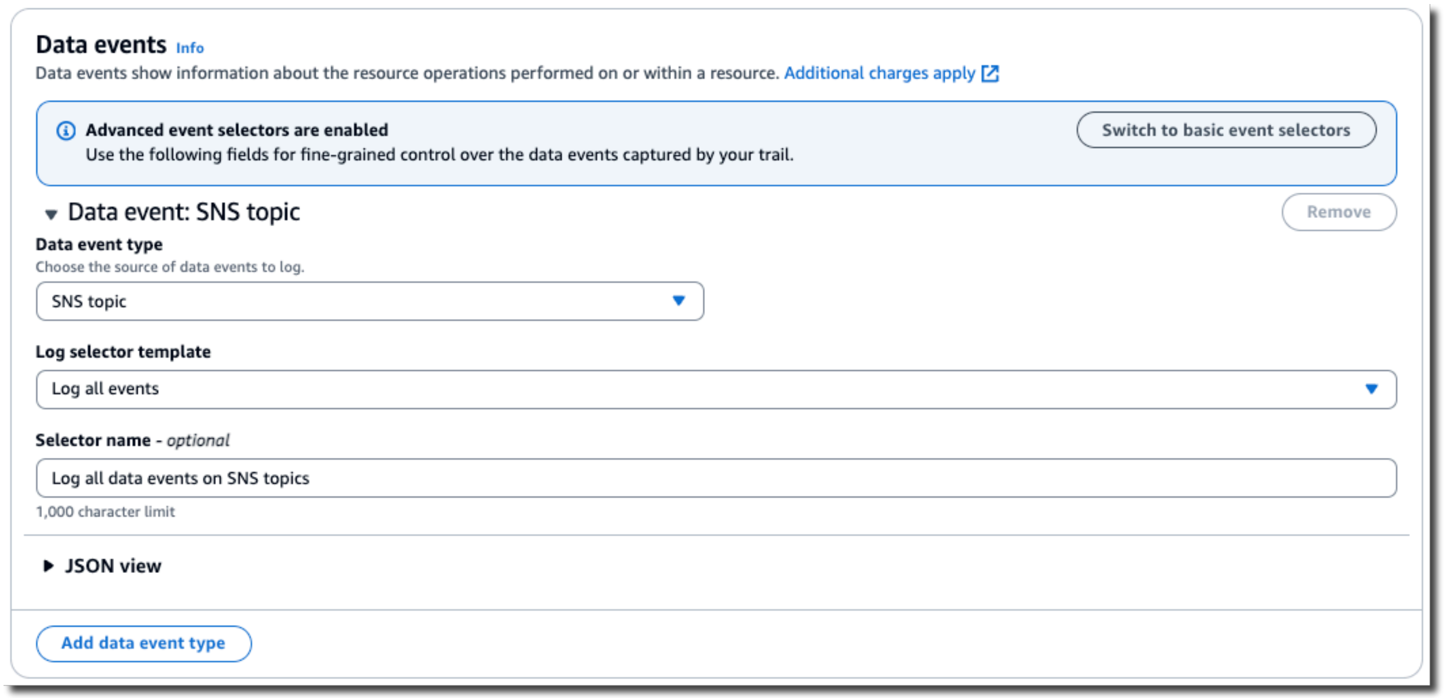
- You can include or exclude specific API calls by adding a filter on the `eventName` field.
- You can include or exclude logging for specific resources by adding a filter on the `resources.ARN` field. For example, if you were logging S3 data events, you could exclude logging for the S3 bucket for your trail.
- You can choose to log only write-only events or read-only events by adding a filter on the `readOnly` field.

The following table provides additional information about the configurable fields for advanced event selectors.

Field	Required	Valid operators	Description
`eventCategory`	Yes	`Equals`	This field is set to `Data` to log data events.
`resources.type`	Yes	`Equals`	This field is used to select the resource type for which you want to log data events. The Data events table shows the possible values.
`readOnly`	No	`Equals`	This is an optional field used to include or exclude data events based on the `readOnly` value. A value of `true` logs only read events. A value of `false` logs only write events. If you do not add this field, CloudTrail logs both read and write events.

Field	Required	Valid operators	Description
`eventName`	No	Any	This is an optional filed used to filter in or filter out any data event logged to CloudTrail, such as `PutBucket` or `GetSnapshotBlock` . If you're using the AWS CLI, you can specify multiple values by separating each value with a comma. If you're using the console, you can specify multiple values by creating a condition for each `eventName` you want to filter on.
`resources.ARN`	No	Any	This is an optional field used to exclude or include data events for a specific resource by providing the `resources.ARN` . You can use any operator with `resources.ARN` , but if you use `Equals` or `NotEquals` , the value must exactly match the ARN of a valid resource for the `resoureces.type` you've specified. If you're using the AWS CLI, you can specify multiple values by separating each value with a comma. If you're using the console, you can specify multiple values by creating a condition for each `resources.ARN` you want to filter on.

To log data events using the CloudTrail console, you choose the **Data events** option and then select the **Data event type** of interest when you are creating or updating a trail or event data store. The Data events table shows the possible data event types you can choose on the CloudTrail console.



Data events Info

Data events show information about the resource operations performed on or within a resource. Additional charges apply

Advanced event selectors are enabled

Use the following fields for fine-grained control over the data events captured by your trail.

▼ Data event: SNS topic

Data event type

Choose the source of data events to log.

SNS topic

Log selector template

Log all events

Selector name - *optional*

Log all data events on SNS topics

1,000 character limit

▶ **JSON view**

To log data events with the AWS CLI, configure the `--advanced-event-selector` parameter to set the `eventCategory` equal to `Data` and the `resources.type` value equal to the resource type value for which you want to log data events. The Data events table lists the available resource types.

For example, if you wanted to log data events for all Cognito Identity pools, you'd configure the `--advanced-event-selectors` parameter to look like this:

```
--advanced-event-selectors '[
    {
       "Name": "Log Cognito data events on Identity pools",
       "FieldSelectors": [
         { "Field": "eventCategory", "Equals": ["Data"] },
         { "Field": "resources.type", "Equals": ["AWS::Cognito::IdentityPool"] }
       ]
     }
]'
```

The preceding example logs all Cognito data events on Identity pools. You can further refine the advanced event selectors to filter on the `eventName`, `readOnly`, and `resources.ARN` fields to log specific events of interest or exclude events that aren't of interest.

You can configure advanced event selectors to filter data events based on multiple conditions. For example, you can configure advanced event selectors to log all Amazon S3 `PutObject` and `DeleteObject` API calls but exclude event logging for a specific S3 bucket as shown in the following example. Replace *`DOC-EXAMPLE-BUCKET`* with the name of your bucket.

```
--advanced-event-selectors
'[
  {
    "Name": "Log PutObject and DeleteObject events for all but one bucket",
    "FieldSelectors": [
      { "Field": "eventCategory", "Equals": ["Data"] },
      { "Field": "resources.type", "Equals": ["AWS::S3::Object"] },
      { "Field": "eventName", "Equals": ["PutObject","DeleteObject"] },
      { "Field": "resources.ARN", "NotStartsWith": ["arn:aws:s3:::DOC-EXAMPLE-
BUCKET/"] }
    ]
  }
]'
```

You can use advanced event selectors to log both management and data events. To log data events for multiple resource types, add a field selector statement for each resource type that you want to log data events for.

> **Note**
>
> Trails can use either basic event selectors or advanced event selectors, but not both. If you apply advanced event selectors to a trail, any existing basic event selectors are overwritten.

Topics

Filtering data events by `eventName`

Using advanced event selectors, you can include or exclude events based on the value of the `eventName` field. Filtering on the `eventName` can help control costs, because you avoid incurring costs when the AWS service you're logging data events for adds support for new data APIs.

You can use any operator with the `eventName` field. You can use it to filter in or filter out any data event logged to CloudTrail, such as `PutBucket` or `GetSnapshotBlock`.

Topics

- Filtering data events by eventName using the AWS Management Console
- Filtering data events by eventName using the AWS CLI

Filtering data events by `eventName` using the AWS Management Console

Take the following steps to filter on the `eventName` field using the CloudTrail console.

1. Follow the steps in the create trail procedure, or follow the steps in the create event data store procedure.
2. As you follow the steps to create the trail or event data store, make the following selections:

   a. Choose **Data events**.

   b. Choose the **Data event type** for which you want to log data events.

   c. For **Log selector template**, choose **Custom**.

   d. (Optional) In **Selector name**, enter a name to identify your selector. The selector name is a descriptive name for an advanced event selector, such as "Log data events for only two S3 buckets". The selector name is listed as `Name` in the advanced event selector and is viewable if you expand the **JSON view**.

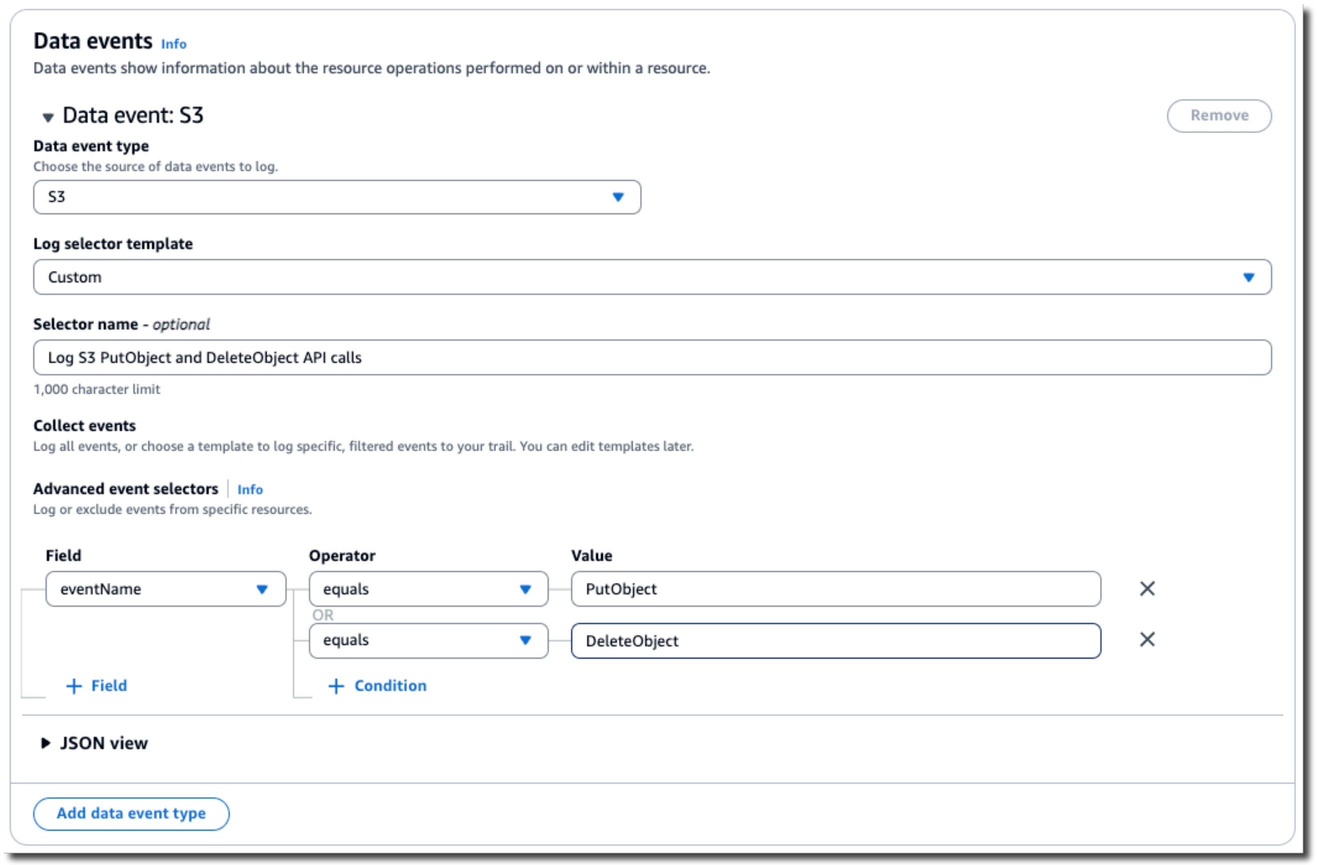
   e. In **Advanced event selectors**, do the following to filter on the `eventName`:

      i. For **Field**, choose **eventName**.

      ii. For **Operator**, choose the condition operator. In this example, we'll choose **equals** because we want to log a specific API call.

      iii. For **Value**, enter the name of the event you want to filter on.

      iv. To filter on another `eventName`, choose **+ Condition**.



Data events Info
Data events show information about the resource operations performed on or within a resource.
Data event: S3
Remove

Data event type
Choose the source of data events to log.
S3

Log selector template
Custom

Selector name - *optional*
Log S3 PutObject and DeleteObject API calls
1,000 character limit

Collect events
Log all events, or choose a template to log specific, filtered events to your trail. You can edit templates later.

Advanced event selectors | Info
Log or exclude events from specific resources.

Field	Operator	Value
eventName	equals	PutObject
	OR equals	DeleteObject

f. Choose **+Field** to add filters on other fields.

Filtering data events by `eventName` using the AWS CLI

Using the AWS CLI, you can filter on the `eventName` field to include or exclude specific events.

If you're updating an existing trail or event data store to log additional event selectors, get the current event selectors by running the [`get-event-selectors`] command for a trail, or the [`get-event-data-store`] command for an event data store. Then, update your event selectors to add a field selector for each data resource type that you want to log.

The following example logs S3 data events on a trail. The `--advanced-event-selectors` are configured to only log data events for the `GetObject`, `PutObject`, and `DeleteObject` API calls.

```
aws cloudtrail put-event-selectors \
--trail-name trailName \
--advanced-event-selectors '[
  {
    "Name": "Log GetObject, PutObject and DeleteObject S3 data events",
    "FieldSelectors": [
      { "Field": "eventCategory", "Equals": ["Data"] },
      { "Field": "resources.type", "Equals": ["AWS::S3::Object"] },
      { "Field": "eventName", "Equals": ["GetObject","PutObject","DeleteObject"] }
    ]
  }
]'
```

The next example creates a new event data store that logs data events for EBS Direct APIs but excludes `ListChangedBlocks` API calls. You can use the update-event-data-store command to update an existing event data store.

```
aws cloudtrail create-event-data-store \
--name "eventDataStoreName"
--advanced-event-selectors '[
    {
        "Name": "Log all EBS Direct API data events except ListChangedBlocks",
        "FieldSelectors": [
            { "Field": "eventCategory", "Equals": ["Data"] },
            { "Field": "resources.type", "Equals": ["AWS::EC2::Snapshot"] },
            { "Field": "eventName", "NotEquals": ["ListChangedBlocks"] }
         ]
    }
]'
```

Filtering data events by `resources.ARN`

Using advanced event selectors, you can filter on the value of the `resources.ARN` field.

You can use any operator with `resources.ARN` , but if you use `Equals` or `NotEquals`, the value must exactly match the ARN of a valid resource for the `resources.type` value you've specified. To log all data events for all objects in a specific S3 bucket, use the `StartsWith` operator, and include only the bucket ARN as the matching value.

The following table shows the valid ARN format for each `resources.type`.

Note

You can't use the `resources.ARN` field to filter resource types that do not have ARNs.

resources.type	resources.ARN
`AWS::DynamoDB::Table` [1]	`arn:partition:dynamodb:region:account_ID:table/table_name`
`AWS::Lambda::Function`	`arn:partition:lambda:region:account_ID:function:function_name`
`AWS::S3::Object` [2]	`arn:partition:s3:::DOC-EXAMPLE-BUCKET/` `arn:partition:s3:::DOC-EXAMPLE-BUCKET/object_or_file_name/`
`AWS::AppConfig::Configuration`	`arn:partition:appconfig:region:account_ID:application/application_ID/environment/environment_ID/configuration/configuration_profile_ID`
`AWS::B2BI::Transformer`	`arn:partition:b2bi:region:account_ID:transformer/transformer_ID`
`AWS::Bedrock::AgentAlias`	`arn:partition:bedrock:region:account_ID:agent-alias/agent_ID/alias_ID`
`AWS::Bedrock::FlowAlias`	`arn:partition:bedrock:region:account_ID:flow/flow_ID/alias/alias_ID`
`AWS::Bedrock::Guardrail`	`arn:partition:bedrock:region:account_ID:guardrail/guardrail_ID`

resources.type	resources.ARN
`AWS::Bedrock::KnowledgeBase`	`arn:partition:bedrock:region:account_ID:knowledge-base/knowledge_base_ID`
`AWS::Cassandra::Table`	`arn:partition:cassandra:region:account_ID:keyspace/keyspace_name/table/table_name`
`AWS::CloudFront::KeyValueStore`	`arn:partition:cloudfront:region:account_ID:key-value-store/KVS_name`
`AWS::CloudTrail::Channel`	`arn:partition:cloudtrail:region:account_ID:channel/channel_UUID`
`AWS::CodeWhisperer::Customization`	`arn:partition:codewhisperer:region:account_ID:customization/customization_ID`
`AWS::CodeWhisperer::Profile`	`arn:partition:codewhisperer:region:account_ID:profile/profile_ID`
`AWS::Cognito::IdentityPool`	`arn:partition:cognito-identity:region:account_ID:identitypool/identity_pool_ID`
`AWS::DynamoDB::Stream`	`arn:partition:dynamodb:region:account_ID:table/table_name/stream/date_time`

resources.type	resources.ARN
AWS::EC2::Snapshot	arn:*partition*:ec2:*region*::snapshot/*snapshot_ID*
AWS::EMRWAL::Workspace	arn:*partition*:emrwal:*region*:*account_ID*:workspace/*workspace_name*
AWS::FinSpace::Environment	arn:*partition*:finspace:*region*:*account_ID*:environment/*environment_ID*
AWS::Glue::Table	arn:*partition*:glue:*region*:*account_ID*:table/*database_name*/*table_name*
AWS::GreengrassV2::ComponentVersion	arn:*partition*:greengrass:*region*:*account_ID*:components/*component_name*
AWS::GreengrassV2::Deployment	arn:*partition*:greengrass:*region*:*account_ID*:deployments/*deployment_ID*
AWS::GuardDuty::Detector	arn:*partition*:guardduty:*region*:*account_ID*:detector/*detector_ID*
AWS::IoT::Certificate	arn:*partition*:iot:*region*:*account_ID*:cert/*certificate_ID*
AWS::IoT::Thing	arn:*partition*:iot:*region*:*account_ID*:thing/*thing_ID*

resources.type	resources.ARN
AWS::IoTSiteWise::Asset	arn:*partition*:iotsitewise:*region*:*account_ID*:asset/*asset_ID*
AWS::IoTSiteWise::TimeSeries	arn:*partition*:iotsitewise:*region*:*account_ID*:timeseries/*timeseries_ID*
AWS::IoTTwinMaker::Entity	arn:*partition*:iottwinmaker:*region*:*account_ID*:workspace/*workspace_ID*/entity/*entity_ID*
AWS::IoTTwinMaker::Workspace	arn:*partition*:iottwinmaker:*region*:*account_ID*:workspace/*workspace_ID*
AWS::KendraRanking::ExecutionPlan	arn:*partition*:kendra-ranking:*region*:*account_ID*:rescore-execution-plan/*rescore_execution_plan_ID*
AWS::Kinesis::Stream	arn:*partition*:kinesis:*region*:*account_ID*:stream/*stream_name*
AWS::Kinesis::StreamConsumer	arn:*partition*:kinesis:*region*:*account_ID*:*stream_type*/*stream_name*/consumer/*consumer_name*:*consumer_creation_timestamp*
AWS::KinesisVideo::Stream	arn:*partition*:kinesisvideo:*region*:*account_ID*:stream/*stream_name*/*creation_time*

resources.type	resources.ARN
`AWS::MachineLearning::MlModel`	`arn:`*`partition`*` :machinelearning: `*`region`*`:`*`account_ID`*` :mlmodel/ `*`model_ID`*
`AWS::ManagedBlockchain::Network`	`arn:`*`partition`*` :managedblockchain:::networks/ `*`network_name`*
`AWS::ManagedBlockchain::Node`	`arn:`*`partition`*` :managedblockchain: `*`region`*`:`*`account_ID`*` :nodes/`*`node_ID`*
`AWS::MedicalImaging::Datastore`	`arn:`*`partition`*` :medical-imaging: `*`region`*`:`*`account_ID`*` :datastore/ `*`data_store_ID`*
`AWS::NeptuneGraph::Graph`	`arn:`*`partition`*` :neptune-graph: `*`region`*`:`*`account_ID`*` :graph/`*`graph_ID`*
`AWS::PaymentCryptography::Alias`	`arn:`*`partition`*` :payment-cryptography: `*`region`*`:`*`account_ID`*` :alias/`*`alias`*
`AWS::PaymentCryptography::Key`	`arn:`*`partition`*` :payment-cryptography: `*`region`*`:`*`account_ID`*` :key/`*`key_ID`*
`AWS::PCAConnectorAD::Connector`	`arn:`*`partition`*` :pca-connector-ad: `*`region`*`:`*`account_ID`*` :connector/ `*`connector_ID`*
`AWS::PCAConnectorSCEP::Connector`	`arn:`*`partition`*` :pca-connector-scep: `*`region`*`:`*`account_ID`*` :connector/ `*`connector_ID`*

resources.type	resources.ARN
`AWS::QApps:QApp`	arn:*partition* :qapps:*region*:*account_ID* :application/ *application_UUID* / qapp/*qapp_UUID*
`AWS::QBusiness::Application`	arn:*partition* :qbusiness: *region*:*account_ID* :application/ *application_ID*
`AWS::QBusiness::DataSource`	arn:*partition* :qbusiness: *region*:*account_ID* :application/ *application_ID* /index/*index_ID*/ data-source/ *datasource_ID*
`AWS::QBusiness::Index`	arn:*partition* :qbusiness: *region*:*account_ID* :application/ *application_ID* /index/*index_ID*
`AWS::QBusiness::WebExperience`	arn:*partition* :qbusiness: *region*:*account_ID* :application/ *application_ID* /web-experience/ *web_experienc_ID*
`AWS::RDS::DBCluster`	arn:*partition* :rds:*region*:*account_ID* :cluster/ *cluster_name*
`AWS::S3::AccessPoint` [3]	arn:*partition* :s3:*region*:*account_ID* :accesspoint/ *access_point_name*
`AWS::S3Express::Object`	arn:*partition* :s3express: *region*:*account_ID* :bucket/*bucket_name*

resources.type	resources.ARN
AWS::S3ObjectLambda::AccessPoint	arn:*partition*:s3-object-lambda:*region*:*account_ID*:accesspoint/*access_point_name*
AWS::S3Outposts::Object	arn:*partition*:s3-outposts:*region*:*account_ID*:*object_path*
AWS::SageMaker::Endpoint	arn:*partition*:sagemaker:*region*:*account_ID*:endpoint/*endpoint_name*
AWS::SageMaker::ExperimentTrialComponent	arn:*partition*:sagemaker:*region*:*account_ID*:experiment-trial-component/*experiment_trial_component_name*
AWS::SageMaker::FeatureGroup	arn:*partition*:sagemaker:*region*:*account_ID*:feature-group/*feature_group_name*
AWS::SCN::Instance	arn:*partition*:scn:*region*:*account_ID*:instance/*instance_ID*
AWS::ServiceDiscovery::Namespace	arn:*partition*:servicediscovery:*region*:*account_ID*:namespace/*namespace_ID*
AWS::ServiceDiscovery::Service	arn:*partition*:servicediscovery:*region*:*account_ID*:service/*service_ID*

resources.type	resources.ARN
`AWS::SNS::PlatformEndpoint`	`arn:partition :sns:region:account_ID :endpoint/ endpoint_type /endpoint_name /endpoint_ID`
`AWS::SNS::Topic`	`arn:partition :sns:region:account_ID :topic_name`
`AWS::SQS::Queue`	`arn:partition :sqs:region:account_ID :queue_name`
`AWS::SSM::ManagedNode`	The ARN must be in one of the following formats: - `arn:partition :ssm:region:account_ID :managed-instance/ instance_ID` - `arn:partition :ec2:region:account_ID :instance / instance_ID`
`AWS::SSMMessages::ControlChannel`	`arn:partition :ssmmessages: region:account_ID :control-channel/ control_channel_ID`

resources.type	resources.ARN
`AWS::StepFunctions::StateMachine`	The ARN must be in one of the following formats: - `arn:partition:states:region:account_ID:stateMachine:stateMachine_name` - `arn:partition:states:region:account_ID:stateMachine:stateMachine_name/label_name`
`AWS::SWF::Domain`	`arn:partition:swf:region:account_ID:/domain/domain_name`
`AWS::ThinClient::Device`	`arn:partition:thinclient:region:account_ID:device/device_ID`
`AWS::ThinClient::Environment`	`arn:partition:thinclient:region:account_ID:environment/environment_ID`
`AWS::Timestream::Database`	`arn:partition:timestream:region:account_ID:database/database_name`
`AWS::Timestream::Table`	`arn:partition:timestream:region:account_ID:database/database_name/table/table_name`

resources.type	resources.ARN
AWS::VerifiedPermissions::PolicyStore	`arn:partition :verifiedpermissions: region:account_ID :policy-store/ policy_store_ID`

[1] For tables with streams enabled, the `resources` field in the data event contains both `AWS::DynamoDB::Stream` and `AWS::DynamoDB::Table`. If you specify `AWS::DynamoDB::Table` for the `resources.type`, it will log both DynamoDB table and DynamoDB streams events by default. To exclude streams events, add a filter on the `eventName` field.

[2] To log all data events for all objects in a specific S3 bucket, use the `StartsWith` operator, and include only the bucket ARN as the matching value. The trailing slash is intentional; do not exclude it.

[3] To log events on all objects in an S3 access point, we recommend that you use only the access point ARN, don't include the object path, and use the `StartsWith` or `NotStartsWith` operators.

Topics

- Filtering data events by resources.ARN using the AWS Management Console
- Filtering data events by resources.ARN using the AWS CLI

Filtering data events by `resources.ARN` using the AWS Management Console

Take the following steps to filter on the `resources.ARN` field using the CloudTrail console.

1. Follow the steps in the create trail procedure, or follow the steps in the create event data store procedure.
2. As you follow the steps to create the trail or event data store, make the following selections:

    a. Choose **Data events**.

    b. Choose the **Data event type** for which you want to log data events.

    c. For **Log selector template**, choose **Custom**.

    d. (Optional) In **Selector name**, enter a name to identify your selector. The selector name is a descriptive name for an advanced event selector, such as "Log data events for only two S3 buckets". The selector name is listed as `Name` in the advanced event selector and is viewable if you expand the **JSON view**.

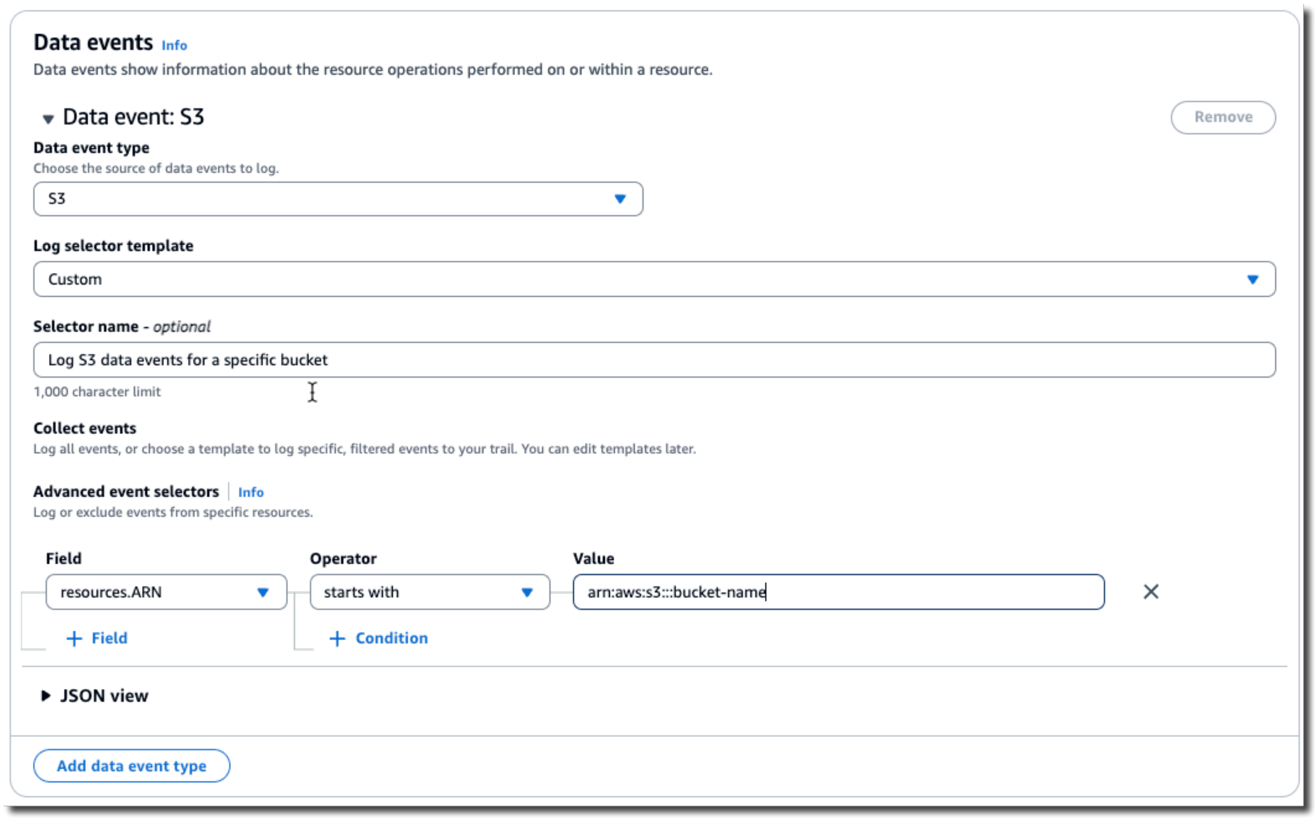
e. In **Advanced event selectors**, do the following to filter on the `resources.ARN`:

   i. For **Field**, choose **resources.ARN**.

   ii. For **Operator**, choose the condition operator. In this example, we'll choose **starts with** because we want to log data events for a specific S3 bucket.

   iii. For **Value**, enter the ARN for your resource type (for example, *`arn:aws:s3:::DOC-EXAMPLE-BUCKET`*).

   iv. To filter another `resources.ARN`, choose **+ Condition**.



Data events Info

Data events show information about the resource operations performed on or within a resource.

▾ Data event: S3

Data event type

Choose the source of data events to log.

S3

Log selector template

Custom

Selector name - *optional*

Log S3 data events for a specific bucket

1,000 character limit

Collect events

Log all events, or choose a template to log specific, filtered events to your trail. You can edit templates later.

Advanced event selectors | Info

Log or exclude events from specific resources.

Field	Operator	Value
resources.ARN	starts with	arn:aws:s3:::bucket-name

f. Choose **+Field** to add filters on other fields.

Filtering data events by `resources.ARN` using the AWS CLI

Using the AWS CLI, you can filter on the `resources.ARN` field to log events for a specific ARN or exclude logging for a specific ARN.

If you're updating an existing trail or event data store to log additional event selectors, get the current event selectors by running the `get-event-selectors` command for a trail, or the `get-event-data-store` command for an event data store. Then, update your event selectors to add a field selector for each data resource type that you want to log.

The following example shows how to configure your trail to include all data events for all Amazon S3 objects in a specific S3 bucket. The value for S3 events for the `resources.type` field is `AWS::S3::Object`. Because the ARN values for S3 objects and S3 buckets are slightly different, you must add the `StartsWith` operator for `resources.ARN` to capture all events.

```
aws cloudtrail put-event-selectors \
--trail-name TrailName \
--region region \
--advanced-event-selectors \
'[
    {
        "Name": "S3EventSelector",
        "FieldSelectors": [
            { "Field": "eventCategory", "Equals": ["Data"] },
                { "Field": "resources.type", "Equals": ["AWS::S3::Object"] },
                { "Field": "resources.ARN", "StartsWith": ["arn:aws:s3:::DOC-EXAMPLE-BUCKET/"] }
            ]
    }
]'
```

Filtering data events by `readOnly` value

Using advanced event selectors, you can filter based on the value of the `readOnly` field.

You can only use the `Equals` operator with the `readOnly` field. You can set the `readOnly` value to `true` or `false`. If you do not add this field, CloudTrail logs both read and write events. A value of `true` logs only read events. A value of `false` logs only write events.

Topics

- Filtering data events by readOnly value using the AWS Management Console
- Filtering data events by readOnly value using the AWS CLI

Filtering data events by `readOnly` value using the AWS Management Console

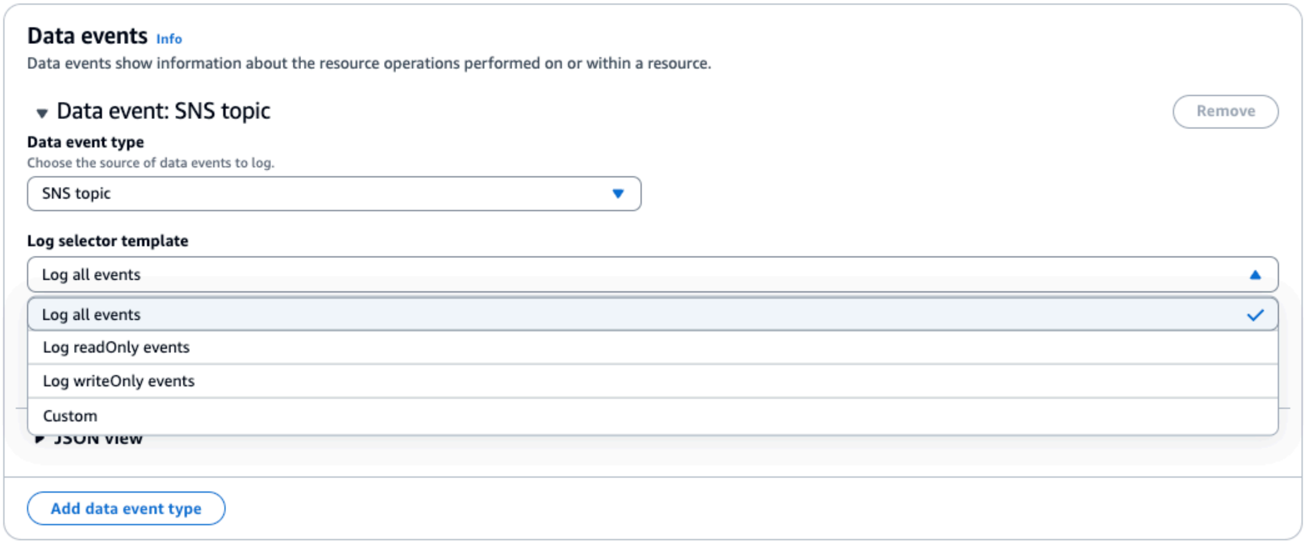
Take the following steps to filter on the `readOnly` field using the CloudTrail console.

1. Follow the steps in the create trail procedure, or follow the steps in the create event data store procedure.
2. As you follow the steps to create the trail or event data store, make the following selections:

   a. Choose **Data events**.

   b. Choose the **Data event type** for which you want to log data events.

   c. For **Log selector template**, choose the appropriate template for your use case.



Data events Info

Data events show information about the resource operations performed on or within a resource.

▾ Data event: SNS topic

Data event type

Choose the source of data events to log.

SNS topic

Log selector template

Log all events

If you plan to do this	Choose this log selector template
Log read events only and apply no other filters (for example, on the `resources.ARN` value).	**Log readOnly events**
Log write events only and apply no other filters (for example, on the `resources.ARN` value).	**Log writeOnly events**
Filter on the `readOnly` value and apply additional filters (for example, on the `resources.ARN` value).	**Custom** In **Advanced event selectors**, do the following to filter on the `readOnly` value:

If you plan to do this	Choose this log selector template
	To log write events a. For **Field**, choose **readOnly**. b. For **Operator**, choose **equals**. c. For **Value**, enter **`false`**. d. Choose **+Field** to add filters on other fields. **To log read events** a. For **Field**, choose **readOnly**. b. For **Operator**, choose **equals**. c. For **Value**, enter **`true`**. d. Choose **+Field** to add filters on other fields.

Filtering data events by `readOnly` value using the AWS CLI

Using the AWS CLI, you can filter on the `readOnly` field.

You can only use the `Equals` operator with the `readOnly` field. You can set the `readOnly` value to `true` or `false`. If you do not add this field, CloudTrail logs both read and write events. A value of `true` logs only read events. A value of `false` logs only write events.

If you're updating an existing trail or event data store to log additional event selectors, get the current event selectors by running the `get-event-selectors` command for a trail, or the `get-event-data-store` command for an event data store. Then, update your event selectors to add a field selector for each data resource type that you want to log.

The following example shows how to configure your trail to log read-only data events for all Amazon S3 objects.

```
aws cloudtrail put-event-selectors \
--trail-name TrailName \
--region region \
--advanced-event-selectors '[
    {
            "Name": "Log read-only S3 data events",
            "FieldSelectors": [
                { "Field": "eventCategory", "Equals": ["Data"] },
                { "Field": "resources.type", "Equals": ["AWS::S3::Object"] },
                { "Field": "readOnly", "Equals": ["true"] }
            ]
    }
]'
```

The next example creates a new event data store that logs only write-only data events for EBS Direct APIs. You can use the update-event-data-store command to update an existing event data store.

```
aws cloudtrail create-event-data-store \
--name "eventDataStoreName" \
--advanced-event-selectors \
'[
    {
            "Name": "Log write-only EBS Direct API data events",
            "FieldSelectors": [
                { "Field": "eventCategory", "Equals": ["Data"] },
                { "Field": "resources.type", "Equals": ["AWS::EC2::Snapshot"] },
                { "Field": "readOnly", "Equals": ["false"] }
            ]
     }
]'
```

Logging data events for AWS Config compliance

If you are using AWS Config conformance packs to help your enterprise maintain compliance with formalized standards such as those required by Federal Risk and Authorization Management Program (FedRAMP) or National Institute of Standards and Technology (NIST), conformance packs for compliance frameworks generally require you to log data events for Amazon S3 buckets, at minimum. Conformance packs for compliance frameworks include a managed rule called `cloudtrail-s3-dataevents-enabled` that checks for S3 data event logging in your account. Many conformance packs that are not associated with compliance frameworks also require S3 data event logging. The following are examples of conformance packs that include this rule.

- Operational Best Practices for AWS Well-Architected Framework Security Pillar

- Operational Best Practices for FDA Title 21 CFR Part 11
- Operational Best Practices for FFIEC
- Operational Best Practices for FedRAMP(Moderate)
- Operational Best Practices for HIPAA Security
- Operational Best Practices for K-ISMS
- Operational Best Practices for Logging

For a full list of sample conformance packs available in AWS Config, see Conformance pack sample templates in the *AWS Config Developer Guide.*

Logging data events with the AWS SDKs

Run the GetEventSelectors operation to see whether your trail is logging data events. You can configure your trails to log data events by running the PutEventSelectors operation. For more information, see the AWS CloudTrail API Reference.

Run the GetEventDataStore operation to see whether your event data store is logging data events. You can configure your event data stores to include data events by running the CreateEventDataStore or UpdateEventDataStore operations and specifying advanced event selectors. For more information, see Create, update, and manage event data stores with the AWS CLI and the AWS CloudTrail API Reference.

Logging Insights events

AWS CloudTrail Insights help AWS users identify and respond to unusual activity associated with API calls and API error rates by continuously analyzing CloudTrail management events. CloudTrail Insights analyzes your normal patterns of API call volume and API error rates, also called the *baseline*, and generates Insights events when the call volume or error rates are outside normal patterns. Insights events on API call volume are generated for `write` management APIs, and Insights events on API error rate are generated for both `read` and `write` management APIs.

> **Note**
>
> To log Insights events on API call volume, the trail or event data store must log `write` management events. To log Insights events on API error rate, the trail or event data store must log `read` or `write` management events.

CloudTrail Insights analyzes management events that occur in a single Region, not globally. A CloudTrail Insights event is generated in the same Region as its supporting management events are generated.

Additional charges apply for Insights events. You will be charged separately if you enable Insights for both trails and event data stores. For more information, see AWS CloudTrail Pricing.

Contents

Understanding Insights events delivery

Unlike other types of events that CloudTrail captures, Insights events are logged only when CloudTrail detects changes in your account's API usage that differ significantly from the account's typical usage patterns.

Where CloudTrail delivers events and how long it takes to receive Insights events differs between trails and event data stores.

Insights events delivery for trails

If you've enabled Insights events on a trail and CloudTrail detects unusual activity, CloudTrail delivers Insights events to the `/CloudTrail-Insight` folder in the chosen destination S3 bucket for your trail. After you enable CloudTrail Insights for the first time on a trail, it can take up to 36 hours for CloudTrail to deliver the first Insights event, if unusual activity is detected.

If you turn off Insights events logging on a trail and then re-enable Insights events, or stop and restart logging on a trail, it can take up to 36 hours for CloudTrail to restart delivery of Insights events, if unusual activity is detected.

Insights events delivery for event data stores

If you've enabled Insights events on a source event data store, CloudTrail delivers Insights events to the destination event data store. After you enable CloudTrail Insights for the first time on the source event data store, it can take up to 7 days for CloudTrail to deliver the first Insights event to the destination event data store, if unusual activity is detected.

If you turn off Insights events logging on a source event data store and then re-enable Insights events, or stop and restart event ingestion on a source event data store, it can take up to 7 days for CloudTrail to restart delivery of Insights events, if unusual activity is detected. Additional charges apply for ingesting Insights events in CloudTrail Lake. You will be charged separately if you enable Insights for both trails and event data stores. For information about CloudTrail pricing, see AWS CloudTrail Pricing.

Logging Insights events with the AWS Management Console

You can enable Insights events on a trail or event data store using the console.

Topics

Enabling CloudTrail Insights events on an existing trail

Use the following procedure to enable CloudTrail Insights events on an existing trail. By default, Insights events are not enabled.

1. In the left navigation pane of the CloudTrail console, open the **Trails** page, and choose a trail name.
2. In **Insights events** choose **Edit**.

> **Note**
>
> Additional charges apply for logging Insights events. For CloudTrail pricing, see AWS CloudTrail Pricing.

3. In **Event type**, choose **Insights events**.
4. In **Insights events**, under **Choose Insights types**, choose **API call rate**, **API error rate**, or both. Your trail must be logging **Write** management events to log Insights events for **API call rate**.

Your trail must be logging **Read** or **Write** management events to log Insights events for **API error rate**.

5. Choose **Save changes** to save your changes.

It can take up to 36 hours for CloudTrail to deliver the first Insights events, if unusual activity is detected.

Enabling CloudTrail Insights events on an existing event data store

Use the following procedure to enable CloudTrail Insights events on an existing event data store. By default, Insights events are not enabled.

Additional charges apply for ingesting Insights events in CloudTrail Lake. You will be charged separately if you enable Insights for both trails and event data stores. For information about CloudTrail pricing, see AWS CloudTrail Pricing.

> **Note**
>
> You can only enable CloudTrail Insights events on event data stores containing CloudTrail management events. You cannot enable CloudTrail Insights events on other event data store types.

1. In the left navigation pane of the CloudTrail console, under **Lake**, choose **Event data stores**.
2. Choose the event data store name.
3. In **Management events**, choose **Edit**.
4. Choose **Enable Insights**.
5. Choose the destination event data store where CloudTrail will deliver Insights events. The destination event data store will collect Insights events based upon the management event activity in this event data store. For information about how to create the destination event data store, see To create a destination event data store that logs Insights events.
6. Under **Choose Insights types**, choose **API call rate**, **API error rate**, or both. Your event data store must be logging **Write** management events to log Insights events for **API call rate**. Your event data store must be logging **Read** or **Write** management events to log Insights events for **API error rate**.
7. Choose **Save changes** to save your changes.

It can take up to 7 days for CloudTrail to deliver the first Insights events, if unusual activity is detected.

Logging Insights events with the AWS Command Line Interface

You can configure your trails and event data stores to log Insights events using the AWS CLI.

> **Note**
>
> To log Insights events on API call volume, the trail or event data store must log `write` management events. To log Insights events on API error rate, the trail or event data store must log `read` or `write` management events.

Topics

- Logging Insights events for a trail using the AWS CLI
- Logging Insights events for an event data store using the AWS CLI

Logging Insights events for a trail using the AWS CLI

To view whether your trail is logging Insights events, run the `get-insight-selectors` command.

```
aws cloudtrail get-insight-selectors --trail-name TrailName
```

The following result shows the default settings for a trail. By default, trails don't log Insights events. The `InsightType` attribute value is empty, and no Insight event selectors are specified, because Insights event collection is not enabled.

If you do not add Insights selectors, the **get-insight-selectors** command returns the following error message: "An error occurred (InsightNotEnabledException) when calling the GetInsightSelectors operation: Trail *name* does not have Insights enabled. Edit the trail settings to enable Insights, and then try the operation again."

```
{
  "InsightSelectors": [ ],
  "TrailARN": "arn:aws:cloudtrail:us-east-1:123456789012:trail/TrailName"
}
```

To configure your trail to log Insights events, run the `put-insight-selectors` command. The following example shows how to configure your trail to include Insights events. Insights selector values can be `ApiCallRateInsight`, `ApiErrorRateInsight`, or both.

```
aws cloudtrail put-insight-selectors --trail-name TrailName --insight-selectors
 '[{"InsightType": "ApiCallRateInsight"},{"InsightType": "ApiErrorRateInsight"}]'
```

The following result shows the Insights event selector that is configured for the trail.

```
{
  "InsightSelectors":
      [
         {
            "InsightType": "ApiErrorRateInsight"
         },
         {
            "InsightType": "ApiCallRateInsight"
         }
      ],
  "TrailARN": "arn:aws:cloudtrail:us-east-1:123456789012:trail/TrailName"
}
```

Logging Insights events for an event data store using the AWS CLI

To enable Insights on an event data store, you must have a source event data store that logs management events and a destination event data store that logs Insights events.

To view whether Insights events are enabled on an event data store, run the **get-insight-selectors** command.

```
aws cloudtrail get-insight-selectors --event-data-store arn:aws:cloudtrail:us-
east-1:123456789012:eventdatastore/EXAMPLE-f852-4e8f-8bd1-bcf6cEXAMPLE
```

To view whether an event data store is configured to receive Insights events or management events, run the **get-event-data-store** command.

```
aws cloudtrail get-event-data-store --event-data-store arn:aws:cloudtrail:us-
east-1:123456789012:eventdatastore/EXAMPLE-d483-5c7d-4ac2-adb5dEXAMPLE
```

The following procedure shows you how to create the destination and source event data stores and then enable Insights events.

1. Run the **aws cloudtrail create-event-data-store** command to create a destination event data store that collects Insights events. The value for `eventCategory` must be `Insight`. Replace *`retention-period-days`* with the number of days you would like to retain events in your event data store.

   If you are signed in with the management account for an AWS Organizations organization, include the `--organization-enabled` parameter if you want to give your delegated administrator access to the event data store.

```
aws cloudtrail create-event-data-store \
--name insights-event-data-store \
--no-multi-region-enabled \
--retention-period retention-period-days \
--advanced-event-selectors '[
    {
      "Name": "Select Insights events",
      "FieldSelectors": [
          { "Field": "eventCategory", "Equals": ["Insight"] }
        ]
    }
  ]'
```

   The following is an example response.

```
{
    "Name": "insights-event-data-store",
    "ARN": "arn:aws:cloudtrail:us-east-1:111122223333:eventdatastore/
EXAMPLEf852-4e8f-8bd1-bcf6cEXAMPLE",
    "AdvancedEventSelectors": [
        {
           "Name": "Select Insights events",
           "FieldSelectors": [
              {
                  "Field": "eventCategory",
                  "Equals": [
                      "Insight"
                    ]
                }
            ]
        }
    ],
    "MultiRegionEnabled": false,
    "OrganizationEnabled": false,
    "BillingMode": "EXTENDABLE_RETENTION_PRICING",
    "RetentionPeriod": "90",
    "TerminationProtectionEnabled": true,
    "CreatedTimestamp": "2023-11-08T15:22:33.578000+00:00",
    "UpdatedTimestamp": "2023-11-08T15:22:33.714000+00:00"
}
```

You will use the ARN (or ID suffix of the ARN) from the response as the value for the `--insights-destination` parameter in step 3.

2. Run the **aws cloudtrail create-event-data-store** command to create a source event data store that logs management events. By default, event data stores log all management events. You don't need to specify the advanced event selectors if you want to log all management events. Replace *`retention-period-days`* with the number of days you would like to retain events in your event data store. If you are creating an organization event data store, include the `--organization-enabled` parameter.

```
aws cloudtrail create-event-data-store --name source-event-data-store --retention-
period retention-period-days
```

The following is an example response.

```
{
    "EventDataStoreArn": "arn:aws:cloudtrail:us-east-1:111122223333:eventdatastore/
EXAMPLE9952-4ab9-49c0-b788-f4f3EXAMPLE",
    "Name": "source-event-data-store",
    "Status": "CREATED",
    "AdvancedEventSelectors": [
        {
            "Name": "Default management events",
            "FieldSelectors": [
                {
                    "Field": "eventCategory",
                    "Equals": [
                        "Management"
                    ]
                }
            ]
        }
    ],
    "MultiRegionEnabled": true,
    "OrganizationEnabled": false,
    "BillingMode": "EXTENDABLE_RETENTION_PRICING",
    "RetentionPeriod": 90,
    "TerminationProtectionEnabled": true,
    "CreatedTimestamp": "2023-11-08T15:25:35.578000+00:00",
    "UpdatedTimestamp": "2023-11-08T15:25:35.714000+00:00"
}
```

You will use the ARN (or ID suffix of the ARN) from the response as the value for the `--event-data-store` parameter in step 3.

3. Run the **put-insight-selectors** command to enable Insights events. Insights selector values can be `ApiCallRateInsight`, `ApiErrorRateInsight`, or both. For the `--event-data-store` parameter, specify the ARN (or ID suffix of the ARN) of the source event data store that logs management events and will enable Insights. For the `--insights-destination` parameter, specify the ARN (or ID suffix of the ARN) of the destination event data store that will log Insights events.

```
aws cloudtrail put-insight-selectors --event-data-store arn:aws:cloudtrail:us-
east-1:111122223333:eventdatastore/EXAMPLE9952-4ab9-49c0-b788-f4f3EXAMPLE --
insights-destination arn:aws:cloudtrail:us-east-1:111122223333:eventdatastore/
EXAMPLEf852-4e8f-8bd1-bcf6cEXAMPLE --insight-selectors '[{"InsightType":
 "ApiCallRateInsight"},{"InsightType": "ApiErrorRateInsight"}]'
```

The following result shows the Insights event selector that is configured for the event data store.

```
{
  "EventDataStoreARN": "arn:aws:cloudtrail:us-east-1:111122223333:eventdatastore/
EXAMPLE9952-4ab9-49c0-b788-f4f3EXAMPLE",
  "InsightsDestination": "arn:aws:cloudtrail:us-east-1:111122223333:eventdatastore/
EXAMPLEf852-4e8f-8bd1-bcf6cEXAMPLE",
  "InsightSelectors":
      [
         {
            "InsightType": "ApiErrorRateInsight"
         },
         {
            "InsightType": "ApiCallRateInsight"
         }
```

```
        ]
}
```

After you enable CloudTrail Insights for the first time on an event data store, it can take up to 7 days for CloudTrail to deliver the first Insights event, if unusual activity is detected.

CloudTrail Insights analyzes management events that occur in a single Region, not globally. A CloudTrail Insights event is generated in the same Region as its supporting management events are generated.

For an organization event data store, CloudTrail analyzes management events from each member's account instead of analyzing the aggregation of all management events for the organization.

Additional charges apply for ingesting Insights events in CloudTrail Lake. You will be charged separately if you enable Insights for both trails and event data stores. For information about CloudTrail pricing, see AWS CloudTrail Pricing.

Logging Insights events with the AWS SDKs

Run the GetInsightSelectors operation to see whether your trail or event data store enables Insights events. You can configure your trails or event data stores to enable Insights events with the PutInsightSelectors operation. For more information, see the AWS CloudTrail API Reference.

CloudTrail record contents

The body of the record contains fields that help you determine the requested action as well as when and where the request was made. When the value of **Optional** is **True**, the field is only present when it applies to the service, API, or event type. An **Optional** value of **False** means that the field is either always present, or that its presence does not depend on the service, API, or event type. An example is `responseElements`, which is present in events for actions that make changes (create, update, or delete actions).

CloudTrail truncates a field if the field's contents exceeds the maximum field size. If a field is truncated, `omitted` is present with a value of `true`.

`eventTime`

The date and time the request was completed, in coordinated universal time (UTC). An event's time stamp comes from the local host that provides the service API endpoint on which the API call was made. For example, a **CreateBucket** API event that is run in the US West (Oregon) Region would get its time stamp from the time on an AWS host running the Amazon S3 endpoint, `s3.us-west-2.amazonaws.com`. In general, AWS services use Network Time Protocol (NTP) to synchronize their system clocks.

Since: 1.0

Optional: False

`eventVersion`

The version of the log event format. The current version is 1.10.

The `eventVersion` value is a major and minor version in the form *`major_version`*.*`minor_version`*. For example, you can have an `eventVersion` value of `1.09`, where `1` is the major version, and `09` is the minor version.

CloudTrail increments the major version if a change is made to the event structure that is not backward-compatible. This includes removing a JSON field that already exists, or changing how the contents of a field are represented (for example, a date format). CloudTrail increments the minor version if a change adds new fields to the event structure. This can occur if new information is available for some or all existing events, or if new information is available only for new event types. Applications can ignore new fields to stay compatible with new minor versions of the event structure.

If CloudTrail introduces new event types, but the structure of the event is otherwise unchanged, the event version does not change.

To be sure that your applications can parse the event structure, we recommend that you perform an equal-to comparison on the major version number. To be sure that fields that are expected by your application exist, we also recommend performing a greater-than-or-equal-to comparison on the minor version. There are no leading zeroes in the minor version. You can interpret both *`major_version`* and *`minor_version`* as numbers, and perform comparison operations.

Since: 1.0

Optional: False

`userIdentity`

Information about the IAM identity that made a request. For more information, see [CloudTrail userIdentity element]().

Since: 1.0

Optional: False

`eventSource`

The service that the request was made to. This name is typically a short form of the service name without spaces plus `.amazonaws.com`. For example:

- AWS CloudFormation is `cloudformation.amazonaws.com`.
- Amazon EC2 is `ec2.amazonaws.com`.
- Amazon Simple Workflow Service is `swf.amazonaws.com`.

This convention has some exceptions. For example, the `eventSource` for Amazon CloudWatch is `monitoring.amazonaws.com`.

Since: 1.0

Optional: False

`eventName`

The requested action, which is one of the actions in the API for that service.

Since: 1.0

Optional: False

`awsRegion`

The AWS Region that the request was made to, such as `us-east-2`. See [CloudTrail supported Regions]().

Since: 1.0

Optional: False

sourceIPAddress

The IP address that the request was made from. For actions that originate from the service console, the address reported is for the underlying customer resource, not the console web server. For services in AWS, only the DNS name is displayed.

> **Note**
>
> For events originated by AWS, this field is usually `AWS Internal/#`, where `#` is a number used for internal purposes.

Since: 1.0

Optional: False

userAgent

The agent through which the request was made, such as the AWS Management Console, an AWS service, the AWS SDKs or the AWS CLI. This field has a maximum size of 1 KB; content exceeding that limit is truncated. The following are example values:

- `lambda.amazonaws.com` – The request was made with AWS Lambda.
- `aws-sdk-java` – The request was made with the AWS SDK for Java.
- `aws-sdk-ruby` – The request was made with the AWS SDK for Ruby.
- `aws-cli/1.3.23 Python/2.7.6 Linux/2.6.18-164.el5` – The request was made with the AWS CLI installed on Linux.

> **Note**
>
> For events originated by AWS, if CloudTrail knows which AWS service made the call, this field is the event source of the calling service (for example, `ec2.amazonaws.com`). Otherwise, this field is `AWS Internal/#`, where `#` is a number used for internal purposes.

Since: 1.0

Optional: True

`errorCode`

The AWS service error if the request returns an error. For an example that shows this field, see Error code and message log example. This field has a maximum size of 1 KB; content exceeding that limit is truncated.

Since: 1.0

Optional: True

`errorMessage`

If the request returns an error, the description of the error. This message includes messages for authorization failures. CloudTrail captures the message logged by the service in its exception handling. For an example, see Error code and message log example. This field has a maximum size of 1 KB; content exceeding that limit is truncated.

> **Note**
>
> Some AWS services provide the `errorCode` and `errorMessage` as top-level fields in the event. Other AWS services provide error information as part of `responseElements`.

Since: 1.0

Optional: True

`requestParameters`

The parameters, if any, that were sent with the request. These parameters are documented in the API reference documentation for the appropriate AWS service. This field has a maximum size of 100 KB; content exceeding that limit is truncated.

Since: 1.0

Optional: False

`responseElements`

The response elements, if any, for actions that make changes (create, update, or delete actions). If the action
doesn't return response elements, this field is `null`. If
an action does not change state (for example, a request to get or list objects),
this element is omitted. The response elements for actions are documented in the API reference
documentation for the appropriate AWS service. This field has a maximum size
of 100 KB; content exceeding that limit is truncated.

The `responseElements` value is useful to help you trace a request
with AWS Support. Both `x-amz-request-id` and `x-amz-id-2`
contain information that helps you trace a request with AWS Support. These values are
the same as those that the service returns in the response to the request that
initiates the events, so you can use them to match the event to the
request.

Since: 1.0

Optional: False

`additionalEventData`

Additional data about the event that was not part of the request or response. This field has a maximum size of 28 KB; content exceeding that limit is truncated.

Since: 1.0

Optional: True

`requestID`

The value that identifies the request. The service being called generates this value. This field has a maximum size of 1 KB; content exceeding that limit is truncated.

Since: 1.01

Optional: True

`eventID`

GUID generated by CloudTrail to uniquely identify each event. You can use this value to identify a single event. For example, you can use the ID as a primary key to retrieve log data from a searchable database.

Since: 1.01

Optional: False

`eventType`

Identifies the type of event that generated the event record. This can be the one of the following values:

- `AwsApiCall` – An API was called.
- `AwsServiceEvent` – The service generated an event related to your trail. For example, this can occur when another account made a call with a resource that you own.
- `AwsConsoleAction` – An action was taken in the console that was not an API call.
- `AwsConsoleSignIn` – A user in your account (root, IAM, federated, SAML, or SwitchRole) signed in to the AWS Management Console.
- `AwsCloudTrailInsight` – If Insights events are enabled, CloudTrail generates Insights events when CloudTrail detects unusual operational activity such as spikes in resource provisioning or bursts of AWS Identity and Access Management (IAM) actions.

  `AwsCloudTrailInsight` events do *not* use the following fields:

  - `eventName`
  - `eventSource`
  - `sourceIPAddress`
  - `userAgent`
  - `userIdentity`

Since: 1.02

Optional: False

`apiVersion`

Identifies the API version associated with the `AwsApiCall` `eventType` value.

Since: 1.01

Optional: True

`managementEvent`

A Boolean value that identifies whether the event is a management event. `managementEvent` is shown in an event record if `eventVersion` is 1.06 or higher, and the event type is one of the following:

- `AwsApiCall`
- `AwsConsoleAction`
- `AwsConsoleSignIn`
- `AwsServiceEvent`

Since: 1.06

Optional: True

`readOnly`

Identifies whether this operation is a read-only operation. This can be one of the following values:

- `true` – The operation is read-only (for example, `DescribeTrails`).
- `false` – The operation is write-only (for example, `DeleteTrail`).

Since: 1.01

Optional: True

`resources`

A list of resources accessed in the event. The field can contain the following information:

- Resource ARNs

- Account ID of the resource owner
- Resource type identifier in the format: `AWS::`*`aws-service-name`*`::`*`data-type-name`*

For example, when an `AssumeRole` event is logged, the `resources` field can appear like the following:

- ARN: `arn:aws:iam::123456789012:role/`*`myRole`*
- Account ID: `123456789012`
- Resource type identifier: `AWS::IAM::Role`

For example logs with the `resources` field, see AWS STS API Event in CloudTrail Log File in the *IAM User Guide* or Logging AWS KMS API Calls in the *AWS Key Management Service Developer Guide*.

Since: 1.01

Optional: True

`recipientAccountId`

Represents the account ID that received this event. The `recipientAccountID` may be different from the CloudTrail userIdentity element `accountId`. This can occur in cross-account resource access. For example, if a KMS key, also known as an AWS KMS key, was used by a separate account to call the Encrypt API, the `accountId` and `recipientAccountID` values will be the same for the event delivered to the account that made the call, but the values will be different for the event that is delivered to the account that owns the KMS key.

Since: 1.02

Optional: True

`serviceEventDetails`

Identifies the service event, including what triggered the event and the result. For more information, see AWS service events. This field has a maximum size of 100 KB; content exceeding that limit is truncated.

Since: 1.05

Optional: True

`sharedEventID`

GUID generated by CloudTrail to uniquely identify CloudTrail events from the same AWS action that is sent to different AWS accounts.

For example, when an account uses an AWS KMS key that belongs to another account, the account that used the KMS key and the account that owns the KMS key receive separate CloudTrail events for the same action. Each CloudTrail event delivered for this AWS action shares the same `sharedEventID`, but also has a unique `eventID` and `recipientAccountID`.

For more information, see Example sharedEventID.

> **Note**
>
> The `sharedEventID` field is present only when CloudTrail events are delivered to multiple accounts. If the caller and owner are the same AWS account, CloudTrail sends only one event, and the `sharedEventID` field is not present.

Since: 1.03

Optional: True

`vpcEndpointId`

Identifies the VPC endpoint in which requests were made from a VPC to another AWS service, such as Amazon S3.

Since: 1.04

Optional: True

`vpcEndpointAccountId`

Identifies the AWS account ID of the VPC endpoint owner for the corresponding endpoint for which a request has traversed.

Since: 1.09

Optional: True

`eventCategory`

Shows the event category. The `eventCategory` is used in [`LookupEvents`]() calls for management and Insights events.

- For management events, the value is `Management`.
- For data events, the value is `Data`.
- For Insights events, the value is `Insight`.

Since: 1.07

Optional: False

`addendum`

If an event delivery was delayed, or additional information about an existing event becomes available after the event is logged, an addendum field shows information about why the event was delayed. If information was missing from an existing event, the addendum field includes the missing information and a reason for why it was missing. Contents include the following.

- **`reason`** - The reason that the event or some of its contents were missing. Values can be any of the following.
  - **`DELIVERY_DELAY`** – There was a delay delivering events. This could be caused by high network traffic, connectivity issues, or a CloudTrail service issue.
  - **`UPDATED_DATA`** – A field in the event record was missing or had an incorrect value.
  - **`SERVICE_OUTAGE`** – A service that logs events to CloudTrail had an outage, and couldn't log events to CloudTrail. This is exceptionally rare.
- **`updatedFields`** - The event record fields that are updated by the addendum. This is only provided if the reason is `UPDATED_DATA`.
- **`originalRequestID`** - The original unique ID of the request. This is only provided if the reason is `UPDATED_DATA`.
- **`originalEventID`** - The original event ID. This is only provided if the reason is `UPDATED_DATA`.

Since: 1.08

Optional: True

`sessionCredentialFromConsole`

Shows whether or not an event originated from an AWS Management Console session. This field is not shown unless the value is `true`, meaning that the client that was used to make the API call was either a proxy or an external client. If a proxy client was used, the`tlsDetails` event field is not shown.

Since: 1.08

Optional: True

`edgeDeviceDetails`

Shows information about edge devices that are targets of a request. Currently, [`S3 Outposts`] device events include this field. This field has a maximum size of 28 KB; content exceeding that limit is truncated.

Since: 1.08

Optional: True

`tlsDetails`

Shows information about the Transport Layer Security (TLS) version, cipher suites, and the fully qualified domain name (FQDN) of the client-provided host name used in the service API call, which is typically the FQDN of the service endpoint. CloudTrail still logs partial TLS details if expected information is missing or empty. For example, if the TLS version and cipher suite are present, but the `HOST` header is empty, available TLS details are still logged in the CloudTrail event.

- **`tlsVersion`** - The TLS version of a request.
- **`cipherSuite`** - The cipher suite (combination of security algorithms used) of a request.
- **`clientProvidedHostHeader`** - The client-provided host name used in the service API call, which is typically the FQDN of the service endpoint.

> **Note**
>
> There are some cases when the `tlsDetails` field is not present in an event record.
>
> - The `tlsDetails` field is not present if the API call was made by an AWS service on your behalf. The `invokedBy` field in the `userIdentity` element identifies the AWS service that made the API call.

- If `sessionCredentialFromConsole` is present with a value of true, `tlsDetails` is present in an event record only if an external client was used to make the API call.

Since: 1.08

Optional: True

Record fields for Insights events

The following are attributes shown in the JSON structure of an Insights event that differ from those in a management or data event.

`sharedEventId`

A `sharedEventID` for CloudTrail Insights events differs from the `sharedEventID` for the management and data types of CloudTrail events. In Insights events, a `sharedEventID` is a GUID that is generated by CloudTrail Insights to uniquely identify an Insights event. `sharedEventID` is common between the start and the end Insights events, and helps to connect both events to uniquely identify unusual activity. You can think of the `sharedEventID` as the overall Insights event ID.

Since: 1.07

Optional: False

`insightDetails`

Insights events only. Shows information about the underlying triggers of an Insights event, such as event source, user agent, statistics, API name, and whether the event is the start or end of the Insights event. For more information about the contents of the `insightDetails` block, see CloudTrail Insights `insightDetails` element.

Since: 1.07

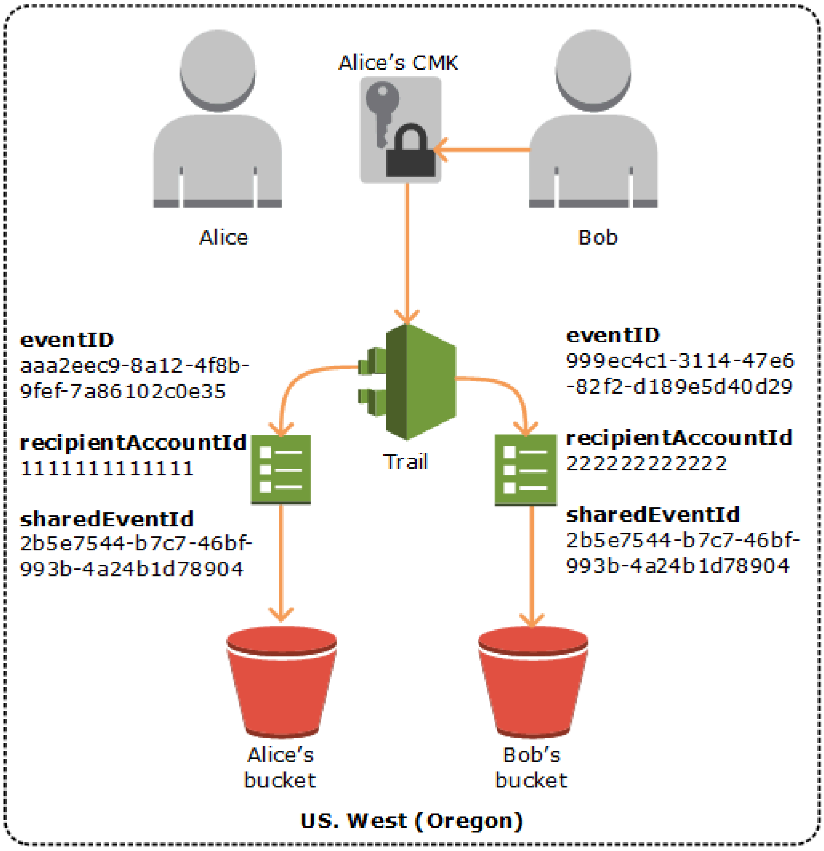
Optional: False

Example sharedEventID

The following is an example that describes how CloudTrail delivers two events for the same action:

1. Alice has AWS account (111111111111) and creates an AWS KMS key. She is the owner of this KMS key.
2. Bob has AWS account (222222222222). Alice gives Bob permission to use the KMS key.
3. Each account has a trail and a separate bucket.
4. Bob uses the KMS key to call the `Encrypt` API.
5. CloudTrail sends two separate events.

    - One event is sent to Bob. The event shows that he used the KMS key.
    - One event is sent to Alice. The event shows that Bob used the KMS key.
    - The events have the same `sharedEventID`, but the `eventID` and `recipientAccountID` are unique.



Shared event IDs in CloudTrail Insights

A `sharedEventID` for CloudTrail Insights events differs from the `sharedEventID` for the management and data types of CloudTrail events. In Insights events, a `sharedEventID` is a GUID that is generated by CloudTrail Insights to uniquely identify a start and end pair of Insights events. `sharedEventID` is common between the start and the end Insights event, and helps to create a correlation between both events to uniquely identify unusual activity.

You can think of the `sharedEventID` as the overall Insights event ID.

CloudTrail userIdentity element

AWS Identity and Access Management (IAM) provides different types of identities. The `userIdentity` element contains details about the type of IAM identity that made the request, and which credentials were used. If temporary credentials were used, the element shows how the credentials were obtained.

Contents

Examples

`userIdentity` with IAM user credentials

The following example shows the `userIdentity` element of a simple request made with the credentials of the IAM user named `Alice`.

```
"userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDAJ45Q7YFFAREXAMPLE",
    "arn": "arn:aws:iam::123456789012:user/Alice",
    "accountId": "123456789012",
    "accessKeyId": "",
    "userName": "Alice"
}
```

`userIdentity` with temporary security credentials

The following example shows a `userIdentity` element for a request made with temporary security credentials obtained by assuming an IAM role. The element contains additional details about the role that was assumed to get credentials.

```
"userIdentity": {
    "type": "AssumedRole",
    "principalId": "AROAIDPPEZS35WEXAMPLE:AssumedRoleSessionName",
    "arn": "arn:aws:sts::123456789012:assumed-role/RoleToBeAssumed/MySessionName",
    "accountId": "123456789012",
    "accessKeyId": "",
    "sessionContext": {
        "sessionIssuer": {
            "type": "Role",
            "principalId": "AROAIDPPEZS35WEXAMPLE",
            "arn": "arn:aws:iam::123456789012:role/RoleToBeAssumed",
            "accountId": "123456789012",
            "userName": "RoleToBeAssumed"
        },
        "attributes": {
            "mfaAuthenticated": "false",
            "creationDate": "20131102T010628Z"
        )

    }
}
```

`userIdentity` for a request made on behalf of an IAM Identity Center user

The following example shows a `userIdentity` element for a request made on behalf of an IAM Identity Center user.

```
"userIdentity": {
    "type": "IdentityCenterUser",
    "accountId": "123456789012",
    "onBehalfOf": {
        "userId": "544894e8-80c1-707f-60e3-3ba6510dfac1",
        "identityStoreArn": "arn:aws:identitystore::123456789012:identitystore/
d-9067642ac7"
    },
    "credentialId": "EXAMPLEVHULjJdTUdPJfofVa1sufHDoj7aYcOYcxFVllWR_Whr1fEXAMPLE"
}
```

Fields

The following fields can appear in a userIdentity element.

type

The type of the identity. The following values are possible:

- Root – The request was made with your AWS account credentials. If the userIdentity type is Root, and you set an alias for your account, the userName field contains your account alias. For more information, see Your AWS account ID and its alias.
- IAMUser – The request was made with the credentials of an IAM user.
- AssumedRole – The request was made with temporary security credentials that were obtained with a role by making a call to the AWS Security Token Service (AWS STS) AssumeRole API. This can include roles for Amazon EC2 and cross-account API access.
- Role – The request was made with a persistent IAM identity that has specific permissions. The issuer of the role sessions is always the role. For more information about roles, see Roles terms and concepts in the *IAM User Guide*.
- FederatedUser – The request was made with temporary security credentials obtained from a call to the AWS STS GetFederationToken API. The sessionIssuer element indicates if the API was called with root or IAM user credentials.

  For more information about temporary security credentials, see Temporary Security Credentials in the *IAM User Guide*.
- Directory – The request was made to a directory service, and the type is unknown. Directory services include the following: Amazon WorkDocs and Amazon QuickSight.
- AWSAccount – The request was made by another AWS account
- AWSService – The request was made by an AWS account that belongs to an AWS service. For example, AWS Elastic Beanstalk assumes an IAM role in your account to call other AWS services on your behalf.
- IdentityCenterUser – The request was made on behalf of an IAM Identity Center user.
- Unknown – The request was made with an identity type that CloudTrail can't determine.

Optional: False

AWSAccount and AWSService appear for type in your logs when there is cross-account access using an IAM role that you own.

Example: Cross-account access initiated by another AWS account

1. You own an IAM role in your account.
2. Another AWS account switches to that role to assume the role for your account.
3. Because you own the IAM role, you receive a log that shows the other account assumed the role. The type is AWSAccount. For an example log entry, see AWS STS API event in CloudTrail log file.

Example: Cross-account access initiated by an AWS service

1. You own an IAM role in your account.
2. An AWS account owned by an AWS service assumes that role.
3. Because you own the IAM role, you receive a log that shows the AWS service assumed the role. The type is AWSService.

userName

The friendly name of the identity that made the call. The value that appears in userName is based on the value in type. The following table shows the relationship between type and userName:

type	userName	Description
Root (no alias set)	Not present	If you haven't set up an alias for your AWS account, the userName field doesn't appear. For more information about account aliases, see Your AWS account ID and its alias. Note that the userName field can't contain Root, because Root is an identity type and not a user name.
Root (alias set)	The account alias	For more information about AWS account aliases, see Your AWS account ID and its alias.

type	userName	Description
IAMUser	The user name of the IAM user	
AssumedRole	Not present	For the AssumedRole type, you can find the userName field in sessionContext as part of the sessionIssuer element. For an example entry, see Examples.
Role	User-defined	The sessionContext and sessionIssuer section contains information about the identity that issued the session for the role.
FederatedUser	Not present	The sessionContext and sessionIssuer section contains information about the identity that issued the session for the federated user.
Directory	Can be present	For example, the value can be the account alias or email address of the associated AWS account ID.
AWSService	Not present	
AWSAccount	Not present	
IdentityCenterUser	Not present	The onBehalfOf section contains information about the IAM Identity Center user ID and identity store ARN for which the call was made. For more information about IAM Identity Center, see the *AWS IAM Identity Center User Guide*.
Unknown	Can be present	For example, the value can be the account alias or email address of the associated AWS account ID.

> **Note**
>
> The `userName` field contains the string `HIDDEN_DUE_TO_SECURITY_REASONS` when the recorded event is a console sign-in failure caused by incorrect user name input. CloudTrail does not record the contents in this case because the text could contain sensitive information, as in the following examples:
>
> - A user accidentally types a password in the user name field.
> - A user clicks the link for one AWS account's sign-in page, but then types the account number for a different one.
> - A user accidentally types the account name of a personal email account, a bank sign-in identifier, or some other private ID.

Optional: True

`principalId`

A unique identifier for the entity that made the call. For requests made with temporary security credentials, this value includes the session name that is passed to the `AssumeRole`, `AssumeRoleWithWebIdentity`, or `GetFederationToken` API call.

Optional: True

`arn`

The Amazon Resource Name (ARN) of the principal that made the call. The last section of the arn contains the user or role that made the call.

Optional: True

`accountId`

The account that owns the entity that granted permissions for the request. If the request was made with temporary security credentials, this is the account that owns the IAM user or role used to obtain credentials.

If the request was made with an IAM Identity Center authorized access token, this is the account that owns the IAM Identity Center instance.

Optional: True

accessKeyId

The access key ID that was used to sign the request. If the request was made with temporary security credentials, this is the access key ID of the temporary credentials. For security reasons, `accessKeyId` might not be present, or might be displayed as an empty string.

Optional: True

sessionContext

If the request was made with temporary security credentials, `sessionContext` provides information about the session created for those credentials. You create a session when you call any API that returns temporary credentials. Users also create sessions when they work in the console and make requests with APIs that include multi-factor authentication. The following attributes can appear in `sessionContext`:

- `sessionIssuer` – If a user make a request with temporary security credentials, `sessionIssuer` provides information about how the user obtained credentials. For example, if the they obtained temporary security credentials by assuming a role, this element provides information about the assumed role. If they obtained credentials with root or IAM user credentials to call AWS STS `GetFederationToken`, the element provides information about the root account or IAM user. This element has the following attributes:
    - `type` – The source of the temporary security credentials, such as `Root`, `IAMUser`, or `Role`.
    - `userName` – The friendly name of the user or role that issued the session. The value that appears depends on the `sessionIssuer` identity `type`. The following table shows the relationship between `sessionIssuer type` and `userName`:

sessionIssuer type	**userName**	**Description**
Root (no alias set)	Not present	If you have not set up an alias for your account, the `userName` field does not appear. For more information about AWS account aliases, see Your AWS account ID and its alias. Note that the `userName` field can't contain `Root`, because `Root` is an identity type, not a user name.

sessionIssuer type	userName	Description
Root (alias set)	The account alias	For more information about AWS account aliases, see Your AWS account ID and its alias.
IAMUser	The user name of the IAM user	This also applies when a federated user is using a session issued by IAMUser.
Role	The role name	A role assumed by an IAM user, AWS service, or web identity federated user in a role session.

  - principalId – The internal ID of the entity used to get credentials.
  - arn – The ARN of the source (account, IAM user, or role) that was used to get temporary security credentials.
  - accountId – The account that owns the entity that was used to get credentials.

- webIdFederationData – If the request was made with temporary security credentials obtained by web identity federation, webIdFederationData lists information about the identity provider.

  This element has the following attributes:

  - federatedProvider – The principal name of the identity provider (for example, www.amazon.com for Login with Amazon or accounts.google.com for Google).
  - attributes – The application ID and user ID as reported by the provider (for example, www.amazon.com:app_id and www.amazon.com:user_id for Login with Amazon).

  Note

  The omission of this field or presence of this field with an empty value signifies that there is no information about the identity provider.

- creationDate – The date and time when the temporary security credentials were issued. Represented in ISO 8601 basic notation.
- mfaAuthenticated – The value is true if the root user or IAM user who used their credentials for the request also authenticated with an MFA device; otherwise, false.

- `sourceIdentity` – See AWS STS source identity in this topic. The `sourceIdentity` field occurs in events when users assume an IAM role to perform an action. `sourceIdentity` identifies the original user identity making the request, whether that user's identity is an IAM user, an IAM role, a user authenticated through SAML-based federation, or a user authenticated through OpenID Connect (OIDC)-compliant web identity federation. For more information about configuring AWS STS to collect source identity information, see Monitor and control actions taken with assumed roles in the *IAM User Guide*.
- `ec2RoleDelivery` – The value is `1.0` if the credentials were provided by Amazon EC2 Instance Metadata Service Version 1 (IMDSv1). The value is `2.0` if the credentials were provided using the new IMDS scheme.

  AWS credentials provided by the Amazon EC2 Instance Metadata Service (IMDS) include an ec2:RoleDelivery IAM context key. This context key makes it easy to enforce use of the new scheme on a service-by-service or resource-by-resource basis by using the context key as a condition in IAM policies, resource policies, or AWS Organizations service control policies. For more information, see Instance metadata and user data in the *Amazon EC2 User Guide*.

Optional: True

`invokedBy`

The name of the AWS service that made the request, when a request is made by an AWS service such as Amazon EC2 Auto Scaling or AWS Elastic Beanstalk. This field is only present when a request is made by an AWS service. This includes requests made by services using forward access sessions (FAS), AWS service principals, service-linked roles, or service roles used by an AWS service.

Optional: True

`onBehalfOf`

If the request was made by an IAM Identity Center caller, `onBehalfOf` provides information about the IAM Identity Center user ID and identity store ARN for which the call was made. This element has the following attributes:

- `userId` – The ID of the IAM Identity Center user who the call was made on behalf of.
- `identityStoreArn` – The ARN of the IAM Identity Center identity store that the call was made on behalf of.

Optional: True

`credentialId`

The credential ID for the request. This is only set when the caller uses a bearer token, such as an IAM Identity Center authorized access token.

Optional: True

Values for AWS STS APIs with SAML and web identity federation

AWS CloudTrail supports logging AWS Security Token Service (AWS STS) API calls made with Security Assertion Markup Language (SAML) and web identity federation. When a user makes a call to the `AssumeRoleWithSAML` and `AssumeRoleWithWebIdentity` APIs, CloudTrail records the call and delivers the event to your Amazon S3 bucket.

The `userIdentity` element for these APIs contains the following values.

`type`

The identity type.

- `SAMLUser` – The request was made with SAML assertion.
- `WebIdentityUser` – The request was made by a web identity federation provider.

`principalId`

A unique identifier for the entity that made the call.

- For `SAMLUser`, this is a combination of the `saml:namequalifier` and `saml:sub` keys.
- For `WebIdentityUser`, this is a combination of the issuer, application ID, and user ID.

`userName`

The name of the identity that made the call.

- For `SAMLUser`, this is the `saml:sub` key.
- For `WebIdentityUser`, this is the user ID.

`identityProvider`

The principal name of the external identity provider. This field appears only for `SAMLUser` or `WebIdentityUser` types.

- For SAMLUser, this is the saml:namequalifier key for the SAML assertion.
- For WebIdentityUser, this is the issuer name of the web identity federation provider. This can be a provider that you configured, such as the following:
  - cognito-identity.amazon.com for Amazon Cognito
  - www.amazon.com for Login with Amazon
  - accounts.google.com for Google
  - graph.facebook.com for Facebook

The following is an example userIdentity element for the AssumeRoleWithWebIdentity action.

```
"userIdentity": {
    "type": "WebIdentityUser",
    "principalId": "accounts.google.com:application-id.apps.googleusercontent.com:user-id",
    "userName": "user-id",
    "identityProvider": "accounts.google.com"
  }
```

For example logs of how the userIdentity element appears for SAMLUser and WebIdentityUser types, see Logging IAM and AWS STS API calls with AWS CloudTrail.

AWS STS source identity

An IAM administrator can configure AWS Security Token Service to require that users specify their identity when they use temporary credentials to assume roles. The sourceIdentity field occurs in events when users assume an IAM role or perform any actions with the assumed role.

The sourceIdentity field identifies the original user identity making the request, whether that user's identity is an IAM user, an IAM role, a user authenticated by using SAML-based federation, or a user authenticated by using OpenID Connect (OIDC)-compliant web identity federation. After the IAM administrator configures AWS STS, CloudTrail logs sourceIdentity information in the following events and locations within the event record:

- The AWS STS AssumeRole, AssumeRoleWithSAML, or AssumeRoleWithWebIdentity calls that a user identity makes when it assumes a role. sourceIdentity is found in the requestParameters block of the AWS STS calls.

- The AWS STS `AssumeRole`, `AssumeRoleWithSAML`, or `AssumeRoleWithWebIdentity` calls that a user identity makes if it uses a role to assume another role, known as role chaining. `sourceIdentity` is found in the `requestParameters` block of the AWS STS calls.
- The AWS service API calls that the user identity makes while assuming a role and using the temporary credentials assigned by AWS STS. In service API events, `sourceIdentity` is found in the `sessionContext` block. For example, if a user identity creates a new S3 bucket, `sourceIdentity` occurs in the `sessionContext` block of the `CreateBucket` event.

For more information about how to configure AWS STS to collect source identity information, see Monitor and control actions taken with assumed roles in the *IAM User Guide*. For more information about AWS STS events that are logged to CloudTrail, see Logging IAM and AWS STS API calls with AWS CloudTrail in the *IAM User Guide*.

The following are example snippets of events that show the `sourceIdentity` field.

Example `requestParameters` section

In the following example event snippet, a user makes an AWS STS `AssumeRole` request, and sets a source identity, represented here by *`source-identity-value-set`*. The user assumes a role represented by the role ARN `arn:aws:iam::123456789012:role/Assumed_Role`. The `sourceIdentity` field is in the `requestParameters` block of the event.

```
"eventVersion": "1.05",
    "userIdentity": {
        "type": "AWSAccount",
        "principalId": "AIDAJ45Q7YFFAREXAMPLE",
        "accountId": "123456789012"
    },
    "eventTime": "2020-04-02T18:20:53Z",
    "eventSource": "sts.amazonaws.com",
    "eventName": "AssumeRole",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "203.0.113.64",
    "userAgent": "aws-cli/1.16.96 Python/3.6.0 Windows/10 botocore/1.12.86",
    "requestParameters": {
        "roleArn": "arn:aws:iam::123456789012:role/Assumed_Role",
        "roleSessionName": "Test1",
        "sourceIdentity": "source-identity-value-set",
    },
```

Example `responseElements` section

In the following example event snippet, a user makes an AWS STS `AssumeRole` request to assume a role named `Developer_Role`, and sets a source identity, `Admin`. The user assumes a role represented by the role ARN `arn:aws:iam::111122223333:role/Developer_Role`. The `sourceIdentity` field is shown in both the `requestParameters` and `responseElements` blocks of the event. The temporary credentials used to assume the role, the session token string, and the assumed role ID, session name, and session ARN are shown in the `responseElements` block, along with the source identity.

```
    "requestParameters": {
        "roleArn": "arn:aws:iam::111122223333:role/Developer_Role",
        "roleSessionName": "Session_Name",
        "sourceIdentity": "Admin"
    },
    "responseElements": {
        "credentials": {
            "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
            "expiration": "Jan 22, 2021 12:46:28 AM",
            "sessionToken": "XXYYaz...
                             EXAMPLE_SESSION_TOKEN
                             XXyYaZAz"
        },
        "assumedRoleUser": {
            "assumedRoleId": "AROACKCEVSQ6C2EXAMPLE:Session_Name",
            "arn": "arn:aws:sts::111122223333:assumed-role/Developer_Role/Session_Name"
        },
        "sourceIdentity": "Admin"
    }
...
```

Example `sessionContext` section

In the following example event snippet, a user is assuming a role named `DevRole` to call an AWS service API. The user sets a source identity, represented here by *`source-identity-value-set`*. The `sourceIdentity` field is in the `sessionContext` block, within the `userIdentity` block of the event.

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "AROAJ45Q7YFFAREXAMPLE: Dev1",
    "arn": "arn: aws: sts: : 123456789012: assumed-role/DevRole/Dev1",
    "accountId": "123456789012",
    "accessKeyId": "ASIAIOSFODNN7EXAMPLE",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "AROAJ45Q7YFFAREXAMPLE",
        "arn": "arn: aws: iam: : 123456789012: role/DevRole",
        "accountId": "123456789012",
        "userName": "DevRole"
      },
      "webIdFederationData": {},
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2021-02-21T23: 46: 28Z"
      },
      "sourceIdentity": "source-identity-value-set"
    }
  }
}
```

CloudTrail Insights `insightDetails` element

AWS CloudTrail Insights event records include fields that are different from other CloudTrail events in their JSON structure, sometimes called *payload*. A CloudTrail Insights event record includes an `insightDetails` block that contains information about the underlying triggers of an Insights event, such as event source, user identities, user agents, historical averages or *baselines*, statistics, API name, and whether the event is the start or end of the Insights event. The `insightDetails` block contains the following information.

- **`state`** - Whether the event is the starting or ending Insights event. The value can be `Start` or `End`.

  Since: 1.07

  Optional: False

- **`eventSource`** - The AWS service endpoint that was the source of the unusual activity, such as `ec2.amazonaws.com`.

Since: 1.07

Optional: False

- **`eventName`** - The name of the Insights event, typically the name of the API that was the source of the unusual activity.

  Since: 1.07

  Optional: False

- **`insightType`** - The type of Insights event. This value can be `ApiCallRateInsight`, `ApiErrorRateInsight`, or both.

  Since: 1.07

  Optional: False

- **`insightContext`** -

  Information about the AWS tools (called *user agents*), IAM users and roles (called *user identities*), and error codes associated with the events that CloudTrail analyzed to generate the Insights event. This element also includes statistics that show how the unusual activity in an Insights event compares to *baseline*, or normal, activity.

  Since: 1.07

  Optional: False

  - **`statistics`** - Includes data about the *baseline*, or typical average rate of calls to or errors on the subject API by an account as measured during the baseline period, the average rate of calls or errors that triggered the Insights event over the first minute of the Insights event, the duration, in minutes, of the Insights event, and the duration, in minutes, of the baseline measuring period.

    Since: 1.07

    Optional: False

    - **`baseline`** - The average number of API calls or errors per minute during the baseline duration on the Insights event's subject API for the account, calculated over the seven days preceding the start of the Insights event.

    Since: 1.07

    Optional: False

  - **`insight`** -

    For a starting Insights event, this value is the average number of API calls or errors per minute during the start of the unusual activity. For an ending Insights event, this value is the average number of API calls or errors per minute over the duration of the unusual activity.

    Since: 1.07

    Optional: False

  - **`insightDuration`** - The duration, in minutes, of an Insights event (the time period from the start to the end of unusual activity on the subject API). `insightDuration` occurs in both starting and ending Insights events.

    Since: 1.07

    Optional: False

  - **`baselineDuration`** - The duration, in minutes, of the baseline period (the time period that normal activity is measured on the subject API). `baselineDuration` is at minimum the seven days (10080 minutes) preceding an Insights event. This field occurs in both starting and ending Insights events. The ending time of `baselineDuration` measurement is always the start of an Insights event.

    Since: 1.07

    Optional: False

- **`attributions`** - This block includes information about the user identities, user agents, and error codes correlated with unusual and baseline activity. A maximum of five user identities, five user agents, and five error codes are captured in an Insights event `attributions` block, sorted by an average of the count of activity, in descending order from highest to lowest.

  Since: 1.07

  Optional: True

  - **`attribute`** - Contains the attribute type. Value can be `userIdentityArn`, `userAgent`, or `errorCode`.

- **`userIdentityArn`** - A block that shows up to the top five AWS users or IAM roles that contributed to API calls or errors during the unusual activity and baseline periods. See also `userIdentity` in [CloudTrail record contents](#).

  Since: 1.07

  Optional: False

  - **`insight`** - A block that shows up to the top five user identity ARNs that contributed to the API calls made during the unusual activity period, in descending order from largest number of API calls to smallest. It also shows the average number of API calls made by the user identities during the unusual activity period.

    Since: 1.07

    Optional: False

    - **`value`** - The ARN of one of the top five user identities that contributed to the API calls made during the unusual activity period.

      Since: 1.07

      Optional: False

    - **`average`** - The number of API calls or errors per minute during the unusual activity period for the user identity in the `value` field.

      Since: 1.07

      Optional: False

  - **`baseline`** - A block that shows up to the top five user identity ARNs that contributed the most to the API calls or errors during the normal activity period. It also shows the average number of API calls or errors logged by the user identities during the normal activity period.

    Since: 1.07

    Optional: False

    - **`value`** - The ARN of one of the top five user identities that contributed to the API calls or errors during the normal activity period.

      Since: 1.07

        Optional: False

        - **`average`** - The historic average of API calls or errors per minute during the seven days preceding the Insights activity start time for the user identity in the `value` field.

          Since: 1.07

          Optional: False

- **`userAgent`** - A block that shows up to the top five AWS tools by which the user identity contributed to API calls during the unusual activity and baseline periods. These tools include the AWS Management Console, AWS CLI, or the AWS SDKs. See also `userAgent` in CloudTrail record contents.

  Since: 1.07

  Optional: False

  - **`insight`** - A block that shows up to the top five user agents that contributed to the API calls made during the unusual activity period, in descending order from largest number of API calls to smallest. It also shows the average number of API calls or errors logged by the user agents during the unusual activity period.

    Since: 1.07

    Optional: False

    - **`value`** - One of the top five user agents that contributed to the API calls made during the unusual activity period.

      Since: 1.07

      Optional: False

    - **`average`** - The number of API calls or errors logged per minute during the unusual activity period for the user agent in the `value` field.

      Since: 1.07

      Optional: False

  - **`baseline`** - A block that shows up to the top five user agents that contributed the most to the API calls made during the normal activity period. It also shows the average number of API calls or errors logged by the user agents during the normal activity period.

    Since: 1.07

    Optional: False

    - **`value`** - One of the top five user agents that contributed to the API calls or errors logged during the normal activity period.

      Since: 1.07

      Optional: False

    - **`average`** - The historic average of API calls or errors per minute during the seven days preceding the Insights activity start time for the user agent in the `value` field.

      Since: 1.07

      Optional: False

- **`errorCode`** - A block that shows up to the top five error codes that occurred on API calls during the unusual activity and baseline periods, in descending order from largest number of API calls to smallest. See also `errorCode` in CloudTrail record contents.

  Since: 1.07

  Optional: False

  - **`insight`** - A block that shows up to the top five error codes that occurred on the API calls made during the unusual activity period, in descending order from largest number of associated API calls to smallest. It also shows the average number of API calls on which the errors occurred during the unusual activity period.

    Since: 1.07

    Optional: False

    - **`value`** - One of the top five error codes that occurred on the API calls made during the unusual activity period, such as `AccessDeniedException`.

      If none of the calls that triggered the Insights event resulted in errors, this value is `null`.

    Since: 1.07

    Optional: False

  - **`average`** - The number of API calls per minute during the unusual activity period for the error code in the `value` field.

    If the error code value is `null`, and there are no other error codes in the `insight` block, the value of the `average` is the same as that in the `statistics` block for the Insights event overall.

    Since: 1.07

    Optional: False

- **`baseline`** - A block that shows up to the top five error codes that occurred on the API calls made during the normal activity period. It also shows the average number of API calls made by the user agents during the normal activity period.

  Since: 1.07

  Optional: False

  - **`value`** - One of the top five error codes that occurred on the API calls made during the normal activity period, such as `AccessDeniedException`.

    Since: 1.07

    Optional: False

  - **`average`** - The historic average of API calls or errors per minute during the seven days preceding the Insights activity start time for the error code in the `value` field.

    Since: 1.07

    Optional: False

Example `insightDetails` block

The following is an example of an Insights event `insightDetails` block for an Insights event that occurred when the Application Auto Scaling API `CompleteLifecycleAction` was called an unusual number of times. For an example of a full Insights event, see [Insights events](#).

This example is from a starting Insights event, indicated by `"state": "Start"`. The top user identities that called the APIs associated with the Insights event, `CodeDeployRole1`, `CodeDeployRole2`, and `CodeDeployRole3`, are shown in the `attributions` block, along with their average API call rates for this Insights event, and the baseline for the `CodeDeployRole1` role. The `attributions` block also shows that the user agent is `codedeploy.amazonaws.com`, meaning the top user identities used the AWS CodeDeploy console to run the API calls.

Because there are no error codes associated with the events that were analyzed to generate the Insights event (the value is `null`), the `insight` average for the error code is the same as the overall `insight` average for the entire Insights event, shown in the `statistics` block.

```
          "insightDetails": {
            "state": "Start",
            "eventSource": "autoscaling.amazonaws.com",
            "eventName": "CompleteLifecycleAction",
            "insightType": "ApiCallRateInsight",
            "insightContext": {
              "statistics": {
                "baseline": {
                  "average": 0.0000882145
                },
                "insight": {
                  "average": 0.6
                },
                "insightDuration": 5,
                "baselineDuration": 11336
              },
              "attributions": [
                {
                  "attribute": "userIdentityArn",
                  "insight": [
                    {
                      "value": "arn:aws:sts::012345678901:assumed-role/
CodeDeployRole1",
                      "average": 0.2
                    },
                    {
                      "value": "arn:aws:sts::012345678901:assumed-role/
CodeDeployRole2",
                      "average": 0.2
                    },
                    {
                      "value": "arn:aws:sts::012345678901:assumed-role/
CodeDeployRole3",
                      "average": 0.2
                    }
                  ],
                  "baseline": [
                    {
                      "value": "arn:aws:sts::012345678901:assumed-role/
CodeDeployRole1",
                      "average": 0.0000882145
                    }
                  ]
                },
                {
                  "attribute": "userAgent",
                  "insight": [
                    {
                      "value": "codedeploy.amazonaws.com",
                      "average": 0.6
                    }
                  ],
                  "baseline": [
                    {
                      "value": "codedeploy.amazonaws.com",
                      "average": 0.0000882145
                    }
                  ]
                },
                {
                  "attribute": "errorCode",
                  "insight": [
                    {
                      "value": "null",
                      "average": 0.6
                    }
                  ],
                  "baseline": [
                    {
                      "value": "null",
                      "average": 0.0000882145
                    }
                  ]
                }
              ]
```

```
            }
          }
```

Non-API events captured by CloudTrail

In addition to logging AWS API calls, CloudTrail captures other related events that might have a security or compliance impact on your AWS account or that might help you troubleshoot operational problems.

- AWS service events – CloudTrail supports logging non-API service events. These events are created by AWS services but are not directly triggered by a request to a public AWS API. For these events, the `eventType` field is `AwsServiceEvent`.
- AWS Management Console sign-in events – CloudTrail logs attempts to sign in to the AWS Management Console, the AWS Discussion Forums, and the AWS Support Center. All IAM user and root user sign-in events, as well as all federated user sign-in events, generate records in CloudTrail. For sign-in events, the `eventType` field is `AwsConsoleSignIn`.

AWS service events

CloudTrail supports logging non-API service events. These events are created by AWS services but are not directly triggered by a request to a public AWS API. For these events, the `eventType` field is `AwsServiceEvent`.

The following is an example scenario of an AWS service event when a customer managed key is automatically rotated in AWS Key Management Service (AWS KMS). For more information about rotating KMS keys, see Rotating KMS keys.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "accountId": "111122223333",
        "invokedBy": "AWS Internal"
    },
    "eventTime": "2021-01-14T01:41:59Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "RotateKey",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "AWS Internal",
    "userAgent": "AWS Internal",
    "requestParameters": null,
    "responseElements": null,
    "eventID": "a24b3967-ddad-417f-9b22-2332b918db06",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        }
    ],
    "eventType": "AwsServiceEvent",
    "recipientAccountId": "111122223333",
    "serviceEventDetails": {
        "rotationType": "AUTOMATIC",
        "keyId": "1234abcd-12ab-34cd-56ef-1234567890ab"
    },
    "eventCategory": "Management"
}
```

AWS Management Console sign-in events

CloudTrail logs attempts to sign in to the AWS Management Console, the AWS Discussion Forums, and the AWS Support Center. All IAM user and root user sign-in events, as well as all federated user sign-in events, generate records in CloudTrail log files. For information about finding and viewing logs, see Finding your CloudTrail log files and Downloading your CloudTrail log files.

Note

The Region recorded in a `ConsoleLogin` event varies based on the user type and whether you use a global or regional endpoint to sign in.

- If you sign in as the root user, CloudTrail records the event in us-east-1.
- If you sign in with an IAM user and use the global endpoint, CloudTrail records the Region of the `ConsoleLogin` event as follows:
    - If an account alias cookie is present in the browser, CloudTrail records the `ConsoleLogin` event in one of the following regions: us-east-2, eu-north-1, or ap-southeast-2. This is because the console proxy redirects the user based on the latency from the user sign-in location.

    - If an account alias cookie is not present in the browser, CloudTrail records the ConsoleLogin event in us-east-1. This is because the console proxy redirects back to the global sign-in.
- If you sign in with an IAM user and use a Regional endpoint, CloudTrail records the ConsoleLogin event in the appropriate Region for the endpoint. For more information about AWS Sign-In endpoints, see AWS Sign-In endpoints and quotas.

Topics

- Example event records for IAM users
- Example event records for root users
- Example event records for federated users

Example event records for IAM users

The following examples show event records for several IAM user sign-in scenarios.

Topics

- IAM user, successful sign-in without MFA
- IAM user, successful sign-in with MFA
- IAM user, unsuccessful sign-in
- IAM user, sign-in process checks for MFA (single MFA device type)
- IAM user, sign-in process checks for MFA (multiple MFA device types)

IAM user, successful sign-in without MFA

The following record shows that a user named Anaya successfully signed in to the AWS Management Console without using multi-factor authentication (MFA).

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EXAMPLE6E4XEGITWATV6R",
        "arn": "arn:aws:iam::999999999999:user/Anaya",
        "accountId": "999999999999",
        "userName": "Anaya"
    },
    "eventTime": "2023-07-19T21:44:40Z",
    "eventSource": "signin.amazonaws.com",
    "eventName": "ConsoleLogin",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:102.0) Gecko/20100101
 Firefox/102.0",
    "requestParameters": null,
    "responseElements": {
        "ConsoleLogin": "Success"
    },
    "additionalEventData": {
        "LoginTo": "https://console.aws.amazon.com/console/home?hashArgs=
%23&isauthcode=true&state=hashArgsFromTB_us-east-1_examplee9aba7f8",
        "MobileVersion": "No",
        "MFAUsed": "No"
    },
    "eventID": "e1bf1000-86a4-4a78-81d7-EXAMPLE83102",
    "readOnly": false,
    "eventType": "AwsConsoleSignIn",
    "managementEvent": true,
    "recipientAccountId": "999999999999",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_128_GCM_SHA256",
        "clientProvidedHostHeader": "us-east-1.signin.aws.amazon.com"
    }
}
```

IAM user, successful sign-in with MFA

The following record shows that an IAM user named Anaya successfully signed in to the AWS Management Console using multi-factor authentication (MFA).

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EXAMPLE6E4XEGITWATV6R",
        "arn": "arn:aws:iam::999999999999:user/Anaya",
        "accountId": "999999999999",
        "userName": "Anaya"
    },
    "eventTime": "2023-07-19T22:01:30Z",
    "eventSource": "signin.amazonaws.com",
    "eventName": "ConsoleLogin",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:102.0) Gecko/20100101
 Firefox/102.0",
    "requestParameters": null,
    "responseElements": {
        "ConsoleLogin": "Success"
    },
    "additionalEventData": {
        "LoginTo": "https://console.aws.amazon.com/console/home?hashArgs=
%23&isauthcode=true&state=hashArgsFromTB_us-east-1_examplebde32f3c9",
        "MobileVersion": "No",
        "MFAIdentifier": "arn:aws:iam::999999999999:mfa/mfa-device",
        "MFAUsed": "Yes"
    },
    "eventID": "e1f76697-5beb-46e8-9cfc-EXAMPLEbde31",
    "readOnly": false,
    "eventType": "AwsConsoleSignIn",
    "managementEvent": true,
    "recipientAccountId": "999999999999",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_128_GCM_SHA256",
        "clientProvidedHostHeader": "us-east-1.signin.aws.amazon.com"
    }
}
```

IAM user, unsuccessful sign-in

The following record shows an unsuccessful sign-in attempt from an IAM user named Paulo.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EXAMPLE6E4XEGITWATV6R",
        "accountId": "123456789012",
        "accessKeyId": "",
        "userName": "Paulo"
    },
    "eventTime": "2023-07-19T22:01:20Z",
    "eventSource": "signin.amazonaws.com",
    "eventName": "ConsoleLogin",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:102.0) Gecko/20100101
 Firefox/102.0",
    "errorMessage": "Failed authentication",
    "requestParameters": null,
    "responseElements": {
        "ConsoleLogin": "Failure"
    },
    "additionalEventData": {
        "LoginTo": "https://console.aws.amazon.com/console/home?hashArgs=
%23&isauthcode=true&state=hashArgsFromTB_us-east-1_examplebde32f3c9",
        "MobileVersion": "No",
        "MFAUsed": "Yes"
    },
    "eventID": "66c97220-2b7d-43b6-a7a0-EXAMPLEbae9c",
    "readOnly": false,
    "eventType": "AwsConsoleSignIn",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_128_GCM_SHA256",
        "clientProvidedHostHeader": "us-east-1.signin.aws.amazon.com"
    }
}
```

IAM user, sign-in process checks for MFA (single MFA device type)

The following shows that the sign-process checked whether multi-factor authentication (MFA) is required for an IAM user during sign-in. In this example, the `mfaType` value is `U2F MFA`, which indicates that the IAM user enabled either a single MFA device or multiple MFA devices of the same type (`U2F MFA`).

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EXAMPLE6E4XEGITWATV6R",
        "accountId": "123456789012",
        "accessKeyId": "",
        "userName": "Alice"
    },
    "eventTime": "2023-07-19T22:01:26Z",
    "eventSource": "signin.amazonaws.com",
    "eventName": "CheckMfa",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:102.0) Gecko/20100101
 Firefox/102.0",
    "requestParameters": null,
    "responseElements": {
        "CheckMfa": "Success"
    },
    "additionalEventData": {
        "MfaType": "Virtual MFA"
    },
    "eventID": "7d8a0746-b2e7-44f5-9917-EXAMPLEfb77c",
    "readOnly": false,
    "eventType": "AwsConsoleSignIn",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_128_GCM_SHA256",
        "clientProvidedHostHeader": "us-east-1.signin.aws.amazon.com"
    }
}
```

IAM user, sign-in process checks for MFA (multiple MFA device types)

The following shows that the sign-process checked whether multi-factor authentication (MFA) is required for an IAM user during sign-in. In this example, the `mfaType` value is `Multiple MFA Devices`, which indicates that the IAM user enabled multiple MFA device types.

```
 {
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EXAMPLE6E4XEGITWATV6R",
        "accountId": "123456789012",
        "accessKeyId": "",
        "userName": "Mary"
    },
    "eventTime": "2023-07-19T23:10:09Z",
    "eventSource": "signin.amazonaws.com",
    "eventName": "CheckMfa",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:102.0) Gecko/20100101
 Firefox/102.0",
    "requestParameters": null,
    "responseElements": {
        "CheckMfa": "Success"
    },
    "additionalEventData": {
        "MfaType": "Multiple MFA Devices"
    },
    "eventID": "19bd1a1c-76b1-4806-9d8f-EXAMPLE02a96",
    "readOnly": false,
    "eventType": "AwsConsoleSignIn",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_128_GCM_SHA256",
        "clientProvidedHostHeader": "signin.aws.amazon.com"
    }
}
```

Example event records for root users

The following examples show event records for several `root` user sign-in scenarios. When you sign-in using the root user, CloudTrail records the `ConsoleLogin` event in us-east-1.

Topics

- Root user, successful sign-in without MFA
- Root user, successful sign-in with MFA

- Root user, unsuccessful sign-in
- Root user, MFA changed
- Root user, password changed

Root user, successful sign-in without MFA

The following shows a successful sign-in event for a root user not using multi-factor authentication (MFA).

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "Root",
        "principalId": "111122223333",
        "arn": "arn:aws:iam::111122223333:root",
        "accountId": "111122223333",
        "accessKeyId": ""
    },
    "eventTime": "2023-07-12T13:35:31Z",
    "eventSource": "signin.amazonaws.com",
    "eventName": "ConsoleLogin",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
 like Gecko) Chrome/114.0.0.0 Safari/537.36",
    "requestParameters": null,
    "responseElements": {
        "ConsoleLogin": "Success"
    },
    "additionalEventData": {
        "LoginTo": "https://console.aws.amazon.com/console/home?hashArgs=
%23&isauthcode=true&nc2=h_ct&src=header-signin&state=hashArgsFromTB_ap-
southeast-2_example80afacd389",
        "MobileVersion": "No",
        "MFAUsed": "No"
    },
    "eventID": "4217cc13-7328-4820-a90c-EXAMPLE8002e6",
    "readOnly": false,
    "eventType": "AwsConsoleSignIn",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_128_GCM_SHA256",
        "clientProvidedHostHeader": "signin.aws.amazon.com"
    }
}
```

Root user, successful sign-in with MFA

The following shows a successful sign-in event for a root user using multi-factor authentication (MFA).

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "Root",
        "principalId": "444455556666",
        "arn": "arn:aws:iam::444455556666:root",
        "accountId": "444455556666",
        "accessKeyId": ""
    },
    "eventTime": "2023-07-13T03:04:43Z",
    "eventSource": "signin.amazonaws.com",
    "eventName": "ConsoleLogin",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like
 Gecko) Chrome/114.0.0.0 Safari/537.36",
    "requestParameters": null,
    "responseElements": {
        "ConsoleLogin": "Success"
    },
    "additionalEventData": {
        "LoginTo": "https://ap-southeast-1.console.aws.amazon.com/ec2/home?region=ap-
southeast-1&state=hashArgs%23Instances%3Av%3D3%3B%24case%3Dtags%3Atrue%255C%2Cclient
%3Afalse%3B%24regex%3Dtags%3Afalse%255C%2Cclient%3Afalse&isauthcode=true",
        "MobileVersion": "No",
        "MFAIdentifier": "arn:aws:iam::444455556666:mfa/root-account-mfa-device",
        "MFAUsed": "Yes"
    },
    "eventID": "e0176723-ea76-4275-83a3-EXAMPLEf03fb",
    "readOnly": false,
    "eventType": "AwsConsoleSignIn",
    "managementEvent": true,
    "recipientAccountId": "444455556666",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_128_GCM_SHA256",
        "clientProvidedHostHeader": "signin.aws.amazon.com"
    }
}
```

Root user, unsuccessful sign-in

The following shows an unsuccessful sign-in event for a root user not using MFA.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "Root",
        "principalId": "123456789012",
        "arn": "arn:aws:iam::123456789012:root",
        "accountId": "123456789012",
        "accessKeyId": ""
    },
    "eventTime": "2023-07-16T04:33:40Z",
    "eventSource": "signin.amazonaws.com",
    "eventName": "ConsoleLogin",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
 like Gecko) Chrome/111.0.0.0 Safari/537.36",
    "errorMessage": "Failed authentication",
    "requestParameters": null,
    "responseElements": {
        "ConsoleLogin": "Failure"
    },
    "additionalEventData": {
        "LoginTo": "https://us-east-1.console.aws.amazon.com/billing/home?region=us-
east-1&state=hashArgs%23%2Faccount&isauthcode=true",
        "MobileVersion": "No",
        "MFAUsed": "No"
    },
    "eventID": "f28d4329-5050-480b-8de0-EXAMPLE07329",
    "readOnly": false,
    "eventType": "AwsConsoleSignIn",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_128_GCM_SHA256",
        "clientProvidedHostHeader": "signin.aws.amazon.com"
    }
}
```

Root user, MFA changed

The following shows an example event for a root user changing multi-factor authentication (MFA) settings.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "Root",
        "principalId": "111122223333",
        "arn": "arn:aws:iam::111122223333:root",
        "accountId": "111122223333",
        "accessKeyId": "EXAMPLE4XX3IEV4PFQTH",
        "userName": "AWS ROOT USER",
        "sessionContext": {
            "sessionIssuer": {},
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2023-07-15T03:51:12Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2023-07-15T04:37:08Z",
    "eventSource": "iam.amazonaws.com",
    "eventName": "EnableMFADevice",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
 like Gecko) Chrome/111.0.0.0 Safari/537.36",
    "requestParameters": {
        "userName": "AWS ROOT USER",
        "serialNumber": "arn:aws:iam::111122223333:mfa/root-account-mfa-device"
    },
    "responseElements": null,
    "requestID": "9b45cd4c-a598-41e7-9170-EXAMPLE535f0",
    "eventID": "b4f18d55-d36f-49a0-afcb-EXAMPLEc026b",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management",
    "sessionCredentialFromConsole": "true"
}
```

Root user, password changed

The following shows an example event for a root user changing their password.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "Root",
        "principalId": "444455556666",
        "arn": "arn:aws:iam::444455556666:root",
        "accountId": "444455556666",
        "accessKeyId": "EXAMPLEAOTKEG44KPW5P",
        "sessionContext": {
            "sessionIssuer": {},
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2022-11-25T13:01:14Z",
                "mfaAuthenticated": "false"
             }
        }
    },
    "eventTime": "2022-11-25T13:01:14Z",
    "eventSource": "iam.amazonaws.com",
    "eventName": "ChangePassword",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
 like Gecko) Chrome/111.0.0.0 Safari/537.36",
    "requestParameters": null,
    "responseElements": null,
    "requestID": "c64254c2-e4ff-49c0-900e-EXAMPLE9e6d2",
    "eventID": "d059176c-4f4d-4a9e-b8d7-EXAMPLE2b7b3",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "444455556666",
    "eventCategory": "Management"
}
```

Example event records for federated users

The following examples show event records for federated users. Federated users are given temporary security credentials to access AWS resources through an `AssumeRole` request.

The following shows an example event for a federation encryption request. The original access key ID is provided in the `accessKeyId` field of the `userIdentity` element. The `accessKeyId` field in the `responseElements` contains a new access key ID if the requested `sessionDuration` is passed in the encryption request, otherwise it contains the value of the original access key ID.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "EXAMPLEUU4MH7OYK5ZCOA:JohnDoe",
        "arn": "arn:aws:sts::123456789012:assumed-role/roleName/JohnDoe",
        "accountId": "123456789012",
        "accessKeyId": "originalAccessKeyID",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "EXAMPLEUU4MH7OYK5ZCOA",
                "arn": "arn:aws:iam::123456789012:role/roleName",
                "accountId": "123456789012",
                "userName": "roleName"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2023-09-25T21:30:39Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2023-09-25T21:30:39Z",
    "eventSource": "signin.amazonaws.com",
    "eventName": "GetSigninToken",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "Java/1.8.0_382",
    "requestParameters": null,
    "responseElements": {
        "credentials": {
            "accessKeyId": "accessKeyID"
        },
        "GetSigninToken": "Success"
    },
    "additionalEventData": {
        "MobileVersion": "No",
        "MFAUsed": "No"
    },
    "eventID": "1d66615b-a417-40da-a38e-EXAMPLE8c89b",
    "readOnly": false,
    "eventType": "AwsConsoleSignIn",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_128_GCM_SHA256",
        "clientProvidedHostHeader": "us-east-1.signin.aws.amazon.com"
    }
}
```

The following shows a successful sign-in event for a federated user; not using multi-factor authentication (MFA).

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "EXAMPLEPHCNW7ZCASLJOH:JohnDoe",
        "arn": "arn:aws:sts::123456789012:assumed-role/RoleName/JohnDoe",
        "accountId": "123456789012",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "EXAMPLEPHCNW7ZCASLJOH",
                "arn": "arn:aws:iam::123456789012:role/RoleName",
                "accountId": "123456789012",
                "userName": "RoleName"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2023-09-22T16:15:47Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2023-09-22T16:15:47Z",
    "eventSource": "signin.amazonaws.com",
    "eventName": "ConsoleLogin",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36
 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36",
    "requestParameters": null,
    "responseElements": {
        "ConsoleLogin": "Success"
    },
    "additionalEventData": {
        "MobileVersion": "No",
        "MFAUsed": "No"
    },
    "eventID": "b73f1ec6-c064-4cd3-ba83-EXAMPLE441d7",
    "readOnly": false,
    "eventType": "AwsConsoleSignIn",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_128_GCM_SHA256",
        "clientProvidedHostHeader": "us-east-1.signin.aws.amazon.com"
    }
}
```

Working with CloudTrail log files

You can perform more advanced tasks with your CloudTrail files.

- Create multiple trails per Region.
- Monitor CloudTrail log files by sending them to CloudWatch Logs.
- Share log files between accounts.
- Use the AWS CloudTrail Processing Library to write log processing applications in Java.
- Validate your log files to verify that they have not changed after delivery by CloudTrail.

When an event occurs in your account, CloudTrail evaluates whether the event matches the settings for your trails. Only events that match your trail settings are delivered to your Amazon S3 bucket and Amazon CloudWatch Logs log group.

You can configure multiple trails differently so that the trails process and log only the events that you specify. For example, one trail can log read-only data and management events, so that all read-only events are delivered to one S3 bucket. Another trail can log only write-only data and management events, so that all write-only events are delivered to a separate S3 bucket.

You can also configure your trails to have one trail log and deliver all management events to one S3 bucket, and configure another trail to log and deliver all data events to another S3 bucket.

You can configure your trails to log the following:

- **Data events**: These events provide visibility into the resource operations performed on or within a resource. These are also known as data plane operations.
- **Management events**: Management events provide visibility into management operations that are performed on resources in your AWS account. These are also known as control plane operations. Management events can also include non-API events that occur in your account. For example, when a user logs in to your account, CloudTrail logs the `ConsoleLogin` event. For more information, see Non-API events captured by CloudTrail.
- **Insights events**: Insights events capture unusual activity that is detected in your account. If you have Insights events enabled, and CloudTrail detects unusual activity, Insights events are logged to the destination S3 bucket for your trail, but in a different folder. You can also see the type of Insights event and the incident time period when you view Insights events on the CloudTrail console. Unlike other types of events captured in a CloudTrail trail, Insights events are logged

only when CloudTrail detects changes in your account's API usage that differ significantly from the account's typical usage patterns.

Insights events are generated only for management APIs. For more information, see Logging Insights events.

Note

CloudTrail typically delivers logs within an average of about 5 minutes of an API call. This time is not guaranteed. Review the AWS CloudTrail Service Level Agreement for more information.
If you misconfigure your trail (for example, the S3 bucket is unreachable), CloudTrail will attempt to redeliver the log files to your S3 bucket for 30 days, and these attempted-to-deliver events will be subject to standard CloudTrail charges. To avoid charges on a misconfigured trail, you need to delete the trail.

Topics

Receiving CloudTrail log files from multiple Regions

You can configure CloudTrail to deliver log files from multiple Regions to a single S3 bucket for a single account. For example, you have a trail in the US West (Oregon) Region that is configured to deliver log files to a S3 bucket, and a CloudWatch Logs log group. When you change an existing single-Region trail to log all Regions, CloudTrail logs events from all Regions that are in a single AWS partition in your account. CloudTrail delivers log files to the same S3 bucket and CloudWatch

Logs log group. As long as CloudTrail has permissions to write to an S3 bucket, the bucket for a multi-Region trail does not have to be in the trail's home Region.

To log events across all Regions in all AWS partitions in your account, create a multi-Region trail in each partition.

In the console, by default, you create a trail that logs events in all AWS Regions in the AWS partition in which you are working. This is a recommended best practice. To log events in a single Region (not recommended), use the AWS CLI. To configure an existing single-Region trail to log in all Regions, you must use the AWS CLI.

To change an existing trail so that it applies to all Regions, add the `--is-multi-region-trail` option to the **update-trail** command.

```
aws cloudtrail update-trail --name my-trail --is-multi-region-trail
```

To confirm that the trail now applies to all Regions, the `IsMultiRegionTrail` element in the output shows `true`.

```
{
    "IncludeGlobalServiceEvents": true,
    "Name": "my-trail",
    "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/my-trail",
    "LogFileValidationEnabled": false,
    "IsMultiRegionTrail": true,
    "IsOrganizationTrail": false,
    "S3BucketName": "DOC-EXAMPLE-BUCKET"
}
```

> **Note**
>
> When a new Region launches in the aws partition, CloudTrail automatically creates a trail for you in the new Region with the same settings as your original trail.

For more information, see the following resources:

- Working with CloudTrail trails
- CloudTrail FAQs

Managing data consistency in CloudTrail

CloudTrail uses a distributed computing model called eventual consistency. Any change that you make to your CloudTrail configuration (or other AWS services), including tags used in attribute-based access control (ABAC), takes time to become visible from all possible endpoints. Some of the delay results from the time it takes to send the data from server to server, from replication zone to replication zone, and from Region to Region around the world. CloudTrail also uses caching to improve performance, but in some cases this can add time. The change might not be visible until the previously cached data times out.

You must design your applications to account for these potential delays. Ensure that they work as expected, even when a change made in one location is not instantly visible at another. Such changes include creating or updating trails or event data stores, updating event selectors, and starting or stopping logging. When you create or update a trail or event data store, CloudTrail delivers logs to the S3 bucket or event data store based on the last known configuration until the changes propagate to all locations.

For more information about how this affects other AWS services, see the following resources:

- **Amazon DynamoDB**: What is the consistency model of DynamoDB? in the *DynamoDB FAQ*, and Read consistency in the *Amazon DynamoDB Developer Guide*.
- **Amazon EC2**: Eventual consistency in the *Amazon Elastic Compute Cloud API Reference*.
- **Amazon EMR**: Ensuring Consistency When Using Amazon S3 and Amazon Elastic MapReduce for ETL Workflows in the *AWS Big Data Blog*.
- **AWS Identity and Access Management (IAM)**: Changes that I make are not always immediately visible in the *IAM User Guide*.
- **Amazon Redshift**: Managing data consistency in the *Amazon Redshift Database Developer Guide*.
- **Amazon S3**: Amazon S3 data consistency model in the *Amazon Simple Storage Service User Guide*.

Monitoring CloudTrail Log Files with Amazon CloudWatch Logs

You can use Amazon CloudWatch Logs to monitor, store, and access your log files from CloudTrail.

CloudWatch Logs enables you to centralize the logs from all of your systems, applications, and AWS services that you use, in a single, highly scalable service. You can then easily view them, search them for specific error codes or patterns, filter them based on specific fields, or archive them securely for future analysis. CloudWatch Logs enables you to see all of your logs, regardless of their source, as a single and consistent flow of events ordered by time.

Complete the following steps to configure CloudTrail with CloudWatch Logs to monitor your trail logs and be notified when specific activity occurs.

1. Configure your trail to send log events to CloudWatch Logs.
2. Define CloudWatch Logs metric filters to evaluate log events for matches in terms, phrases, or values. For example, you can monitor for `ConsoleLogin` events.
3. Assign CloudWatch metrics to the metric filters.
4. Create CloudWatch alarms that are triggered according to thresholds and time periods that you specify. You can configure alarms to send notifications when alarms are triggered, so that you can take action.
5. You can also configure CloudWatch to automatically perform an action in response to an alarm.

Standard pricing for Amazon CloudWatch and Amazon CloudWatch Logs applies. For more information, see Amazon CloudWatch Pricing.

For more information about the Regions in which you can configure your trails to send logs to CloudWatch Logs, see Amazon CloudWatch Logs Regions and Quotas in the *AWS General Reference*.

Topics

Sending events to CloudWatch Logs

When you configure your trail to send events to CloudWatch Logs, CloudTrail sends only the events that match your trail settings. For example, if you configure your trail to log data events only, your trail sends data events only to your CloudWatch Logs log group. CloudTrail supports sending data, Insights, and management events to CloudWatch Logs. For more information, see Working with CloudTrail log files.

Note

Only the management account can configure a CloudWatch Logs log group for an organization trail using the console. The delegated administrator can configure a CloudWatch Logs log group using the AWS CLI or CloudTrail `CreateTrail` or `UpdateTrail` API operations.

To send events to a CloudWatch Logs log group:

- Make sure you have sufficient permissions to create or specify an IAM role. For more information, see Granting permission to view and configure Amazon CloudWatch Logs information on the CloudTrail console.
- If you're configuring the CloudWatch Logs log group using the AWS CLI, make sure you have sufficient permissions to create a CloudWatch Logs log stream in the log group you specify and to deliver CloudTrail events to that log stream. For more information, see Creating a policy document.
- Create a new trail or specify an existing one. For more information, see Creating and updating a trail with the console.
- Create a log group or specify an existing one.
- Specify an IAM role. If you are modifying an existing IAM role for an organization trail, you must manually update the policy to allow logging for the organization trail. For more information, see this policy example and Creating a trail for an organization.
- Attach a role policy or use the default.

Contents

Configuring CloudWatch Logs monitoring with the console

You can use the AWS Management Console to configure your trail to send events to CloudWatch Logs for monitoring.

Creating a log group or specifying an existing log group

CloudTrail uses a CloudWatch Logs log group as a delivery endpoint for log events. You can create a log group or specify an existing one.

To create or specify a log group for an existing trail

1. Make sure you log in with an administrative user or role with sufficient permissions to configure CloudWatch Logs integration. For more information, see Granting permission to view and configure Amazon CloudWatch Logs information on the CloudTrail console.

   > **Note**
   >
   > Only the management account can configure a CloudWatch Logs log group for an organization trail using the console. The delegated administrator can configure a CloudWatch Logs log group using the AWS CLI or CloudTrail `CreateTrail` or `UpdateTrail` API operations.

2. Open the CloudTrail console at https://console.aws.amazon.com/cloudtrail/.
3. Choose the trail name. If you choose a trail that applies to all Regions, you will be redirected to the Region in which the trail was created. You can create a log group or choose an existing log group in the same Region as the trail.

   > **Note**
   >
   > A trail that applies to all Regions sends log files from all Regions to the CloudWatch Logs log group that you specify.

4. In **CloudWatch Logs**, choose **Edit**.
5. For **CloudWatch Logs**, choose **Enabled**.

6. For **Log group name**, choose **New** to create a new log group, or **Existing** to use an existing one. If you choose **New**, CloudTrail specifies a name for the new log group for you, or you can type a name. For more information about naming, see CloudWatch log group and log stream naming for CloudTrail.
7. If you choose **Existing**, choose a log group from the drop-down list.
8. For **Role name**, choose **New** to create a new IAM role for permissions to send logs to CloudWatch Logs. Choose **Existing** to choose an existing IAM role from the drop-down list. The policy statement for the new or existing role is displayed when you expand **Policy document**. For more information about this role, see Role policy document for CloudTrail to use CloudWatch Logs for monitoring.

   > **Note**
   >
   > When you configure a trail, you can choose an S3 bucket and SNS topic that belong to another account. However, if you want CloudTrail to deliver events to a CloudWatch Logs log group, you must choose a log group that exists in your current account.

9. Choose **Save changes**.

Specifying an IAM role

You can specify a role for CloudTrail to assume to deliver events to the log stream.

To specify a role

1. By default, the `CloudTrail_CloudWatchLogs_Role` is specified for you. The default role policy has the required permissions to create a CloudWatch Logs log stream in a log group that you specify, and to deliver CloudTrail events to that log stream.

   > **Note**
   >
   > If you want to use this role for a log group for an organization trail, you must manually modify the policy after you create the role. For more information, see this policy example and Creating a trail for an organization.

   a. To verify the role, go to the AWS Identity and Access Management console at https://console.aws.amazon.com/iam/.

   b. Choose **Roles** and then choose the **CloudTrail_CloudWatchLogs_Role**.

   c. From the **Permissions** tab, expand the policy to view its contents.

2. You can specify another role, but you must attach the required role policy to the existing role if you want to use it to send events to CloudWatch Logs. For more information, see Role policy document for CloudTrail to use CloudWatch Logs for monitoring.

Viewing events in the CloudWatch console

After you configure your trail to send events to your CloudWatch Logs log group, you can view the events in the CloudWatch console. CloudTrail typically delivers events to your log group within an average of about 5 minutes of an API call. This time is not guaranteed. Review the AWS CloudTrail Service Level Agreement for more information.

To view events in the CloudWatch console

1. Open the CloudWatch console at https://console.aws.amazon.com/cloudwatch/.
2. In the left navigation pane, under **Logs**, choose **Log groups**.
3. Choose the log group that you specified for your trail.
4. Choose the log stream that you want to view.
5. To see the details of the event that your trail logged, choose an event.

> **Note**
>
> The **Time (UTC)** column in the CloudWatch console shows when the event was delivered to your log group. To see the actual time that the event was logged by CloudTrail, see the `eventTime` field.

Configuring CloudWatch Logs monitoring with the AWS CLI

You can use the AWS CLI to configure CloudTrail to send events to CloudWatch Logs for monitoring.

Creating a log group

1. If you don't have an existing log group, create a CloudWatch Logs log group as a delivery endpoint for log events using the CloudWatch Logs `create-log-group` command.

   ```
   aws logs create-log-group --log-group-name name
   ```

   The following example creates a log group named `CloudTrail/logs`:

   ```
   aws logs create-log-group --log-group-name CloudTrail/logs
   ```

2. Retrieve the log group Amazon Resource Name (ARN).

   ```
   aws logs describe-log-groups
   ```

Creating a role

Create a role for CloudTrail that enables it to send events to the CloudWatch Logs log group. The IAM `create-role` command takes two parameters: a role name and a file path to an assume role policy document in JSON format. The policy document that you use gives `AssumeRole` permissions to CloudTrail. The `create-role` command creates the role with the required permissions.

To create the JSON file that will contain the policy document, open a text editor and save the following policy contents in a file called `assume_role_policy_document.json`.

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

Run the following command to create the role with AssumeRole permissions for CloudTrail.

```
aws iam create-role --role-name role_name --assume-role-policy-document file://<path to
 assume_role_policy_document>.json
```

When the command completes, take a note of the role ARN in the output.

Creating a policy document

Create the following role policy document for CloudTrail. This document grants CloudTrail the permissions required to create a CloudWatch Logs log stream in the log group you specify and to deliver CloudTrail events to that log stream.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {

      "Sid": "AWSCloudTrailCreateLogStream2014110",
      "Effect": "Allow",
      "Action": [
        "logs:CreateLogStream"
      ],
      "Resource": [
        "arn:aws:logs:region:accountID:log-group:log_group_name:log-
stream:accountID_CloudTrail_region*"
      ]

    },
    {
      "Sid": "AWSCloudTrailPutLogEvents20141101",
      "Effect": "Allow",
      "Action": [
        "logs:PutLogEvents"
      ],
      "Resource": [
        "arn:aws:logs:region:accountID:log-group:log_group_name:log-
stream:accountID_CloudTrail_region*"
      ]
    }
  ]
}
```

Save the policy document in a file called `role-policy-document.json`.

If you're creating a policy that might be used for organization trails as well, you will need to configure it slightly differently. For example, the following policy grants CloudTrail the permissions required to create a CloudWatch Logs log stream in the log group you specify and to deliver CloudTrail events to that log stream for both trails in the AWS account 111111111111 and for organization trails created in the 111111111111 account that are applied to the AWS Organizations organization with the ID of *`o-exampleorgid`*:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AWSCloudTrailCreateLogStream20141101",
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogStream"
            ],
            "Resource": [
                "arn:aws:logs:us-east-2:111111111111:log-group:CloudTrail/
DefaultLogGroupTest:log-stream:111111111111_CloudTrail_us-east-2*",
                "arn:aws:logs:us-east-2:111111111111:log-group:CloudTrail/
DefaultLogGroupTest:log-stream:o-exampleorgid_*"
            ]
        },
        {
            "Sid": "AWSCloudTrailPutLogEvents20141101",
            "Effect": "Allow",
            "Action": [
                "logs:PutLogEvents"
            ],
            "Resource": [
                "arn:aws:logs:us-east-2:111111111111:log-group:CloudTrail/
DefaultLogGroupTest:log-stream:111111111111_CloudTrail_us-east-2*",
                "arn:aws:logs:us-east-2:111111111111:log-group:CloudTrail/
DefaultLogGroupTest:log-stream:o-exampleorgid_*"
            ]
        }
    ]
}
```

For more information about organization trails, see Creating a trail for an organization.

Run the following command to apply the policy to the role.

```
aws iam put-role-policy --role-name role_name --policy-name cloudtrail-policy --policy-
document file://<path to role-policy-document>.json
```

Updating the trail

Update the trail with the log group and role information using the CloudTrail `update-trail` command.

```
aws cloudtrail update-trail --name trail_name --cloud-watch-logs-log-group-
arn log_group_arn --cloud-watch-logs-role-arn role_arn
```

For more information about the AWS CLI commands, see the AWS CloudTrail Command Line Reference.

Limitation

CloudWatch Logs and EventBridge each allow a maximum event size of 256 KB. Although most service events have a maximum size of 256 KB, some services still have events that are larger. CloudTrail does not send these events to CloudWatch Logs or EventBridge.

Starting with CloudTrail event version 1.05, events have a maximum size of 256 KB. This is to help prevent exploitation by malicious actors, and allow events to be consumed by other AWS services, such as CloudWatch Logs and EventBridge.

Creating CloudWatch alarms for CloudTrail events: examples

This topic describes how to configure alarms for CloudTrail events, and includes examples.

Topics

Prerequisites

Before you can use the examples in this topic, you must:

- Create a trail with the console or CLI.
- Create a log group, which you can do as part of creating a trail. For more information about creating a trail, see Creating a trail with the CloudTrail console.
- Specify or create an IAM role that grants CloudTrail the permissions to create a CloudWatch Logs log stream in the log group that you specify and to deliver CloudTrail events to that log stream. The default `CloudTrail_CloudWatchLogs_Role` does this for you.

For more information, see Sending events to CloudWatch Logs. Examples in this section are performed in the Amazon CloudWatch Logs console. For more information about how to create metric filters and alarms, see Creating metrics from log events using filters and Using Amazon CloudWatch alarms in the *Amazon CloudWatch User Guide*.

Create a metric filter and create an alarm

To create an alarm, you must first create a metric filter, and then configure an alarm based on the filter. The procedures are shown for all examples. For more information about syntax for metric filters and patterns for CloudTrail log events, see the JSON-related sections of Filter and pattern syntax in the *Amazon CloudWatch Logs User Guide*.

Example security group configuration changes

Follow this procedure to create an Amazon CloudWatch alarm that is triggered when configuration changes occur on security groups.

Create a metric filter

1. Open the CloudWatch console at https://console.aws.amazon.com/cloudwatch/.
2. In the navigation pane, under **Logs**, choose **Log groups**.
3. In the list of log groups, choose the log group that you created for your trail.
4. From the **Metric filters** or **Actions** menu, choose **Create metric filter**.
5. On the **Define pattern** page, in **Create filter pattern**, enter the following for **Filter pattern**.

```
{ ($.eventName = AuthorizeSecurityGroupIngress) || ($.eventName =
 AuthorizeSecurityGroupEgress) || ($.eventName = RevokeSecurityGroupIngress) ||
 ($.eventName = RevokeSecurityGroupEgress) || ($.eventName = CreateSecurityGroup)
 || ($.eventName = DeleteSecurityGroup) }
```

6. In **Test pattern**, leave defaults. Choose **Next**.
7. On the **Assign metric** page, for **Filter name**, enter **`SecurityGroupEvents`**.
8. In **Metric details**, turn on **Create new**, and then enter **`CloudTrailMetrics`** for **Metric namespace**.
9. For **Metric name**, type **`SecurityGroupEventCount`**.
10. For **Metric value**, type **1**.
11. Leave **Default value** blank.
12. Choose **Next**.
13. On the **Review and create** page, review your choices. Choose **Create metric filter** to create the filter, or choose **Edit** to go back and change values.

Create an alarm

After you create the metric filter, the CloudWatch Logs log group details page for your CloudTrail trail log group opens. Follow this procedure to create an alarm.

1. On the **Metric filters** tab, find the metric filter you created in the section called "Create a metric filter". Fill the check box for the metric filter. In the **Metric filters** bar, choose **Create alarm**.
2. For **Specify metric and conditions**, enter the following.

   a. For **Graph**, the line is set at **1** based on other settings you make when you create your alarm.

   b. For **Metric name**, keep the current metric name, **`SecurityGroupEventCount`**.

   c. For **Statistic**, keep the default, **Sum**.

   d. For **Period**, keep the default, **`5 minutes`**.

   e. In **Conditions**, for **Threshold type**, choose **Static**.

   f. For **Whenever *`metric_name`* is**, choose **Greater/Equal**.

   g. For the threshold value, enter **1**.

   h. In **Additional configuration**, leave defaults. Choose **Next**.

3. On the **Configure actions** page, choose **Notification**, and then choose **In alarm**, which indicates that the action is taken when the threshold of 1 change event in 5 minutes is crossed, and **SecurityGroupEventCount** is in an alarm state.

   a. For **Send a notification to the following SNS topic**, choose **Create new topic**.

   b. Enter **`SecurityGroupChanges_CloudWatch_Alarms_Topic`** as the name for the new Amazon SNS topic.

   c. In **Email endpoints that will receive the notification**, enter the email addresses of users whom you want to receive notifications if this alarm is raised. Separate email addresses with commas.

      Each email recipient will receive an email asking them to confirm that they want to be subscribed to the Amazon SNS topic.

   d. Choose **Create topic**.

4. For this example, skip the other action types. Choose **Next**.

5. On the **Add name and description** page, enter a friendly name for the alarm, and a description. For this example, enter **`Security group configuration changes`** for the name, and **`Raises alarms if security group configuration changes occur`** for the description. Choose **Next**.

6. On the **Preview and create** page, review your choices. Choose **Edit** to make changes, or choose **Create alarm** to create the alarm.

   After you create the alarm, CloudWatch opens the **Alarms** page. The alarm's **Actions** column shows **Pending confirmation** until all email recipients on the SNS topic have confirmed that they want to subscribe to SNS notifications.

Example AWS Management Console sign-in failures

Follow this procedure to create an Amazon CloudWatch alarm that is triggered when there are three or more AWS Management Console sign-in failures during a five minute period.

Create a metric filter

1. Open the CloudWatch console at https://console.aws.amazon.com/cloudwatch/.
2. In the navigation pane, under **Logs**, choose **Log groups**.
3. In the list of log groups, choose the log group that you created for your trail.

4. From the **Metric filters** or **Actions** menu, choose **Create metric filter**.
5. On the **Define pattern** page, in **Create filter pattern**, enter the following for **Filter pattern**.

   ```
   { ($.eventName = ConsoleLogin) && ($.errorMessage = "Failed authentication") }
   ```

6. In **Test pattern**, leave defaults. Choose **Next**.
7. On the **Assign metric** page, for **Filter name**, enter **`ConsoleSignInFailures`**.
8. In **Metric details**, turn on **Create new**, and then enter **`CloudTrailMetrics`** for **Metric namespace**.
9. For **Metric name**, type **`ConsoleSigninFailureCount`**.
10. For **Metric value**, type **1**.
11. Leave **Default value** blank.
12. Choose **Next**.
13. On the **Review and create** page, review your choices. Choose **Create metric filter** to create the filter, or choose **Edit** to go back and change values.

Create an alarm

After you create the metric filter, the CloudWatch Logs log group details page for your CloudTrail trail log group opens. Follow this procedure to create an alarm.

1. On the **Metric filters** tab, find the metric filter you created in the section called “Create a metric filter”. Fill the check box for the metric filter. In the **Metric filters** bar, choose **Create alarm**.
2. On the **Create Alarm** page, in **Specify metric and conditions**, enter the following.

   a. For **Graph**, the line is set at **3** based on other settings you make when you create your alarm.

   b. For **Metric name**, keep the current metric name, **`ConsoleSigninFailureCount`**.

   c. For **Statistic**, keep the default, **`Sum`**.

   d. For **Period**, keep the default, **`5 minutes`**.

   e. In **Conditions**, for **Threshold type**, choose **Static**.

   f. For **Whenever *`metric_name`* is**, choose **Greater/Equal**.

   g. For the threshold value, enter **3**.

    h. In **Additional configuration**, leave defaults. Choose **Next**.

3. On the **Configure actions** page, for **Notification**, choose **In alarm**, which indicates that the action is taken when the threshold of 3 change events in 5 minutes is crossed, and **ConsoleSigninFailureCount** is in an alarm state.

    a. For **Send a notification to the following SNS topic**, choose **Create new topic**.

    b. Enter **`ConsoleSignInFailures_CloudWatch_Alarms_Topic`** as the name for the new Amazon SNS topic.

    c. In **Email endpoints that will receive the notification**, enter the email addresses of users whom you want to receive notifications if this alarm is raised. Separate email addresses with commas.

        Each email recipient will receive an email asking them to confirm that they want to be subscribed to the Amazon SNS topic.

    d. Choose **Create topic**.

4. For this example, skip the other action types. Choose **Next**.

5. On the **Add name and description** page, enter a friendly name for the alarm, and a description. For this example, enter **`Console sign-in failures`** for the name, and **`Raises alarms if more than 3 console sign-in failures occur in 5 minutes`** for the description. Choose **Next**.

6. On the **Preview and create** page, review your choices. Choose **Edit** to make changes, or choose **Create alarm** to create the alarm.

    After you create the alarm, CloudWatch opens the **Alarms** page. The alarm's **Actions** column shows **Pending confirmation** until all email recipients on the SNS topic have confirmed that they want to subscribe to SNS notifications.

Example: IAM policy changes

Follow this procedure to create an Amazon CloudWatch alarm that is triggered when an API call is made to change an IAM policy.

Create a metric filter

1. Open the CloudWatch console at https://console.aws.amazon.com/cloudwatch/.

2. In the navigation pane, choose **Logs**.

3. In the list of log groups, choose the log group that you created for your trail.
4. Choose **Actions**, and then choose **Create metric filter**.
5. On the **Define pattern** page, in **Create filter pattern**, enter the following for **Filter pattern**.

```
{($.eventName=DeleteGroupPolicy)||($.eventName=DeleteRolePolicy)||
($.eventName=DeleteUserPolicy)||($.eventName=PutGroupPolicy)||
($.eventName=PutRolePolicy)||($.eventName=PutUserPolicy)||
($.eventName=CreatePolicy)||($.eventName=DeletePolicy)||
($.eventName=CreatePolicyVersion)||($.eventName=DeletePolicyVersion)||
($.eventName=AttachRolePolicy)||($.eventName=DetachRolePolicy)||
($.eventName=AttachUserPolicy)||($.eventName=DetachUserPolicy)||
($.eventName=AttachGroupPolicy)||($.eventName=DetachGroupPolicy)}
```

6. In **Test pattern**, leave defaults. Choose **Next**.
7. On the **Assign metric** page, for **Filter name**, enter **`IAMPolicyChanges`**.
8. In **Metric details**, turn on **Create new**, and then enter **`CloudTrailMetrics`** for **Metric namespace**.
9. For **Metric name**, type **`IAMPolicyEventCount`**.
10. For **Metric value**, type **1**.
11. Leave **Default value** blank.
12. Choose **Next**.
13. On the **Review and create** page, review your choices. Choose **Create metric filter** to create the filter, or choose **Edit** to go back and change values.

Create an alarm

After you create the metric filter, the CloudWatch Logs log group details page for your CloudTrail trail log group opens. Follow this procedure to create an alarm.

1. On the **Metric filters** tab, find the metric filter you created in the section called “Create a metric filter”. Fill the check box for the metric filter. In the **Metric filters** bar, choose **Create alarm**.
2. On the **Create Alarm** page, in **Specify metric and conditions**, enter the following.

   a. For **Graph**, the line is set at **1** based on other settings you make when you create your alarm.

   b. For **Metric name**, keep the current metric name, **`IAMPolicyEventCount`**.

c. For **Statistic**, keep the default, **`Sum`**.

d. For **Period**, keep the default, **`5 minutes`**.

e. In **Conditions**, for **Threshold type**, choose **Static**.

f. For **Whenever *`metric_name`* is**, choose **Greater/Equal**.

g. For the threshold value, enter **1**.

h. In **Additional configuration**, leave defaults. Choose **Next**.

i.

3. On the **Configure actions** page, for **Notification**, choose **In alarm**, which indicates that the action is taken when the threshold of 1 change event in 5 minutes is crossed, and **IAMPolicyEventCount** is in an alarm state.

   a. For **Send a notification to the following SNS topic**, choose **Create new topic**.

   b. Enter **`IAM_Policy_Changes_CloudWatch_Alarms_Topic`** as the name for the new Amazon SNS topic.

   c. In **Email endpoints that will receive the notification**, enter the email addresses of users whom you want to receive notifications if this alarm is raised. Separate email addresses with commas.

      Each email recipient will receive an email asking them to confirm that they want to be subscribed to the Amazon SNS topic.

   d. Choose **Create topic**.

4. For this example, skip the other action types. Choose **Next**.

5. On the **Add name and description** page, enter a friendly name for the alarm, and a description. For this example, enter **`IAM Policy Changes`** for the name, and **`Raises alarms if IAM policy changes occur`** for the description. Choose **Next**.

6. On the **Preview and create** page, review your choices. Choose **Edit** to make changes, or choose **Create alarm** to create the alarm.

   After you create the alarm, CloudWatch opens the **Alarms** page. The alarm's **Actions** column shows **Pending confirmation** until all email recipients on the SNS topic have confirmed that they want to subscribe to SNS notifications.

Configuring notifications for CloudWatch Logs alarms

You can configure CloudWatch Logs to send a notification whenever an alarm is triggered for CloudTrail. Doing so enables you to respond quickly to critical operational events captured in CloudTrail events and detected by CloudWatch Logs. CloudWatch uses Amazon Simple Notification Service (SNS) to send email. For more information, see Setting up Amazon SNS notifications in the *CloudWatch User Guide*.

Stopping CloudTrail from sending events to CloudWatch Logs

You can stop sending AWS CloudTrail events to Amazon CloudWatch Logs by updating a trail to disable CloudWatch Logs settings.

Stop sending events to CloudWatch Logs (console)

To stop sending CloudTrail events to CloudWatch Logs

1. Sign in to the AWS Management Console and open the CloudTrail console at https://console.aws.amazon.com/cloudtrail/.
2. In the navigation pane, choose **Trails**.
3. Choose the name of the trail for which you want to disable CloudWatch Logs integration.
4. In **CloudWatch Logs**, choose **Edit**.
5. Clear the **Enabled** check box.
6. Choose **Save changes**.

Stop sending events to CloudWatch Logs (CLI)

You can remove the CloudWatch Logs log group as a delivery endpoint by running the **update-trail** command. The following command clears the log group and role from the trail configuration by replacing the values for the log group ARN and CloudWatch Logs role ARN with empty values.

```
aws cloudtrail update-trail --name trail_name --cloud-watch-logs-log-group-arn="" --
cloud-watch-logs-role-arn=""
```

CloudWatch log group and log stream naming for CloudTrail

Amazon CloudWatch will display the log group that you created for CloudTrail events alongside any other log groups you have in a Region. We recommend that you use a log group name that helps you easily distinguish the log group from others. For example, **`CloudTrail/logs`**.

Follow these guidelines when naming a log group:

- Log group names must be unique within a Region for an AWS account.
- Log group names can be between 1 and 512 characters long.
- Log group names consist of the following characters: a-z, A-Z, 0-9, '_' (underscore), '-' (hyphen), '/' (forward slash), '.' (period), and '#' (number sign).

When CloudTrail creates the log stream for the log group, it names the log stream according to the following format: *`account_ID`*_CloudTrail_*`trail_region`*.

> **Note**
>
> If the volume of CloudTrail logs is large, multiple log streams may be created to deliver log data to your log group. When there are multiple log streams, CloudTrail names each log stream according to the following format: *`account_ID`*_CloudTrail_*`trail_region`*_*`number`*.

For more information about CloudWatch log groups, see Working with log groups and log streams in the *Amazon CloudWatch Logs User Guide* and CreateLogGroup in the *Amazon CloudWatch Logs API Reference*.

Role policy document for CloudTrail to use CloudWatch Logs for monitoring

This section describes the permissions policy required for the CloudTrail role to send log events to CloudWatch Logs. You can attach a policy document to a role when you configure CloudTrail to send events, as described in Sending events to CloudWatch Logs. You can also create a role using IAM. For more information, see Creating a role to delegate permissions to an AWS service or Creating an IAM role (AWS CLI).

The following example policy document contains the permissions required to create a CloudWatch log stream in the log group that you specify and to deliver CloudTrail events to that log stream in the US East (Ohio) Region. (This is the default policy for the default IAM role `CloudTrail_CloudWatchLogs_Role`.)

> **Note**
>
> Confused deputy prevention is not applicable to the role policy for CloudWatch Logs monitoring. The role policy doesn't support the use of `aws:SourceArn` and `aws:SourceAccount`.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {

      "Sid": "AWSCloudTrailCreateLogStream2014110",
      "Effect": "Allow",
      "Action": [
        "logs:CreateLogStream"
      ],
      "Resource": [
        "arn:aws:logs:us-east-2:accountID:log-group:log_group_name:log-
stream:CloudTrail_log_stream_name_prefix*"
      ]

    },
    {
      "Sid": "AWSCloudTrailPutLogEvents20141101",
      "Effect": "Allow",
      "Action": [
        "logs:PutLogEvents"
      ],
      "Resource": [
        "arn:aws:logs:us-east-2:accountID:log-group:log_group_name:log-
stream:CloudTrail_log_stream_name_prefix*"
      ]
    }
  ]
}
```

If you're creating a policy that might be used for organization trails as well, you will need to modify it from the default policy created for the role. For example, the following policy grants CloudTrail the permissions required to create a CloudWatch Logs log stream in the log group you specify as the value of *log_group_name*, and to deliver CloudTrail events to that log stream for both trails in the AWS account 111111111111 and for organization trails created in the 111111111111 account that are applied to the AWS Organizations organization with the ID of *o-exampleorgid*:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AWSCloudTrailCreateLogStream20141101",
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogStream"
            ],
            "Resource": [
                "arn:aws:logs:us-east-2:111111111111:log-group:log_group_name:log-
stream:111111111111_CloudTrail_us-east-2*",
                "arn:aws:logs:us-east-2:111111111111:log-group:log_group_name:log-
stream:o-exampleorgid_*"
            ]
        },
        {
            "Sid": "AWSCloudTrailPutLogEvents20141101",
            "Effect": "Allow",
            "Action": [
                "logs:PutLogEvents"
            ],
            "Resource": [
                "arn:aws:logs:us-east-2:111111111111:log-group:log_group_name:log-
stream:111111111111_CloudTrail_us-east-2*",
                "arn:aws:logs:us-east-2:111111111111:log-group:log_group_name:log-
stream:o-exampleorgid_*"
            ]
        }
    ]
}
```

For more information about organization trails, see Creating a trail for an organization.

Receiving CloudTrail log files from multiple accounts

You can have CloudTrail deliver log files from multiple AWS accounts into a single Amazon S3 bucket. For example, you have four AWS accounts with account IDs 111111111111, 222222222222, 333333333333, and 444444444444, and you want to configure CloudTrail to deliver log files from all four of these accounts to a bucket belonging to account 111111111111. To accomplish this, complete the following steps in order:

1. Create a trail in the account where the destination bucket will belong (111111111111 in this example). Do not create a trail for any other accounts yet.

   For instructions, see Creating a trail in the console.
2. Update the bucket policy on your destination bucket to grant cross-account permissions to CloudTrail.

   For instructions, see Setting bucket policy for multiple accounts.
3. Create a trail in the other accounts (222222222222, 333333333333, and 444444444444 in this example) for which you want to log activity. When you create the trail in each account, specify the Amazon S3 bucket belonging to the account that you specified in step 1 (111111111111 in this example). For instructions, see Create trails in additional accounts.

   > **Note**
   >
   > If you choose to enable SSE-KMS encryption, the KMS key policy must allow CloudTrail to use the key to encrypt your log files, and allow the users you specify to read log files in unencrypted form. For information about manually editing the key policy, see Configure AWS KMS key policies for CloudTrail.

Redacting bucket owner account IDs for data events called by other accounts

Historically, if CloudTrail data events were enabled in the AWS account of an Amazon S3 data event API caller, CloudTrail showed the account ID of the S3 bucket owner in the data event (such as `PutObject`). This occurred even if the bucket owner account did not have S3 data events enabled.

Now, CloudTrail removes the account ID of the S3 bucket owner in the `resources` block if both of the following conditions are met:

- The data event API call is from a different AWS account than the Amazon S3 bucket owner.
- The API caller received an `AccessDenied` error that was only for the caller account.

The owner of the resource on which the API call was made still receives the full event.

The following event record snippets are an example of the expected behavior. In the `Historic` snippet, the account ID 123456789012 of the S3 bucket owner is shown to an API caller from a different account. In the example of current behavior, the account ID of the bucket owner is not shown.

```
# Historic

"resources": [
    {
        "type": "AWS::S3::Object",
        "ARNPrefix": "arn:aws:s3:::DOC-EXAMPLE-BUCKET2/"
    },
    {
        "accountId": "123456789012",
        "type": "AWS::S3::Bucket",
        "ARN": "arn:aws:s3:::DOC-EXAMPLE-BUCKET2"
    }
]
```

The following is the current behavior.

```
# Current

"resources": [
    {
        "type": "AWS::S3::Object",
        "ARNPrefix": "arn:aws:s3:::DOC-EXAMPLE-BUCKET2/"
    },
    {
        "accountId": "",
        "type": "AWS::S3::Bucket",
        "ARN": "arn:aws:s3:::DOC-EXAMPLE-BUCKET2"
    }
]
```

Topics

- Setting bucket policy for multiple accounts
- Create trails in additional accounts

Setting bucket policy for multiple accounts

For a bucket to receive log files from multiple accounts, its bucket policy must grant CloudTrail permission to write log files from all the accounts you specify. This means that you must modify the bucket policy on your destination bucket to grant CloudTrail permission to write log files from each specified account.

> **Note**
>
> For security reasons, unauthorized users cannot create a trail that includes `AWSLogs/` as the `S3KeyPrefix` parameter.

To modify bucket permissions so that files can be received from multiple accounts

1. Sign in to the AWS Management Console using the account that owns the bucket (111111111111 in this example) and open the Amazon S3 console.
2. Choose the bucket where CloudTrail delivers your log files and then choose **Permissions**.
3. For **Bucket policy**, choose **Edit**.
4. Modify the existing policy to add a line for each additional account whose log files you want delivered to this bucket. See the following example policy and note the underlined `Resource` line specifying a second account ID. As a security best practice, add an `aws:SourceArn` condition key to the Amazon S3 bucket policy. This helps prevent unauthorized access to your S3 bucket. If you have existing trails, be sure to add one or more condition keys.

   > **Note**
   >
   > An AWS account ID is a twelve-digit number, including leading zeros.

   ```
   {
     "Version": "2012-10-17",
     "Statement": [
       {
   ```

```
   "Sid": "AWSCloudTrailAclCheck20131101",
   "Effect": "Allow",
   "Principal": {
    "Service": "cloudtrail.amazonaws.com"
   },
   "Action": "s3:GetBucketAcl",
   "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
   "Condition": {
     "StringEquals": {
      "aws:SourceArn": [
       "arn:aws:cloudtrail:region:111111111111:trail/primaryTrailName",
       "arn:aws:cloudtrail:region:222222222222:trail/secondaryTrailName"
      ]
     }
   }
  },
  {
   "Sid": "AWSCloudTrailWrite20131101",
   "Effect": "Allow",
   "Principal": {
    "Service": "cloudtrail.amazonaws.com"
   },
   "Action": "s3:PutObject",
   "Resource": [
    "arn:aws:s3:::DOC-EXAMPLE-
BUCKET/optionalLogFilePrefix/AWSLogs/111111111111/*",
    "arn:aws:s3:::DOC-EXAMPLE-
BUCKET/optionalLogFilePrefix/AWSLogs/222222222222/*"
   ],
   "Condition": {
    "StringEquals": {
     "aws:SourceArn": [
      "arn:aws:cloudtrail:region:111111111111:trail/primaryTrailName",
      "arn:aws:cloudtrail:region:222222222222:trail/secondaryTrailName"
     ],
     "s3:x-amz-acl": "bucket-owner-full-control"
    }
   }
  }
 ]
}
```

Create trails in additional accounts

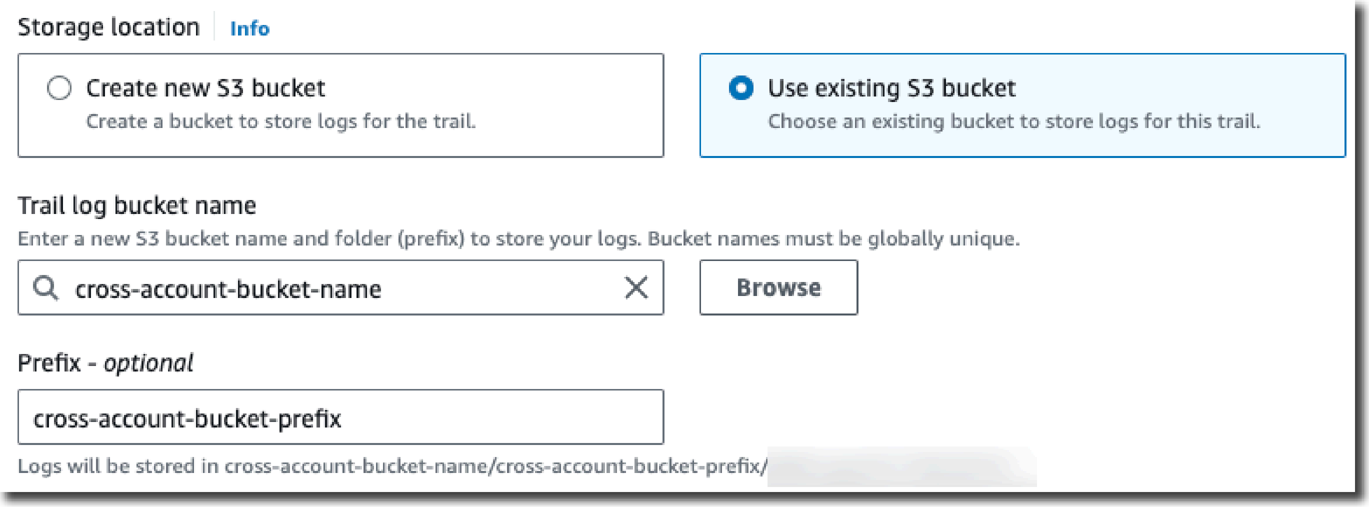
You can use the console or the AWS CLI to create trails in additional AWS accounts and aggregate their log files to one Amazon S3 bucket. Alternatively, you could create an organization trail to log all AWS accounts that are part of an organization in AWS Organizations. For more information, see Creating a trail for an organization.

Using the console to create trails in additional AWS accounts

You can use the CloudTrail console to create trails in additional accounts.

1. Sign in to AWS Management Console with the account for which you want to create a trail. Follow the steps in Creating a trail in the console to create a trail using the console.
2. For **Storage location**, choose **Use existing S3 bucket**. Use the text box to enter the name of the bucket you're using to store log files across accounts.

   > **Note**
   >
   > The bucket policy must grant CloudTrail permission to write to it. For information about manually editing the bucket policy, see Setting bucket policy for multiple accounts.



Storage location Info

○ Create new S3 bucket
Create a bucket to store logs for the trail.

● Use existing S3 bucket
Choose an existing bucket to store logs for this trail.

Trail log bucket name
Enter a new S3 bucket name and folder (prefix) to store your logs. Bucket names must be globally unique.

cross-account-bucket-name

Prefix - *optional*

cross-account-bucket-prefix

Logs will be stored in cross-account-bucket-name/cross-account-bucket-prefix/

3. For **Prefix**, enter the prefix you are using to store log files across accounts. If you choose to use a prefix that is different from what you specified in your bucket policy, you must edit the

bucket policy on your destination bucket to allow CloudTrail to write log files to your bucket using this new prefix.

Using the CLI to create a trail in additional AWS accounts

You can use the AWS command line tools to create trails in additional accounts and aggregate their log files to one Amazon S3 bucket. For more information about these tools, see cloudtrail in the *AWS CLI Command Reference*.

Create a trail by using the **create-trail** command, specifying the following:

- `--name` specifies the name of the trail.
- `--s3-bucket-name` specifies the Amazon S3 bucket you are using to store log files across accounts.
- `--s3-prefix` specifies a prefix for the log file delivery path (optional).
- `--is-multi-region-trail` specifies that this trail will log events in all AWS Regions in the partition in which you are working.

You can create one trail for each Region in which an account is running AWS resources.

The following example command shows how to create a trail for your additional accounts by using the AWS CLI. To have log files for these account delivered to the bucket you created in your first account (111111111111 in this example), specify the bucket name in the `--s3-bucket-name` option. Amazon S3 bucket names are globally unique.

```
aws cloudtrail create-trail --name my-trail --s3-bucket-name DOC-EXAMPLE-BUCKET --is-
multi-region-trail
```

When you run the command, you will see output similar to the following:

```
{
    "IncludeGlobalServiceEvents": true,
    "Name": "AWSCloudTrailExample",
    "TrailARN": "arn:aws:cloudtrail:us-east-2:222222222222:trail/my-trail",
    "LogFileValidationEnabled": false,
    "IsMultiRegionTrail": true,
    "IsOrganizationTrail": false,
    "S3BucketName": "DOC-EXAMPLE-BUCKET"
}
```

For more information about using CloudTrail from the AWS command line tools, see the CloudTrail command line reference.

Sharing CloudTrail log files between AWS accounts

This section explains how to share CloudTrail log files between multiple AWS accounts. The approach you use to share logs between AWS accounts depends on the configuration of your S3 bucket. These are the options for sharing log files:

- **Bucket owner enforced** – S3 Object Ownership is an Amazon S3 bucket-level setting that you can use to control ownership of objects uploaded to your bucket and to disable or enable access control lists (ACLs). By default, Object Ownership is set to the **Bucket owner enforced** setting and all ACLs are disabled. When ACLs are disabled, the bucket owner owns all the objects in the bucket and manages access to data exclusively using access management policies. When the **Bucket owner enforced** option is set, access is managed through the bucket policy, eliminating the need for users to assume a role.
- **Assume a role to share log files** – If you haven't chosen the **Bucket owner enforced** setting, users will need to assume a role to access the log files in your S3 bucket.

Share log files between accounts by assuming a role

Note

This section applies only to Amazon S3 buckets that are not using the **Bucket owner enforced** setting.

This section explains how to share CloudTrail log files between multiple AWS accounts by assuming a role and describes the scenarios for sharing log files.

- **Scenario 1**: Grant read-only access to the accounts that generated the log files that have been placed into your Amazon S3 bucket.
- **Scenario 2**: Grant access to all of the log files in your Amazon S3 bucket to a third-party account that can analyze the log files for you.

To grant read-only access to the log files in your Amazon S3 bucket

1. Create an IAM role for each account you want to share log files with. You must be an administrator to grant permission.

   When you create the role, do the following:

   - Choose the **Another AWS account** option.
   - Enter the twelve-digit account ID of the account to be granted access.
   - Check the **Require MFA** box if you want the user to provide multi-factor authentication before assuming the role.
   - Choose the **AmazonS3ReadOnlyAccess** policy.

   > **Note**
   >
   > By default, the **AmazonS3ReadOnlyAccess** policy grants retrieval and list rights to all Amazon S3 buckets within your account.

   For details about permissions management for IAM roles, see IAM roles in the IAM User Guide.

2. Create an access policy that grants read-only access to the account you want to share the log files with.
3. Instruct each account to assume a role to retrieve the log files.

To grant read-only access to the log files with a third-party account

1. Create an IAM role for the third-party account you want to share log files with. You must be an administrator to grant permission.

   When you create the role, do the following:

   - Choose the **Another AWS account** option.
   - Enter the twelve-digit account ID of the account to be granted access.
   - Enter an external ID that provides additional control over who can assume the role. For more information, see How to Use an External ID When Granting Access to Your AWS Resources to a Third Party in the *IAM User Guide*.
   - Choose the **AmazonS3ReadOnlyAccess** policy.

> **Note**
>
> By default, the **AmazonS3ReadOnlyAccess** policy grants retrieval and list rights to all Amazon S3 buckets within your account.

2. Create an access policy that grants read-only access to the third-party account you want to share the log files with.
3. Instruct the third-party account to assume a role to retrieve the log files.

The following sections provide more detail about these steps.

Topics

- Creating an access policy to grant access to accounts you own
- Creating an access policy to grant access to a third party
- Assuming a role
- Stop sharing CloudTrail log files between AWS accounts

Creating an access policy to grant access to accounts you own

As the Amazon S3 bucket owner, you have full control over the Amazon S3 bucket to which CloudTrail writes log files for the other accounts. You want to share each business unit's log files back to business unit that created them. But, you don't want a unit to be able to read any other unit's log files.

For example, to share account B's log files with account B but not with account C, you must create a new IAM role in your account that specifies that account B is a trusted account. This role trust policy specifies that account B is trusted to assume the role created by your account, and should look like the following example. The trust policy is automatically created if you create the role by using the console. If you use the SDK to create the role, you must supply the trust policy as a parameter to the `CreateRole` API. If you use the CLI to create the role, you must specify the trust policy in the `create-role` CLI command.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::account-B-id:root"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

You must also create an access policy to specify that account B can read from only the location to which B wrote its log files. The access policy will look something like the following. Note that the **Resource** ARN includes the twelve-digit account ID for account B, and the prefix you specified, if any, when you turned on CloudTrail for account B during the aggregation process. For more information about specifying a prefix, see Create trails in additional accounts.

Important

You must ensure that the prefix in the access policy is exactly the same as the prefix that you specified when you turned on CloudTrail for account B. If it is not, then you must edit the IAM role access policy in your account to incorporate the actual prefix for account B. If the prefix in the role access policy is not exactly the same as the prefix you specified when you turned on CloudTrail in account B, then account B will not be able to access its log files.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:Get*",
        "s3:List*"
      ],
      "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/prefix/AWSLogs/account-B-id/*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:Get*",
        "s3:List*"
      ],
      "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET"
    }
  ]
}
```

Use the preceding process for any additional accounts.

After you create roles for each account and specify the appropriate trust and access policies, and after an IAM user in each account has been granted access by the administrator of that account, an IAM user in accounts B or C can programmatically assume the role.

For more information, see Assuming a role.

Creating an access policy to grant access to a third party

You must create a separate IAM role for a third-party account. When you create the role, AWS automatically creates the trust relationship, which specifies that third-party account will be trusted to assume the role. The access policy for the role specifies what actions that account can take. For more information about creating roles, see Create an IAM role.

For example, the trust relationship created by AWS specifies that the third-party account (account Z in this example) is trusted to assume the role that you've created. The following is an example trust policy:

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Sid": "",
        "Effect": "Allow",
        "Principal": {"AWS": "arn:aws:iam::account-Z-id:root"},
        "Action": "sts:AssumeRole"
    }]
}
```

If you specified an external ID when you created the role for the third-party account, your access policy contains an added `Condition` element that tests the unique ID assigned by that account.

The test is performed when the role is assumed. The following example access policy has a Condition element.

For more information, see How to use an external ID when granting access to your AWS resources to a third party in the *IAM User Guide*.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Sid": "",
        "Effect": "Allow",
        "Principal": {"AWS": "arn:aws:iam::account-Z-id:root"},
        "Action": "sts:AssumeRole",
        "Condition": {"StringEquals": {"sts:ExternalId": "external-ID-issued-by-account-Z"}}
    }]
}
```

You must also create an access policy for your account to specify that the third-party account can read all logs from the Amazon S3 bucket. The access policy should look something like the following example. The wild card (*) at the end of the Resource value indicates that the third-party account can access any log file in the S3 bucket to which it has been granted access.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET"
        }
    ]
}
```

After you create a role for the third-party account and specify the appropriate trust relationship and access policy, an IAM user in the third-party account must programmatically assume the role to be able to read log files from the bucket. For more information, see Assuming a role.

Assuming a role

You must designate a separate IAM user to assume each role you create in each account. You must then ensure that each IAM user has appropriate permissions.

IAM users and roles

After you create the necessary roles and policies, you must designate an IAM user in each of the account with which you want to share files. Each IAM user programmatically assumes the appropriate role to access the log files. When a user assumes a role, AWS returns temporary security credentials to that user. They can then make requests to list, retrieve, copy, or delete log files depending on the permissions granted by the access policy associated with the role.

For more information about working with IAM identities, see IAM Identities (users, user groups, and roles).

The primary difference in the access policy that you create for each IAM role in each scenario.

- In scenario 1, the access policy limits each account to reading only its own log files. For more information, see Creating an access policy to grant access to accounts you own.
- In scenario 2, the access policy allows a third-party it to read all the log files that are aggregated in the Amazon S3 bucket. For more information, see Creating an access policy to grant access to a third party.

Creating permissions policies for IAM users

To perform the actions permitted by a role, the IAM user must have permission to call the AWS STS `AssumeRole` API. You must edit the policy for each user to grant them the appropriate permissions. To do this, you set a **Resource** element in the policy that you attach to the IAM user. The following example shows a policy for an IAM user in another account that allows that user to assume a role named `Test` created earlier by Account A.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["sts:AssumeRole"],
      "Resource": "arn:aws:iam::account-A-id:role/Test"
    }
  ]
}
```

To edit a customer managed policy (console)

1. Sign in to the AWS Management Console and open the IAM console at https://console.aws.amazon.com/iam/.
2. In the navigation pane, choose **Policies**.
3. In the list of policies, choose the policy name of the policy to edit. You can use the search box to filter the list of policies.
4. Choose the **Permissions** tab, and then choose **Edit**.
5. Do one of the following:

   - Choose the **Visual** option to change your policy without understanding JSON syntax. You can make changes to the service, actions, resources, or optional conditions for each permission block in your policy. You can also import a policy to add additional permissions to the bottom of your policy. When you are finished making changes, choose **Next** to continue.
   - Choose the **JSON** option to modify your policy by typing or pasting text in the JSON text box. You can also import a policy to add additional permissions to the bottom of your policy. Resolve any security warnings, errors, or general warnings generated during policy validation, and then choose **Next**.

   > **Note**
   >
   > You can switch between the **Visual** and **JSON** editor options any time. However, if you make changes or choose **Next** in the **Visual** editor, IAM might restructure your policy to optimize it for the visual editor. For more information, see Policy restructuring in the *IAM User Guide*.

6. On the **Review and save** page, review **Permissions defined in this policy** and then choose **Save changes** to save your work.
7. If the managed policy already has the maximum of five versions, choosing **Save changes** displays a dialog box. To save your new version, the oldest non-default version of the policy is removed and replaced with this new version. Optionally, you can set the new version as the default policy version.

   Choose **Save changes** to save your new policy version.

Calling AssumeRole

A user can assume a role by creating an application that calls the AWS STS `AssumeRole` API and passes the role session name, the Amazon Resource Number (ARN) of the role to assume, and an optional external ID. The role session name is defined by the account that created the role to assume. The external ID, if any, is defined by the third-party account and passed to owning account for inclusion during role creation. For more information, see How to Use an External ID When Granting Access to Your AWS Resources to a Third Party in the *IAM User Guide*. You can retrieve the ARN from the Account A by opening the IAM console.

To find the ARN Value in Account A with the IAM console

1. Choose **Roles**
2. Choose the role you want to examine.
3. Look for the **Role ARN** in the **Summary** section.

The AssumeRole API returns temporary credentials to use to access resources in owning account. In this example, the resources you want to access are the Amazon S3 bucket and the log files that the bucket contains. The temporary credentials have the permissions that you defined in the role access policy.

The following Python example (using the AWS SDK for Python (Boto)) shows how to call `AssumeRole` and how to use the temporary security credentials returned to list all Amazon S3 buckets controlled by Account A.

```
def list_buckets_from_assumed_role(user_key, assume_role_arn, session_name):
    """
    Assumes a role that grants permission to list the Amazon S3 buckets in the account.
    Uses the temporary credentials from the role to list the buckets that are owned
    by the assumed role's account.

    :param user_key: The access key of a user that has permission to assume the role.
    :param assume_role_arn: The Amazon Resource Name (ARN) of the role that
                            grants access to list the other account's buckets.
    :param session_name: The name of the STS session.
    """
    sts_client = boto3.client(
        "sts", aws_access_key_id=user_key.id, aws_secret_access_key=user_key.secret
    )
    try:
        response = sts_client.assume_role(
            RoleArn=assume_role_arn, RoleSessionName=session_name
        )
        temp_credentials = response["Credentials"]
        print(f"Assumed role {assume_role_arn} and got temporary credentials.")
    except ClientError as error:
        print(
            f"Couldn't assume role {assume_role_arn}. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise

    # Create an S3 resource that can access the account with the temporary credentials.
    s3_resource = boto3.resource(
        "s3",
        aws_access_key_id=temp_credentials["AccessKeyId"],
        aws_secret_access_key=temp_credentials["SecretAccessKey"],
        aws_session_token=temp_credentials["SessionToken"],
    )
    print(f"Listing buckets for the assumed role's account:")
    try:
        for bucket in s3_resource.buckets.all():
            print(bucket.name)
    except ClientError as error:
        print(
            f"Couldn't list buckets for the account. Here's why: "
            f"{error.response['Error']['Message']}"
        )
        raise
```

Stop sharing CloudTrail log files between AWS accounts

To stop sharing log files to another AWS account, delete the role that you created for that account. For information about how to delete a role, see Deleting roles or instance profiles.

Validating CloudTrail log file integrity

To determine whether a log file was modified, deleted, or unchanged after CloudTrail delivered it, you can use CloudTrail log file integrity validation. This feature is built using industry standard algorithms: SHA-256 for hashing and SHA-256 with RSA for digital signing. This makes it computationally infeasible to modify, delete or forge CloudTrail log files without detection. You can use the AWS CLI to validate the files in the location where CloudTrail delivered them.

Why use it?

Validated log files are invaluable in security and forensic investigations. For example, a validated log file enables you to assert positively that the log file itself has not changed, or that particular user credentials performed specific API activity. The CloudTrail log file integrity validation process also lets you know if a log file has been deleted or changed, or assert positively that no log files were delivered to your account during a given period of time.

How it works

When you enable log file integrity validation, CloudTrail creates a hash for every log file that it delivers. Every hour, CloudTrail also creates and delivers a file that references the log files for the last hour and contains a hash of each. This file is called a digest file. CloudTrail signs each digest file using the private key of a public and private key pair. After delivery, you can use the public key to validate the digest file. CloudTrail uses different key pairs for each AWS Region.

The digest files are delivered to the same Amazon S3 bucket associated with your trail as your CloudTrail log files. If your log files are delivered from all Regions or from multiple accounts into a single Amazon S3 bucket, CloudTrail will deliver the digest files from those Regions and accounts into the same bucket.

The digest files are put into a folder separate from the log files. This separation of digest files and log files enables you to enforce granular security policies and permits existing log processing solutions to continue to operate without modification. Each digest file also contains the digital signature of the previous digest file if one exists. The signature for the current digest file is in the metadata properties of the digest file Amazon S3 object. For more information about digest file contents, see CloudTrail digest file structure.

Storing log and digest files

You can store the CloudTrail log files and digest files in Amazon S3 or S3 Glacier securely, durably and inexpensively for an indefinite period of time. To enhance the security of the digest files stored in Amazon S3, you can use Amazon S3 MFA Delete.

Enabling validation and validating files

To enable log file integrity validation, you can use the AWS Management Console, the AWS CLI, or CloudTrail API. Enabling log file integrity validation allows CloudTrail to deliver digest log files to your Amazon S3 bucket, but does not validate the integrity of the files. For more information, see Enabling log file integrity validation for CloudTrail.

To validate the integrity of CloudTrail log files, you can use the AWS CLI or create your own solution. The AWS CLI will validate files in the location where CloudTrail delivered them. If you want to validate logs that you have moved to a different location, either in Amazon S3 or elsewhere, you can create your own validation tools.

For information on validating logs by using the AWS CLI, see Validating CloudTrail log file integrity with the AWS CLI. For information on developing custom implementations of CloudTrail log file validation, see Custom implementations of CloudTrail log file integrity validation.

Enabling log file integrity validation for CloudTrail

You can enable log file integrity validation by using the AWS Management Console, AWS Command Line Interface (AWS CLI), or CloudTrail API. CloudTrail starts delivering digest files in about an hour.

AWS Management Console

To enable log file integrity validation with the CloudTrail console, choose **Yes** for the **Enable log file validation** option when you create or update a trail. By default, this feature is enabled for new trails. For more information, see Creating and updating a trail with the console.

AWS CLI

To enable log file integrity validation with the AWS CLI, use the `--enable-log-file-validation` option with the create-trail or update-trail commands. To disable log file integrity validation, use the `--no-enable-log-file-validation` option.

Example

The following `update-trail` command enables log file validation and starts delivering digest files to the Amazon S3 bucket for the specified trail.

```
aws cloudtrail update-trail --name your-trail-name --enable-log-file-validation
```

CloudTrail API

To enable log file integrity validation with the CloudTrail API, set the `EnableLogFileValidation` request parameter to `true` when calling `CreateTrail` or `UpdateTrail`.

For more information, see CreateTrail and UpdateTrail in the AWS CloudTrail API Reference.

Validating CloudTrail log file integrity with the AWS CLI

To validate logs with the AWS Command Line Interface, use the CloudTrail `validate-logs` command. The command uses the digest files delivered to your Amazon S3 bucket to perform the validation. For information about digest files, see CloudTrail digest file structure.

The AWS CLI allows you to detect the following types of changes:

- Modification or deletion of CloudTrail log files
- Modification or deletion of CloudTrail digest files
- Modification or deletion of both of the above

> **Note**
>
> The AWS CLI validates only log files that are referenced by digest files. For more information, see Checking whether a particular file was delivered by CloudTrail.

Prerequisites

To validate log file integrity with the AWS CLI, the following conditions must be met:

- You must have online connectivity to AWS.

- You must have read access to the Amazon S3 bucket that contains the digest and log files.
- The digest and log files must not have been moved from the original Amazon S3 location where CloudTrail delivered them.

Note

Log files that have been downloaded to local disk cannot be validated with the AWS CLI. For guidance on creating your own tools for validation, see Custom implementations of CloudTrail log file integrity validation.

validate-logs

Syntax

The following is the syntax for `validate-logs`. Optional parameters are shown in brackets.

```
aws cloudtrail validate-logs --trail-arn <trailARN> --start-time <start-time> [--end-time <end-time>] [--s3-bucket <DOC-EXAMPLE-BUCKET>] [--s3-prefix <prefix>] [--account-id <account-id>] [--verbose]
```

Note

The `validate-logs` command is Region specific. You must specify the `--region` global option to validate logs for a specific AWS Region.

Options

The following are the command-line options for `validate-logs`. The `--trail-arn` and `--start-time` options are required. The `--account-id` option is additionally required for organizational trails.

`--start-time`

Specifies that log files delivered on or after the specified UTC timestamp value will be validated. Example: `2015-01-08T05:21:42Z`.

`--end-time`

Optionally specifies that log files delivered on or before the specified UTC timestamp value will be validated. The default value is the current UTC time (`Date.now()`). Example: `2015-01-08T12:31:41Z`.

> **Note**
>
> For the time range specified, the `validate-logs` command checks only the log files that are referenced in their corresponding digest files. No other log files in the Amazon S3 bucket are checked. For more information, see Checking whether a particular file was delivered by CloudTrail.

`--s3-bucket`

Optionally specifies the Amazon S3 bucket where the digest files are stored. If a bucket name is not specified, the AWS CLI will retrieve it by calling `DescribeTrails()`.

`--s3-prefix`

Optionally specifies the Amazon S3 prefix where the digest files are stored. If not specified, the AWS CLI will retrieve it by calling `DescribeTrails()`.

> **Note**
>
> You should use this option only if your current prefix is different from the prefix that was in use during the time range that you specify.

`--account-id`

Optionally specifies the account for validating logs. This parameter is required for organization trails for validating logs for the specific account inside an organization.

`--trail-arn`

Specifies the Amazon Resource Name (ARN) of the trail to be validated. The format of a trail ARN follows.

```
arn:aws:cloudtrail:us-east-2:111111111111:trail/MyTrailName
```

> **Note**
>
> To obtain the trail ARN for a trail, you can use the `describe-trails` command before running `validate-logs`.
> You may want to specify the bucket name and prefix in addition to the trail ARN if log files have been delivered to more than one bucket in the time range that you specified, and you want to restrict the validation to the log files in only one of the buckets.

`--verbose`

Optionally outputs validation information for every log or digest file in the specified time range. The output indicates whether the file remains unchanged or has been modified or deleted. In non-verbose mode (the default), information is returned only for those cases in which there was a validation failure.

Example

The following example validates log files from the specified start time to the present, using the Amazon S3 bucket configured for the current trail and specifying verbose output.

```
aws cloudtrail validate-logs --start-time 2015-08-27T00:00:00Z --end-time
 2015-08-28T00:00:00Z --trail-arn arn:aws:cloudtrail:us-east-2:111111111111:trail/my-
trail-name --verbose
```

How `validate-logs` works

The `validate-logs` command starts by validating the most recent digest file in the specified time range. First, it verifies that the digest file has been downloaded from the location to which it claims to belong. In other words, if the CLI downloads digest file `df1` from the S3 location `p1`, validate-logs will verify that `p1 == df1.digestS3Bucket + '/' + df1.digestS3Object`.

If the signature of the digest file is valid, it checks the hash value of each of the logs referenced in the digest file. The command then goes back in time, validating the previous digest files and their referenced log files in succession. It continues until the specified value for `start-time` is reached, or until the digest chain ends. If a digest file is missing or not valid, the time range that cannot be validated is indicated in the output.

Validation results

Validation results begin with a summary header in the following format:

```
Validating log files for trail trail_ARN  between time_stamp and time_stamp
```

Each line of the main output contains the validation results for a single digest or log file in the following format:

```
<Digest file | Log file> <S3 path> <Validation Message>
```

The following table describes the possible validation messages for log and digest files.

File Type	Validation Message	Description
Digest file	valid	The digest file signature is valid. The log files it references can be checked. This message is included only in verbose mode.
Digest file	INVALID: has been moved from its original location	The S3 bucket or S3 object from which the digest file was retrieved do not match the S3 bucket or S3 object locations that are recorded in the digest file itself.
Digest file	INVALID: invalid format	The format of the digest file is invalid. The log files corresponding to the time range that the digest file represents cannot be validated.
Digest file	INVALID: not found	The digest file was not found. The log files corresponding to the time range that the digest file represents cannot be validated.
Digest file	INVALID: public key not found for fingerprint *fingerprint*	The public key corresponding to the fingerprint recorded in the digest file

File Type	Validation Message	Description
		was not found. The digest file cannot be validated.
`Digest file`	`INVALID: signature verification failed`	The digest file signature is not valid. Because the digest file is not valid, the log files it references cannot be validated, and no assertions can be made about the API activity in them.
`Digest file`	`INVALID: Unable to load PKCS #1 key with fingerprint` *`fingerprint`*	Because the DER encoded public key in PKCS #1 format having the specified fingerprint could not be loaded, the digest file cannot be validated.
`Log file`	`valid`	The log file has been validated and has not been modified since the time of delivery. This message is included only in verbose mode.
`Log file`	`INVALID: hash value doesn't match`	The hash for the log file does not match. The log file has been modified after delivery by CloudTrail.
`Log file`	`INVALID: invalid format`	The format of the log file is invalid. The log file cannot be validated.
`Log file`	`INVALID: not found`	The log file was not found and cannot be validated.

The output includes summary information about the results returned.

Example outputs

Verbose

The following example `validate-logs` command uses the `--verbose` flag and produces the sample output that follows. `[...]` indicates the sample output has been abbreviated.

```
aws cloudtrail validate-logs --trail-arn arn:aws:cloudtrail:us-
east-2:111111111111:trail/example-trail-name --start-time 2015-08-31T22:00:00Z --end-
time 2015-09-01T19:17:29Z --verbose
```

```
Validating log files for trail arn:aws:cloudtrail:us-east-2:111111111111:trail/example-
trail-name between 2015-08-31T22:00:00Z and 2015-09-01T19:17:29Z

Digest file    s3://DOC-EXAMPLE-BUCKET/AWSLogs/111111111111/CloudTrail-Digest/us-
east-2/2015/09/01/111111111111_CloudTrail-Digest_us-east-2_example-trail-name_us-
east-2_20150901T201728Z.json.gz valid
Log file       s3://DOC-EXAMPLE-BUCKET/AWSLogs/111111111111/
CloudTrail/us-east-2/2015/09/01/111111111111_CloudTrail_us-
east-2_20150901T1925Z_WZZw1RymnjCRjxXc.json.gz valid
Log file       s3://DOC-EXAMPLE-BUCKET/AWSLogs/111111111111/
CloudTrail/us-east-2/2015/09/01/111111111111_CloudTrail_us-
east-2_20150901T1915Z_POuvV87nu6pfAV2W.json.gz valid
Log file       s3://DOC-EXAMPLE-BUCKET/AWSLogs/111111111111/
CloudTrail/us-east-2/2015/09/01/111111111111_CloudTrail_us-
east-2_20150901T1930Z_l2QgXhAKVm1QXiIA.json.gz valid
Log file       s3://DOC-EXAMPLE-BUCKET/AWSLogs/111111111111/
CloudTrail/us-east-2/2015/09/01/111111111111_CloudTrail_us-
east-2_20150901T1920Z_eQJteBBrfpBCqOqw.json.gz valid
Log file       s3://DOC-EXAMPLE-BUCKET/AWSLogs/111111111111/
CloudTrail/us-east-2/2015/09/01/111111111111_CloudTrail_us-
east-2_20150901T1950Z_9g5A6qlR2B5KaRdq.json.gz valid
Log file       s3://DOC-EXAMPLE-BUCKET/AWSLogs/111111111111/
CloudTrail/us-east-2/2015/09/01/111111111111_CloudTrail_us-
east-2_20150901T1920Z_i4DNCC12BuXd6Ru7.json.gz valid
Log file       s3://DOC-EXAMPLE-BUCKET/AWSLogs/111111111111/
CloudTrail/us-east-2/2015/09/01/111111111111_CloudTrail_us-
east-2_20150901T1915Z_Sg5caf2RH6Jdx0EJ.json.gz valid
Digest file    s3://DOC-EXAMPLE-BUCKET/AWSLogs/111111111111/CloudTrail-Digest/us-
east-2/2015/09/01/111111111111_CloudTrail-Digest_us-east-2_example-trail-name_us-
east-2_20150901T191728Z.json.gz valid
Log file       s3://DOC-EXAMPLE-BUCKET/AWSLogs/111111111111/
CloudTrail/us-east-2/2015/09/01/111111111111_CloudTrail_us-
east-2_20150901T1910Z_YYSFiuFQk4nrtnEW.json.gz valid
[...]
Log file       s3://DOC-EXAMPLE-BUCKET/AWSLogs/144218288521/
CloudTrail/us-east-2/2015/09/01/144218288521_CloudTrail_us-
east-2_20150901T1055Z_0Sfy6m9f6iBzmoPF.json.gz valid
```

```
Log file       s3://DOC-EXAMPLE-BUCKET/AWSLogs/144218288521/
CloudTrail/us-east-2/2015/09/01/144218288521_CloudTrail_us-
east-2_20150901T1040Z_lLa3QzVLpOed7igR.json.gz valid

Digest file    s3://DOC-EXAMPLE-BUCKET/AWSLogs/144218288521/CloudTrail-Digest/us-
east-2/2015/09/01/144218288521_CloudTrail-Digest_us-east-2_example-trail-name_us-
east-2_20150901T101728Z.json.gz INVALID: signature verification failed

Digest file    s3://DOC-EXAMPLE-BUCKETAWSLogs/144218288521/CloudTrail-Digest/us-
east-2/2015/09/01/144218288521_CloudTrail-Digest_us-east-2_example-trail-name_us-
east-2_20150901T091728Z.json.gz valid
Log file       s3://DOC-EXAMPLE-BUCKET/AWSLogs/144218288521/
CloudTrail/us-east-2/2015/09/01/144218288521_CloudTrail_us-
east-2_20150901T0830Z_eaFvO3dwHo4NCqqc.json.gz valid
Digest file    s3://DOC-EXAMPLE-BUCKET/AWSLogs/144218288521/CloudTrail-Digest/us-
east-2/2015/09/01/144218288521_CloudTrail-Digest_us-east-2_example-trail-name_us-
east-2_20150901T081728Z.json.gz valid
Digest file    s3://DOC-EXAMPLE-BUCKET/AWSLogs/144218288521/CloudTrail-Digest/us-
east-2/2015/09/01/144218288521_CloudTrail-Digest_us-east-2_example-trail-name_us-
east-2_20150901T071728Z.json.gz valid
[...]
Log file       s3://DOC-EXAMPLE-BUCKET/AWSLogs/111111111111/
CloudTrail/us-east-2/2015/08/31/111111111111_CloudTrail_us-
east-2_20150831T2245Z_mbJkEO5kNcDnVhGh.json.gz valid
Log file       s3://DOC-EXAMPLE-BUCKET/AWSLogs/111111111111/
CloudTrail/us-east-2/2015/08/31/111111111111_CloudTrail_us-
east-2_20150831T2225Z_IQ6kXy8sKU03RSPr.json.gz valid
Log file       s3://DOC-EXAMPLE-BUCKET/AWSLogs/111111111111/
CloudTrail/us-east-2/2015/08/31/111111111111_CloudTrail_us-
east-2_20150831T2230Z_eRPVRTxHQ5498ROA.json.gz valid
Log file       s3://DOC-EXAMPLE-BUCKET/AWSLogs/111111111111/
CloudTrail/us-east-2/2015/08/31/111111111111_CloudTrail_us-
east-2_20150831T2255Z_IlWawYZGvTWB5vYN.json.gz valid
Digest file    s3://DOC-EXAMPLE-BUCKET/AWSLogs/111111111111/CloudTrail-Digest/us-
east-2/2015/08/31/111111111111_CloudTrail-Digest_us-east-2_example-trail-name_us-
east-2_20150831T221728Z.json.gz valid

Results requested for 2015-08-31T22:00:00Z to 2015-09-01T19:17:29Z
Results found for 2015-08-31T22:17:28Z to 2015-09-01T20:17:28Z:

22/23 digest files valid, 1/23 digest files INVALID
63/63 log files valid
```

Non-verbose

The following example `validate-logs` command does not use the `--verbose` flag. In the sample output that follows, one error was found. Only the header, error, and summary information are returned.

```
aws cloudtrail validate-logs --trail-arn arn:aws:cloudtrail:us-
east-2:111111111111:trail/example-trail-name --start-time 2015-08-31T22:00:00Z --end-
time 2015-09-01T19:17:29Z
```

```
Validating log files for trail arn:aws:cloudtrail:us-east-2:111111111111:trail/example-
trail-name between 2015-08-31T22:00:00Z and 2015-09-01T19:17:29Z

Digest file s3://DOC-EXAMPLE-BUCKET/AWSLogs/144218288521/CloudTrail-Digest/us-
east-2/2015/09/01/144218288521_CloudTrail-Digest_us-east-2_example-trail-name_us-
east-2_20150901T101728Z.json.gz INVALID: signature verification failed

Results requested for 2015-08-31T22:00:00Z to 2015-09-01T19:17:29Z
Results found for 2015-08-31T22:17:28Z to 2015-09-01T20:17:28Z:

22/23 digest files valid, 1/23 digest files INVALID
63/63 log files valid
```

Checking whether a particular file was delivered by CloudTrail

To check if a particular file in your bucket was delivered by CloudTrail, run `validate-logs` in verbose mode for the time period that includes the file. If the file appears in the output of `validate-logs`, then the file was delivered by CloudTrail.

CloudTrail digest file structure

Each digest file contains the names of the log files that were delivered to your Amazon S3 bucket during the last hour, the hash values for those log files, and the digital signature of the previous digest file. The signature for the current digest file is stored in the metadata properties of the digest file object. The digital signatures and hashes are used for validating the integrity of the log files and of the digest file itself.

Digest file location

Digest files are delivered to an Amazon S3 bucket location that follows this syntax.

```
s3://DOC-EXAMPLE-BUCKET/optional-prefix/AWSLogs/aws-account-id/CloudTrail-Digest/
    region/digest-end-year/digest-end-month/digest-end-date/
    aws-account-id_CloudTrail-Digest_region_trail-
name_region_digest_end_timestamp.json.gz
```

Note

For organization trails, the bucket location also includes the organization unit ID, as follows:

```
s3://DOC-EXAMPLE-BUCKET/optional-prefix/AWSLogs/O-ID/aws-account-id/CloudTrail-
Digest/
    region/digest-end-year/digest-end-month/digest-end-date/
    aws-account-id_CloudTrail-Digest_region_trail-
name_region_digest_end_timestamp.json.gz
```

Sample digest file contents

The following example digest file contains information for a CloudTrail log.

```
{
  "awsAccountId": "111122223333",
  "digestStartTime": "2015-08-17T14:01:31Z",
  "digestEndTime": "2015-08-17T15:01:31Z",
  "digestS3Bucket": "DOC-EXAMPLE-BUCKET",
  "digestS3Object": "AWSLogs/111122223333/CloudTrail-Digest/us-
east-2/2015/08/17/111122223333_CloudTrail-Digest_us-east-2_your-trail-name_us-
east-2_20150817T150131Z.json.gz",
  "digestPublicKeyFingerprint": "31e8b5433410dfb61a9dc45cc65b22ff",
  "digestSignatureAlgorithm": "SHA256withRSA",
  "newestEventTime": "2015-08-17T14:52:27Z",
  "oldestEventTime": "2015-08-17T14:42:27Z",
  "previousDigestS3Bucket": "DOC-EXAMPLE-BUCKET",
  "previousDigestS3Object": "AWSLogs/111122223333/CloudTrail-Digest/us-
east-2/2015/08/17/111122223333_CloudTrail-Digest_us-east-2_your-trail-name_us-
east-2_20150817T140131Z.json.gz",
  "previousDigestHashValue":
 "97fb791cf91ffc440d274f8190dbdd9aa09c34432aba82739df18b6d3c13df2d",
  "previousDigestHashAlgorithm": "SHA-256",
  "previousDigestSignature":
 "50887ccffad4c002b97caa37cc9dc626e3c680207d41d27fa5835458e066e0d3652fc4dfc30937e4d5f4cc7f796e7
  "logFiles": [
    {
      "s3Bucket": "DOC-EXAMPLE-BUCKET",
      "s3Object": "AWSLogs/111122223333/CloudTrail/us-
east-2/2015/08/17/111122223333_CloudTrail_us-
east-2_20150817T1445Z_9nYN7gp2eWAJHIfT.json.gz",
      "hashValue": "9bb6196fc6b84d6f075a56548feca262bd99ba3c2de41b618e5b6e22c1fc71f6",
      "hashAlgorithm": "SHA-256",
      "newestEventTime": "2015-08-17T14:52:27Z",
      "oldestEventTime": "2015-08-17T14:42:27Z"
    }
  ]
}
```

Digest file field descriptions

The following are descriptions for each field in the digest file:

`awsAccountId`

The AWS account ID for which the digest file has been delivered.

`digestStartTime`

The starting UTC time range that the digest file covers, taking as a reference the time in which log files have been delivered by CloudTrail. This means that if the time range is [Ta, Tb], the digest will contain all the log files delivered to the customer between Ta and Tb.

`digestEndTime`

The ending UTC time range that the digest file covers, taking as a reference the time in which log files have been delivered by CloudTrail. This means that if the time range is [Ta, Tb], the digest will contain all the log files delivered to the customer between Ta and Tb.

`digestS3Bucket`

The name of the Amazon S3 bucket to which the current digest file has been delivered.

`digestS3Object`

The Amazon S3 object key (that is, the Amazon S3 bucket location) of the current digest file. The first two Regions in the string show the Region from which the digest file was delivered. The last Region (after `your-trail-name`) is the home Region of the trail. The home Region is the Region in which the trail was created. In the case of a multi-Region trail, this can be different from the Region from which the digest file was delivered.

`newestEventTime`

The UTC time of the most recent event among all of the events in the log files in the digest.

`oldestEventTime`

The UTC time of the oldest event among all of the events in the log files in the digest.

> **Note**
>
> If the digest file is delivered late, the value of `oldestEventTime` will be earlier than the value of `digestStartTime`.

`previousDigestS3Bucket`

The Amazon S3 bucket to which the previous digest file was delivered.

`previousDigestS3Object`

The Amazon S3 object key (that is, the Amazon S3 bucket location) of the previous digest file.

`previousDigestHashValue`

The hexadecimal encoded hash value of the uncompressed contents of the previous digest file.

`previousDigestHashAlgorithm`

The name of the hash algorithm that was used to hash the previous digest file.

`publicKeyFingerprint`

The hexadecimal encoded fingerprint of the public key that matches the private key used to sign this digest file. You can retrieve the public keys for the time range corresponding to the digest file by using the AWS CLI or the CloudTrail API. Of the public keys returned, the one whose fingerprint matches this value can be used for validating the digest file. For information about retrieving public keys for digest files, see the AWS CLI `list-public-keys` command or the CloudTrail `ListPublicKeys` API.

> **Note**
>
> CloudTrail uses different private/public key pairs per Region. Each digest file is signed with a private key unique to its Region. Therefore, when you validate a digest file from a particular Region, you must look in the same Region for its corresponding public key.

`digestSignatureAlgorithm`

The algorithm used to sign the digest file.

`logFiles.s3Bucket`

The name of the Amazon S3 bucket for the log file.

`logFiles.s3Object`

The Amazon S3 object key of the current log file.

`logFiles.newestEventTime`

The UTC time of the most recent event in the log file. This time also corresponds to the time stamp of the log file itself.

`logFiles.oldestEventTime`

The UTC time of the oldest event in the log file.

`logFiles.hashValue`

The hexadecimal encoded hash value of the uncompressed log file content.

`logFiles.hashAlgorithm`

The hash algorithm used to hash the log file.

Starting digest file

When log file integrity validation is started, a starting digest file will be generated. A starting digest file will also be generated when log file integrity validation is restarted (by either disabling and then reenabling log file integrity validation, or by stopping logging and then restarting logging with validation enabled). In a starting digest file, the following fields relating to the previous digest file will be null:

- `previousDigestS3Bucket`
- `previousDigestS3Object`
- `previousDigestHashValue`
- `previousDigestHashAlgorithm`
- `previousDigestSignature`

'Empty' digest files

CloudTrail will deliver a digest file even when there has been no API activity in your account during the one hour period that the digest file represents. This can be useful when you need to assert that no log files were delivered during the hour reported by the digest file.

The following example shows the contents of a digest file that recorded an hour when no API activity occurred. Note that the `logFiles:[ ]` field at the end of the digest file contents is empty.

```
{
  "awsAccountId": "111122223333",
  "digestStartTime": "2015-08-20T17:01:31Z",
  "digestEndTime": "2015-08-20T18:01:31Z",
  "digestS3Bucket": "DOC-EXAMPLE-BUCKET",
  "digestS3Object": "AWSLogs/111122223333/CloudTrail-Digest/us-
east-2/2015/08/20/111122223333_CloudTrail-Digest_us-east-2_example-trail-name_us-
east-2_20150820T180131Z.json.gz",
  "digestPublicKeyFingerprint": "31e8b5433410dfb61a9dc45cc65b22ff",
  "digestSignatureAlgorithm": "SHA256withRSA",
  "newestEventTime": null,
  "oldestEventTime": null,
  "previousDigestS3Bucket": "DOC-EXAMPLE-BUCKET",
  "previousDigestS3Object": "AWSLogs/111122223333/CloudTrail-Digest/us-
east-2/2015/08/20/111122223333_CloudTrail-Digest_us-east-2_example-trail-name_us-
east-2_20150820T170131Z.json.gz",
  "previousDigestHashValue":
 "ed96c4bac9eaa8fe9716ca0e515da51938be651b1db31d781956416a9d05cdfa",
  "previousDigestHashAlgorithm": "SHA-256",
  "previousDigestSignature":
 "82705525fb0fe7f919f9434e5b7138cb41793c776c7414f3520c0242902daa8cc8286b29263d2627f2f259471c745
  "logFiles": []
}
```

Signature of the digest file

The signature information for a digest file is located in two object metadata properties of the Amazon S3 digest file object. Each digest file has the following metadata entries:

- `x-amz-meta-signature`

  The hexadecimal encoded value of the digest file signature. The following is an example signature:

  ```
  3be472336fa2989ef34de1b3c1bf851f59eb030eaff3e2fb6600a082a23f4c6a82966565b994f9de4a5989d053d9d
  28f1cc237f372264a51b611c01da429565def703539f4e71009051769469231bc22232fa260df02740047af532229
  05d3ffcb5d2dd5dc28f8bb5b7993938e8a5f912a82b448a367eccb2ec0f198ba71e23eb0b97278cf65f3c8d1e652c
  ```

- `x-amz-meta-signature-algorithm`

  The following shows an example value of the algorithm used to generate the digest signature:

  `SHA256withRSA`

Digest file chaining

The fact that each digest file contains a reference to its previous digest file enables a "chaining" that permits validation tools like the AWS CLI to detect if a digest file has been deleted. It also allows the digest files in a specified time range to be successively inspected, starting with the most recent first.

> **Note**
>
> When you disable log file integrity validation, the chain of digest files is broken after one hour. CloudTrail will not create digest files for log files that were delivered during a period in which log file integrity validation was disabled. For example, if you enable log file integrity validation at noon on January 1, disable it at noon on January 2, and re-enable it at noon on January 10, digest files will not be created for the log files delivered from noon on January 2 to noon on January 10. The same applies whenever you stop CloudTrail logging or delete a trail.

If your trail's S3 bucket policy is misconfigured or CloudTrail experiences an unexpected service disruption, you might not receive all or some digest files. To confirm if your trail has any digest delivery errors, run the `get-trail-status` command and check the `LatestDigestDeliveryError` parameter for errors. After the delivery issue is resolved (for example, by fixing the bucket policy), CloudTrail will attempt to redeliver any missing digest files. During the redelivery period, the digest files might be delivered out of order, so the chain might temporarily appear to be broken.

If logging is stopped or the trail is deleted, CloudTrail will deliver a final digest file. This digest file can contain information for any remaining log files that cover events up to and including the `StopLogging` event.

Custom implementations of CloudTrail log file integrity validation

Because CloudTrail uses industry standard, openly available cryptographic algorithms and hash functions, you can create your own tools to validate the integrity of CloudTrail log files. When log file integrity validation is enabled, CloudTrail delivers digest files to your Amazon S3 bucket. You can use these files to implement your own validation solution. For more information about digest files, see CloudTrail digest file structure.

This topic describes how digest files are signed, and then details the steps that you will need to take to implement a solution that validates the digest files and the log files that they reference.

Understanding how CloudTrail digest files are signed

CloudTrail digest files are signed with RSA digital signatures. For each digest file, CloudTrail does the following:

1. Creates a string for data signing based on designated digest file fields (described in the next section).
2. Gets a private key unique to the Region.
3. Passes the SHA-256 hash of the string and the private key to the RSA signing algorithm, which produces a digital signature.
4. Encodes the byte code of the signature into hexadecimal format.
5. Puts the digital signature into the `x-amz-meta-signature` metadata property of the Amazon S3 digest file object.

Contents of the data signing string

The following CloudTrail objects are included in the string for data signing:

- The ending timestamp of the digest file in UTC extended format (for example, `2015-05-08T07:19:37Z`)
- The current digest file S3 path
- The hexadecimal-encoded SHA-256 hash of the current digest file
- The hexadecimal-encoded signature of the previous digest file

The format for calculating this string and an example string are provided later in this document.

Custom validation implementation steps

When implementing a custom validation solution, you will need to validate the digest file first, and then the log files that it references.

Validate the digest file

To validate a digest file, you need its signature, the public key whose private key was used to signed it, and a data signing string that you compute.

1. Get the digest file.
2. Verify that the digest file has been retrieved from its original location.
3. Get the hexadecimal-encoded signature of the digest file.
4. Get the hexadecimal-encoded fingerprint of the public key whose private key was used to sign the digest file.
5. Retrieve the public keys for the time range corresponding to the digest file.
6. From among the public keys retrieved, choose the public key whose fingerprint matches the fingerprint in the digest file.
7. Using the digest file hash and other digest file fields, recreate the data signing string used to verify the digest file signature.
8. Validate the signature by passing in the SHA-256 hash of the string, the public key, and the signature as parameters to the RSA signature verification algorithm. If the result is true, the digest file is valid.

Validate the log files

If the digest file is valid, validate each of the log files that the digest file references.

1. To validate the integrity of a log file, compute its SHA-256 hash value on its uncompressed content and compare the results with the hash for the log file recorded in hexadecimal in the digest. If the hashes match, the log file is valid.
2. By using the information about the previous digest file that is included in the current digest file, validate the previous digest files and their corresponding log files in succession.

The following sections describe these steps in detail.

A. Get the digest file

The first steps are to get the most recent digest file, verify that you have retrieved it from its original location, verify its digital signature, and get the fingerprint of the public key.

1. Using S3 `GetObject` or the AmazonS3Client class (for example), get the most recent digest file from your Amazon S3 bucket for the time range that you want to validate.
2. Check that the S3 bucket and S3 object used to retrieve the file match the S3 bucket S3 object locations that are recorded in the digest file itself.

3. Next, get the digital signature of the digest file from the `x-amz-meta-signature` metadata property of the digest file object in Amazon S3.
4. In the digest file, get the fingerprint of the public key whose private key was used to sign the digest file from the `digestPublicKeyFingerprint` field.

B. Retrieve the public key for validating the digest file

To get the public key to validate the digest file, you can use either the AWS CLI or the CloudTrail API. In both cases, you specify a time range (that is, a start time and end time) for the digest files that you want to validate. One or more public keys may be returned for the time range that you specify. The returned keys may have validity time ranges that overlap.

> **Note**
>
> Because CloudTrail uses different private/public key pairs per Region, each digest file is signed with a private key unique to its Region. Therefore, when you validate a digest file from a particular Region, you must retrieve its public key from the same Region.

Use the AWS CLI to retrieve public keys

To retrieve public keys for digest files by using the AWS CLI, use the `cloudtrail list-public-keys` command. The command has the following format:

```
aws cloudtrail list-public-keys [--start-time <start-time>] [--end-time <end-time>]
```

The start-time and end-time parameters are UTC timestamps and are optional. If not specified, the current time is used, and the currently active public key or keys are returned.

Sample Response

The response will be a list of JSON objects representing the key (or keys) returned:

```
{
    "publicKeyList": [
        {
            "ValidityStartTime": "1436317441.0",
            "ValidityEndTime": "1438909441.0",
            "Value": "MIIBCgKCAQEAn11L2YZ9h7onug2ILi1MWyHiMRsTQjfWE
+pHVRLk1QjfWhirG+lpOa8NrwQ/r7Ah5bNL6HepznOU9XTDSfmmnP97mqyc7z/upfZdS/AHhYcGaz7n6Wc/
RRBU6VmiPCrAUojuSk6/GjvA8iOPFsYDuBtviXarvuLPlrT9kAd4Lb+rFfR5peEgBEkhlzc5HuWO7S0y
+KunqxX6jQBnXGMtxmPBPP0FylgWGNdFtks/4YSKcgqwH0YDcawP9GGGDAeCIqPWIXDLG1jOjRRzWfCmD0iJUkz8vTsn4hq
            "Fingerprint": "8eba5db5bea9b640d1c96a77256fe7f2"
        },
        {
            "ValidityStartTime": "1434589460.0",
            "ValidityEndTime": "1437181460.0",
            "Value": "MIIBCgKCAQEApfYL2FiZhpN74LNWVUzhR
+VheYhwhYm8w0n5Gf6i95ylW5kBAWKVEmnAQG7BvS5g9SMqFDQx52fW7NWV44IvfJ2xGXT
+wT+DgR6ZQ+6yxskQNqV5YcXj4Aa5Zz4jJfsYjDuO2MDTZNIzNvBNzaBJ+r2WIWAJ/
Xq54kyF63B6WE38vKuDE7nSd1FqQuEoNBFLPInvgggYe2Ym1Refe2z71wNcJ2kY
+q0h1BSHrSM8RWuJIw7MXwF9iQncg9jYzUlNJomozQzAG5wSRfbplcCYNY40xvGd/aAmO0m+Y
+XFMrKwtLCwseHPvj843qVno6x4BJN9bpWnoPo9sdsbGoiK3QIDAQAB",
            "Fingerprint": "8933b39ddc64d26d8e14ffbf6566fee4"
        },
        {
            "ValidityStartTime": "1434589370.0",
            "ValidityEndTime": "1437181370.0",
            "Value":
 "MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAqlzPJbvZJ42UdcmLfPUqXYNfOs6I8lCfao/
tOs8CmzPOEdtLWugB9xoIUz78qVHdKIqxbaG4jWHfJBiOSSFBM0lt8cdVo4TnRa7oG9io5pysS6DJhBBAeXsicufsiFJR
+wrUNh8RSLxL4k6G1+BhLX20tJkZ/erT97tDGBujAelqseGg3vPZbTx9SMfOLN65PdLFudLP7Gat0Z9p5jw/
rjpclKfo9Bfc3heeBxWGKwBBOKnFAaN9V57pOaosCvPKmHd9bg7jsQkI9Xp22IzGLsTFJZYVA3KiTAElDMu80iFXPHEq9hK
+1utKVEiLkR2disdCmPTK0VQIDAQAB",
            "Fingerprint": "31e8b5433410dfb61a9dc45cc65b22ff"
        }
    ]
}
```

Use the CloudTrail API to retrieve public keys

To retrieve public keys for digest files by using the CloudTrail API, pass in start time and end time values to the `ListPublicKeys` API. The `ListPublicKeys` API returns the public keys whose private keys were used to sign digest files within the specified time range. For each public key, the API also returns the corresponding fingerprint.

`ListPublicKeys`

This section describes the request parameters and response elements for the `ListPublicKeys` API.

> **Note**
>
> The encoding for the binary fields for `ListPublicKeys` is subject to change.

Request Parameters

Name	Description
`StartTime`	Optionally specifies, in UTC, the start of the time range to look up public keys for CloudTrail digest files. If StartTime is not specified, the current time is used, and the current public key is returned. Type: DateTime
`EndTime`	Optionally specifies, in UTC, the end of the time range to look up public keys for CloudTrail digest files. If EndTime is not specified, the current time is used. Type: DateTime

Response Elements

`PublicKeyList`, an array of `PublicKey` objects that contains:

Name	Description
`Value`	The DER encoded public key value in PKCS #1 format. Type: Blob
`ValidityStartTime`	The starting time of validity of the public key. Type: DateTime
`ValidityEndTime`	The ending time of validity of the public key. Type: DateTime

`Fingerprint`	The fingerprint of the public key. The fingerprint can be used to identify the public key that you must use to validate the digest file. Type: String

C. Choose the public key to use for validation

From among the public keys retrieved by `list-public-keys` or `ListPublicKeys`, choose the public key returned whose fingerprint matches the fingerprint recorded in the `digestPublicKeyFingerprint` field of the digest file. This is the public key that you will use to validate the digest file.

D. Recreate the data signing string

Now that you have the signature of the digest file and associated public key, you need to calculate the data signing string. After you have calculated the data signing string, you will have the inputs needed to verify the signature.

The data signing string has the following format:

```
Data_To_Sign_String =
  Digest_End_Timestamp_in_UTC_Extended_format + '\n' +
  Current_Digest_File_S3_Path + '\n' +
  Hex(Sha256(current-digest-file-content)) + '\n' +
  Previous_digest_signature_in_hex
```

An example `Data_To_Sign_String` follows.

```
2015-08-12T04:01:31Z
DOC-EXAMPLE-BUCKET/AWSLogs/111122223333/CloudTrail-Digest/us-
east-2/2015/08/12/111122223333_us-east-2_CloudTrail-Digest_us-
east-2_20150812T040131Z.json.gz
4ff08d7c6ecd6eb313257e839645d20363ee3784a2328a7d76b99b53cc9bcacd
6e8540b83c3ac86a0312d971a225361d28ed0af20d70c211a2d405e32abf529a8145c2966e3bb47362383a52441545e
d4c7c09dd152b84e79099ce7a9ec35d2b264eb92eb6e090f1e5ec5d40ec8a0729c02ff57f9e30d5343a8591638f8b79
98b0aee2c1c8af74ec620261529265e83a9834ebef6054979d3e9a6767dfa6fdb4ae153436c567d6ae208f988047ccf
```

After you recreate this string, you can validate the digest file.

E. Validate the digest file

Pass the SHA-256 hash of the recreated data signing string, digital signature, and public key to the RSA signature verification algorithm. If the output is true, the signature of the digest file is verified and the digest file is valid.

F. Validate the log files

After you have validated the digest file, you can validate the log files it references. The digest file contains the SHA-256 hashes of the log files. If one of the log files was modified after CloudTrail delivered it, the SHA-256 hashes will change, and the signature of digest file will not match.

The following shows how validate the log files:

1. Do an `S3 Get` of the log file using the S3 location information in the digest file's `logFiles.s3Bucket` and `logFiles.s3Object` fields.
2. If the `S3 Get` operation is successful, iterate through the log files listed in the digest file's logFiles array using the following steps:
   a. Retrieve the original hash of the file from the `logFiles.hashValue` field of the corresponding log in the digest file.
   b. Hash the uncompressed contents of the log file with the hashing algorithm specified in `logFiles.hashAlgorithm`.
   c. Compare the hash value that you generated with the one for the log in the digest file. If the hashes match, the log file is valid.

G. Validate additional digest and log files

In each digest file, the following fields provide the location and signature of the previous digest file:

- `previousDigestS3Bucket`
- `previousDigestS3Object`
- `previousDigestSignature`

Use this information to visit previous digest files sequentially, validating the signature of each and the log files that they reference by using the steps in the previous sections. The only difference is that for previous digest files, you do not need to retrieve the digital signature from the digest file object's Amazon S3 metadata properties. The signature for the previous digest file is provided for you in the `previousDigestSignature` field.

You can go back until the starting digest file is reached, or until the chain of digest files is broken, whichever comes first.

Validating digest and log files offline

When validating digest and log files offline, you can generally follow the procedures described in the previous sections. However, you must take into account the following areas:

Handling the most recent digest file

The digital signature of the most recent (that is, "current") digest file is in the Amazon S3 metadata properties of the digest file object. In an offline scenario, the digital signature for the current digest file will not be available.

Two possible ways of handling this are:

- Since the digital signature for the previous digest file is in the current digest file, start validating from the next-to-last digest file. With this method, the most recent digest file cannot be validated.
- As a preliminary step, obtain the signature for the current digest file from the digest file object's metadata properties and then store it securely offline. This would allow the current digest file to be validated in addition to the previous files in the chain.

Path resolution

Fields in the downloaded digest files like `s3Object` and `previousDigestS3Object` will still be pointing to Amazon S3 online locations for log files and digest files. An offline solution must find a way to reroute these to the current path of the downloaded log and digest files.

Public keys

In order to validate offline, all of the public keys that you need for validating log files in a given time range must first be obtained online (by calling `ListPublicKeys`, for example) and then stored securely offline. This step must be repeated whenever you want to validate additional files outside the initial time range that you specified.

Sample validation snippet

The following sample snippet provides skeleton code for validating CloudTrail digest and log files. The skeleton code is online/offline agnostic; that is, it is up to you to decide whether to implement it with or without online connectivity to AWS. The suggested implementation uses the Java Cryptography Extension (JCE) and Bouncy Castle as a security provider.

The sample snippet shows:

- How to create the data signing string used to validate the digest file signature.
- How to verify the digest file signature.
- How to verify the log file hashes.
- A code structure for validating a chain of digest files.

```
import java.util.Arrays;
import java.security.MessageDigest;
import java.security.KeyFactory;
import java.security.PublicKey;
import java.security.Security;
import java.security.Signature;
import java.security.spec.X509EncodedKeySpec;
import org.json.JSONObject;
import org.bouncycastle.jce.provider.BouncyCastleProvider;
import org.apache.commons.codec.binary.Hex;

public class DigestFileValidator {

    public void validateDigestFile(String digestS3Bucket, String digestS3Object, String
 digestSignature) {

        // Using the Bouncy Castle provider as a JCE security provider - http://
www.bouncycastle.org/
        Security.addProvider(new BouncyCastleProvider());

        // Load the digest file from S3 (using Amazon S3 Client) or from your local
 copy
        JSONObject digestFile = loadDigestFileInMemory(digestS3Bucket, digestS3Object);

        // Check that the digest file has been retrieved from its original location
        if (!digestFile.getString("digestS3Bucket").equals(digestS3Bucket) ||
                !digestFile.getString("digestS3Object").equals(digestS3Object)) {
            System.err.println("Digest file has been moved from its original 
 location.");
        } else {
            // Compute digest file hash
            MessageDigest messageDigest = MessageDigest.getInstance("SHA-256");
            messageDigest.update(convertToByteArray(digestFile));
            byte[] digestFileHash = messageDigest.digest();
            messageDigest.reset();

            // Compute the data to sign
            String dataToSign = String.format("%s%n%s/%s%n%s%n%s",
                                digestFile.getString("digestEndTime"),
                                digestFile.getString("digestS3Bucket"), 
 digestFile.getString("digestS3Object"), // Constructing the S3 path of the digest file 
 as part of the data to sign
                                Hex.encodeHexString(digestFileHash),
                                digestFile.getString("previousDigestSignature"));

            byte[] signatureContent = Hex.decodeHex(digestSignature);

            /*
                NOTE:
                To find the right public key to verify the signature, call CloudTrail 
 ListPublicKey API to get a list
                of public keys, then match by the publicKeyFingerprint in the digest 
 file. Also, the public key bytes
                returned from ListPublicKey API are DER encoded in PKCS#1 format:

                PublicKeyInfo ::= SEQUENCE {
                    algorithm       AlgorithmIdentifier,
                    PublicKey       BIT STRING
                }

                AlgorithmIdentifier ::= SEQUENCE {
                    algorithm       OBJECT IDENTIFIER,
                    parameters      ANY DEFINED BY algorithm OPTIONAL
                }
            */
            pkcs1PublicKeyBytes = 
 getPublicKey(digestFile.getString("digestPublicKeyFingerprint")));

            // Transform the PKCS#1 formatted public key to x.509 format.
            RSAPublicKey rsaPublicKey = RSAPublicKey.getInstance(pkcs1PublicKeyBytes);
            AlgorithmIdentifier rsaEncryption = new
 AlgorithmIdentifier(PKCSObjectIdentifiers.rsaEncryption, null);
            SubjectPublicKeyInfo publicKeyInfo = new
 SubjectPublicKeyInfo(rsaEncryption, rsaPublicKey);

            // Create the PublicKey object needed for the signature validation
            PublicKey publicKey = KeyFactory.getInstance("RSA",
 "BC").generatePublic(new X509EncodedKeySpec(publicKeyInfo.getEncoded()));

            // Verify signature
            Signature signature = Signature.getInstance("SHA256withRSA", "BC");
            signature.initVerify(publicKey);
            signature.update(dataToSign.getBytes("UTF-8"));

            if (signature.verify(signatureContent)) {
                System.out.println("Digest file signature is valid, validating log
 files…");
                for (int i = 0; i < digestFile.getJSONArray("logFiles").length(); i++)
 {

                    JSONObject logFileMetadata =
 digestFile.getJSONArray("logFiles").getJSONObject(i);

                    // Compute log file hash
                    byte[] logFileContent = loadUncompressedLogFileInMemory(
                                                logFileMetadata.getString("s3Bucket"),
                                                logFileMetadata.getString("s3Object")
                                            );
                    messageDigest.update(logFileContent);
                     byte[] logFileHash = messageDigest.digest();
                    messageDigest.reset();

                    // Retrieve expected hash for the log file being processed
                    byte[] expectedHash =
 Hex.decodeHex(logFileMetadata.getString("hashValue"));

                    boolean signaturesMatch = Arrays.equals(expectedHash, logFileHash);
                    if (!signaturesMatch) {
                        System.err.println(String.format("Log file: %s/%s hash doesn't
 match.\tExpected: %s Actual: %s",
                               logFileMetadata.getString("s3Bucket"),
 logFileMetadata.getString("s3Object"),
                               Hex.encodeHexString(expectedHash),
 Hex.encodeHexString(logFileHash)));
```

```
                    } else {
                        System.out.println(String.format("Log file: %s/%s hash match",
                               logFileMetadata.getString("s3Bucket"),
 logFileMetadata.getString("s3Object")));
                    }
                }

            } else {
                System.err.println("Digest signature failed validation.");
            }

            System.out.println("Digest file validation completed.");

            if (chainValidationIsEnabled()) {
                // This enables the digests' chain validation
                validateDigestFile(
                        digestFile.getString("previousDigestS3Bucket"),
                        digestFile.getString("previousDigestS3Object"),
                        digestFile.getString("previousDigestSignature"));
            }
        }
    }
}
```

CloudTrail log file examples

CloudTrail monitors events for your account. If you create a trail, it delivers those events as log files to your Amazon S3 bucket. If you create an event data store in CloudTrail Lake, events are logged to your event data store. Event data stores do not use S3 buckets.

Topics

- CloudTrail log file name format
- Log file examples

CloudTrail log file name format

CloudTrail uses the following file name format for the log file objects that it delivers to your Amazon S3 bucket:

```
AccountID_CloudTrail_RegionName_YYYYMMDDTHHmmZ_UniqueString.FileNameFormat
```

- The YYYY, MM, DD, HH, and mm are the digits of the year, month, day, hour, and minute when the log file was delivered. Hours are in 24-hour format. The Z indicates that the time is in UTC.

  > **Note**
  >
  > A log file delivered at a specific time can contain records written at any point before that time.

- The 16-character UniqueString component of the log file name is there to prevent overwriting of files. It has no meaning, and log processing software should ignore it.
- FileNameFormat is the encoding of the file. Currently, this is json.gz, which is a JSON text file in compressed gzip format.

Example CloudTrail Log File Name

```
111122223333_CloudTrail_us-east-2_20150801T0210Z_Mu0KsOhtH1ar15ZZ.json.gz
```

Log file examples

A log file contains one or more records. The following examples are snippets of logs that show the records for an action that started the creation of a log file.

For information about CloudTrail event record fields, see CloudTrail record contents.

Contents

Amazon EC2 log examples

Amazon Elastic Compute Cloud (Amazon EC2) provides resizeable computing capacity in the AWS Cloud. You can launch virtual servers, configure security and networking, and manage storage.

Amazon EC2 can also scale up or down quickly to handle changes in requirements or spikes in popularity, thereby reducing your need to forecast server traffic. For more information, see the Amazon EC2 User Guide.

The following example shows that an IAM user named `Mateo` ran the **aws ec2 start-instances** command to call the Amazon EC2 `StartInstances` action for instances `i-EXAMPLE56126103cb` and `i-EXAMPLEaff4840c22`.

```
{"Records": [{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EXAMPLE6E4XEGITWATV6R",
        "arn": "arn:aws:iam::123456789012:user/Mateo",
        "accountId": "123456789012",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "Mateo",
        "sessionContext": {
            "sessionIssuer": {},
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2023-07-19T21:11:57Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2023-07-19T21:17:28Z",
    "eventSource": "ec2.amazonaws.com",
    "eventName": "StartInstances",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "aws-cli/2.13.5 Python/3.11.4 Linux/4.14.255-314-253.539.amzn2.x86_64
 exec-env/CloudShell exe/x86_64.amzn.2 prompt/off command/ec2.start-instances",
    "requestParameters": {
        "instancesSet": {
            "items": [
                {
                    "instanceId": "i-EXAMPLE56126103cb"
                },
                {
                    "instanceId": "i-EXAMPLEaff4840c22"
                }
            ]
        }
    },
    "responseElements": {
        "requestId": "e4336db0-149f-4a6b-844d-EXAMPLEb9d16",
        "instancesSet": {
            "items": [
                {
                    "instanceId": "i-EXAMPLEaff4840c22",
                    "currentState": {
                        "code": 0,
                        "name": "pending"
                    },
                    "previousState": {
                        "code": 80,
                        "name": "stopped"
                    }
                },
                {
                    "instanceId": "i-EXAMPLE56126103cb",
                    "currentState": {
                        "code": 0,
                        "name": "pending"
                    },
                    "previousState": {
                        "code": 80,
                        "name": "stopped"
                    }
                }
            ]
        }
    },
    "requestID": "e4336db0-149f-4a6b-844d-EXAMPLEb9d16",
    "eventID": "e755e09c-42f9-4c5c-9064-EXAMPLE228c7",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "eventCategory": "Management",
     "tlsDetails": {
        "tlsVersion": "TLSv1.2",
        "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
        "clientProvidedHostHeader": "ec2.us-east-1.amazonaws.com"
    },
    "sessionCredentialFromConsole": "true"
```

```
}]}
```

The following example shows that an IAM user named `Nikki` ran the **aws ec2 stop-instances** command to call the Amazon EC2 `StopInstances` action to stop two instances.

```
{"Records": [{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "EXAMPLE6E4XEGITWATV6R",
        "arn": "arn:aws:iam::777788889999:user/Nikki",
        "accountId": "777788889999",
        "accessKeyId": "AKIAI44QH8DHBEXAMPLE",
        "userName": "Nikki",
        "sessionContext": {
            "sessionIssuer": {},
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2023-07-19T21:11:57Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2023-07-19T21:14:20Z",
    "eventSource": "ec2.amazonaws.com",
    "eventName": "StopInstances",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "aws-cli/2.13.5 Python/3.11.4 Linux/4.14.255-314-253.539.amzn2.x86_64
 exec-env/CloudShell exe/x86_64.amzn.2 prompt/off command/ec2.stop-instances",
    "requestParameters": {
        "instancesSet": {
            "items": [
                {
                    "instanceId": "i-EXAMPLE56126103cb"
                },
                {
                    "instanceId": "i-EXAMPLEaff4840c22"
                }
            ]
        },
        "force": false
    },
    "responseElements": {
        "requestId": "c308a950-e43e-444e-afc1-EXAMPLE73e49",
        "instancesSet": {
            "items": [
                {
                    "instanceId": "i-EXAMPLE56126103cb",
                    "currentState": {
                        "code": 64,
                        "name": "stopping"
                    },
                    "previousState": {
                        "code": 16,
                        "name": "running"
                    }
                },
                {
                    "instanceId": "i-EXAMPLEaff4840c22",
                    "currentState": {
                        "code": 64,
                        "name": "stopping"
                    },
                    "previousState": {
                        "code": 16,
                        "name": "running"
                    }
                }
            ]
        }
    },
    "requestID": "c308a950-e43e-444e-afc1-EXAMPLE73e49",
    "eventID": "9357a8cc-a0eb-46a1-b67e-EXAMPLE19b14",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "777788889999",
    "eventCategory": "Management",
     "tlsDetails": {
        "tlsVersion": "TLSv1.2",
        "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
        "clientProvidedHostHeader": "ec2.us-east-1.amazonaws.com"
    },
    "sessionCredentialFromConsole": "true"
}]}
```

The following example shows that an IAM user named `Arnav` ran the **aws ec2 create-key-pair** command to call the `CreateKeyPair` action. Note that the `responseElements` contain a hash of the key pair and that AWS removed the key material.

```
{"Records": [{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDA6ON6E4XEGIEXAMPLE",
        "arn": "arn:aws:iam::444455556666:user/Arnav",
        "accountId": "444455556666",
        "accessKeyId": "AKIAI44QH8DHBEXAMPLE",
        "userName": "Arnav",
        "sessionContext": {
            "sessionIssuer": {},
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2023-07-19T21:11:57Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2023-07-19T21:19:22Z",
    "eventSource": "ec2.amazonaws.com",
    "eventName": "CreateKeyPair",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "aws-cli/2.13.5 Python/3.11.4 Linux/4.14.255-314-253.539.amzn2.x86_64
 exec-env/CloudShell exe/x86_64.amzn.2 prompt/off command/ec2.create-key-pair",
    "requestParameters": {
        "keyName": "my-key",
        "keyType": "rsa",
        "keyFormat": "pem"
    },
    "responseElements": {
        "requestId": "9aa4938f-720f-4f4b-9637-EXAMPLE9a196",
        "keyName": "my-key",
        "keyFingerprint":
 "1f:51:ae:28:bf:89:e9:d8:1f:25:5d:37:2d:7d:b8:ca:9f:f5:f1:6f",
        "keyPairId": "key-abcd12345eEXAMPLE",
        "keyMaterial": "<sensitiveDataRemoved>"
    },
    "requestID": "9aa4938f-720f-4f4b-9637-EXAMPLE9a196",
    "eventID": "2ae450ff-e72b-4de1-87b0-EXAMPLE5227cb",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "444455556666",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.2",
        "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
        "clientProvidedHostHeader": "ec2.us-east-1.amazonaws.com"
    },
    "sessionCredentialFromConsole": "true"
}]}
```

IAM log examples

AWS Identity and Access Management (IAM) is a web service that helps you securely control access to AWS resources. With IAM, you can centrally manage permissions that control which AWS resources users can access. You use IAM to control who is authenticated (signed in) and authorized (has permissions) to use resources. For more information, see the IAM User Guide.

The following example shows that the IAM user named `Mary` ran the **aws iam create-user** command to call the `CreateUser` action to create a new user named `Richard`.

```
{"Records": [{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDA6ON6E4XEGITEXAMPLE",
        "arn": "arn:aws:iam::888888888888:user/Mary",
        "accountId": "888888888888",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "Mary",
        "sessionContext": {
            "sessionIssuer": {},
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2023-07-19T21:11:57Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2023-07-19T21:25:09Z",
    "eventSource": "iam.amazonaws.com",
    "eventName": "CreateUser",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "aws-cli/2.13.5 Python/3.11.4 Linux/4.14.255-314-253.539.amzn2.x86_64
 exec-env/CloudShell exe/x86_64.amzn.2 prompt/off command/iam.create-user",
    "requestParameters": {
        "userName": "Richard"
    },
    "responseElements": {
        "user": {
            "path": "/",
            "arn": "arn:aws:iam::888888888888:user/Richard",
            "userId": "AIDA6ON6E4XEP7EXAMPLE",
            "createDate": "Jul 19, 2023 9:25:09 PM",
            "userName": "Richard"
        }
    },
    "requestID": "2d528c76-329e-410b-9516-EXAMPLE565dc",
    "eventID": "ba0801a1-87ec-4d26-be87-EXAMPLE75bbb",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "888888888888",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.2",
        "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
        "clientProvidedHostHeader": "iam.amazonaws.com"
    },
    "sessionCredentialFromConsole": "true"
}]}
```

The following example shows that the IAM user named Paulo ran the **aws iam add-user-to-group** command to call the AddUserToGroup action to add a user named Jane to the Admin group.

```
{"Records": [{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDA6ON6E4XEGIEXAMPLE",
        "arn": "arn:aws:iam::555555555555:user/Paulo",
        "accountId": "555555555555",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "Paulo",
        "sessionContext": {
            "sessionIssuer": {},
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2023-07-19T21:11:57Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2023-07-19T21:25:09Z",
    "eventSource": "iam.amazonaws.com",
    "eventName": "AddUserToGroup",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "aws-cli/2.13.5 Python/3.11.4 Linux/4.14.255-314-253.539.amzn2.x86_64
 exec-env/CloudShell exe/x86_64.amzn.2 prompt/off command/iam.add-user-to-group",
    "requestParameters": {
        "groupName": "Admin",
        "userName": "Jane"
    },
    "responseElements": null,
    "requestID": "ecd94349-b36f-44bf-b6f5-EXAMPLE9c463",
    "eventID": "2939ba50-1d26-4a5a-83bd-EXAMPLE85850",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "555555555555",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.2",
        "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
        "clientProvidedHostHeader": "iam.amazonaws.com"
    },
    "sessionCredentialFromConsole": "true"
}]}
```

The following example shows that the IAM user named `Saanvi` ran the **aws iam create-role** command to call the `CreateRole` action to create a role.

```
{"Records": [{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDA6ON6E4XEGITEXAMPLE",
        "arn": "arn:aws:iam::777777777777:user/Saanvi",
        "accountId": "777777777777",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "Saanvi",
        "sessionContext": {
            "sessionIssuer": {},
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2023-07-19T21:11:57Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2023-07-19T21:29:12Z",
    "eventSource": "iam.amazonaws.com",
    "eventName": "CreateRole",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "aws-cli/2.13.5 Python/3.11.4 Linux/4.14.255-314-253.539.amzn2.x86_64
 exec-env/CloudShell exe/x86_64.amzn.2 prompt/off command/iam.create-role",
    "requestParameters": {
        "roleName": "TestRole",
        "description": "Allows EC2 instances to call AWS services on your behalf.",
        "assumeRolePolicyDocument": "{\"Version\":\"2012-10-17\",\"Statement\":
[{\"Effect\":\"Allow\",\"Action\":[\"sts:AssumeRole\"],\"Principal\":{\"Service\":
[\"ec2.amazonaws.com\"]}}]}"
    },
    "responseElements": {
        "role": {
            "assumeRolePolicyDocument": "%7B%22Version%22%3A%222012-10-17%22%2C
%22Statement%22%3A%5B%7B%22Effect%22%3A%22Allow%22%2C%22Action%22%3A%5B%22sts
%3AAssumeRole%22%5D%2C%22Principal%22%3A%7B%22Service%22%3A%5B%22ec2.amazonaws.com
%22%5D%7D%7D%5D%7D",
            "arn": "arn:aws:iam::777777777777:role/TestRole",
            "roleId": "AROA6ON6E4XEFFEXAMPLE",
            "createDate": "Jul 19, 2023 9:29:12 PM",
            "roleName": "TestRole",
            "path": "/"
        }
    },
    "requestID": "ff38f36e-ebd3-425b-9939-EXAMPLE1bbe",
    "eventID": "9da77cd0-493f-4c89-8852-EXAMPLEa887c",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "777777777777",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.2",
        "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
        "clientProvidedHostHeader": "iam.amazonaws.com"
    },
    "sessionCredentialFromConsole": "true"
}]}
```

Error code and message log example

The following example shows that the IAM user named `Terry` ran the **aws cloudtrail update-trail** command to call the [`UpdateTrail`] action to update a trail named `myTrail2`, but the trail name was not found. The log shows this error in the `errorCode` and `errorMessage` elements.

```
{"Records": [{
    "eventVersion": "1.09",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDA6ON6E4XEGIEXAMPLE",
        "arn": "arn:aws:iam::111122223333:user/Terry",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "Terry",
        "sessionContext": {
            "attributes": {
                "creationDate": "2023-07-19T21:11:57Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2023-07-19T21:35:03Z",
    "eventSource": "cloudtrail.amazonaws.com",
    "eventName": "UpdateTrail",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "aws-cli/2.13.0 Python/3.11.4 Linux/4.14.255-314-253.539.amzn2.x86_64
 exec-env/CloudShell exe/x86_64.amzn.2 prompt/off command/cloudtrail.update-trail",
    "errorCode": "TrailNotFoundException",
    "errorMessage": "Unknown trail: arn:aws:cloudtrail:us-east-1:111122223333:trail/
myTrail2 for the user: 111122223333",
    "requestParameters": {
        "name": "myTrail2",
        "isMultiRegionTrail": true
    },
    "responseElements": null,
    "requestID": "28d2faaf-3319-4649-998d-EXAMPLE72818",
    "eventID": "694d604a-d190-4470-8dd1-EXAMPLEe20c1",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.2",
        "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
        "clientProvidedHostHeader": "cloudtrail.us-east-1.amazonaws.com"
    },
    "sessionCredentialFromConsole": "true"
}]}
```

CloudTrail Insights event log example

The following example shows a CloudTrail Insights event log. An Insights event is actually a pair of events that mark the start and end of a period of unusual write management API activity or error response activity. The `state` field shows whether the event was logged at the start or end of the period of unusual activity. The event name, `UpdateInstanceInformation`, is the same name as the AWS Systems Manager API for which CloudTrail analyzed management events to determine that unusual activity occurred. Although the start and end events have unique `eventID` values, they also have a `sharedEventID` value that is used by the pair. The Insights event shows the `baseline`, or the normal pattern of activity, the `insight`, or average unusual activity that triggered the start Insights event, and in the end event, the `insight` value for the average unusual activity over the duration of the Insights event. For more information about CloudTrail Insights, see Logging Insights events.

```
{
    "Records": [{
        "eventVersion": "1.08",
        "eventTime": "2023-01-02T02:51:00Z",
        "awsRegion": "us-east-1",
        "eventID": "654a30ff-b0f3-4527-81b6-EXAMPLEf2393",
        "eventType": "AwsCloudTrailInsight",
        "recipientAccountId": "123456789012",
        "sharedEventID": "bcbfc274-8559-4a56-beb0-EXAMPLEa6c34",
        "insightDetails": {
            "state": "Start",
            "eventSource": "ssm.amazonaws.com",
            "eventName": "UpdateInstanceInformation",
            "insightType": "ApiCallRateInsight",
            "insightContext": {
                "statistics": {
                    "baseline": {
                        "average": 84.410596421
                    },
                    "insight": {
                        "average": 669
                    }
                }
            }
        },
        "eventCategory": "Insight"
    },
    {
        "eventVersion": "1.08",
        "eventTime": "2023-01-02T00:22:00Z",
        "awsRegion": "us-east-1",
        "eventID": "258de2fb-e2a9-4fb5-aeb2-EXAMPLE449a4",
        "eventType": "AwsCloudTrailInsight",
        "recipientAccountId": "123456789012",
        "sharedEventID": "8b74a7bc-d5d3-4d19-9d60-EXAMPLE08b51",
        "insightDetails": {
            "state": "End",
            "eventSource": "ssm.amazonaws.com",
            "eventName": "UpdateInstanceInformation",
            "insightType": "ApiCallRateInsight",
            "insightContext": {
                "statistics": {
                    "baseline": {
                        "average": 74.156423842
                    },
                    "insight": {
                        "average": 657
                    },
                    "insightDuration": 1
                }
            }
        },
        "eventCategory": "Insight"
    }]
}
```

Using the CloudTrail Processing Library

The CloudTrail Processing Library is a Java library that provides an easy way to process AWS CloudTrail logs. You provide configuration details about your CloudTrail SQS queue and write code to process events. The CloudTrail Processing Library does the rest. It polls your Amazon SQS queue, reads and parses queue messages, downloads CloudTrail log files, parses events in the log files, and passes the events to your code as Java objects.

The CloudTrail Processing Library is highly scalable and fault-tolerant. It handles parallel processing of log files so that you can process as many logs as needed. It handles network failures related to network timeouts and inaccessible resources.

The following topic shows you how to use the CloudTrail Processing Library to process CloudTrail logs in your Java projects.

The library is provided as an Apache-licensed open-source project, available on GitHub: https://github.com/aws/aws-cloudtrail-processing-library. The library source includes sample code that you can use as a base for your own projects.

Topics

Minimum requirements

To use the CloudTrail Processing Library, you must have the following:

- AWS SDK for Java 1.11.830
- Java 1.8 (Java SE 8)

Processing CloudTrail logs

To process CloudTrail logs in your Java application:

1. Adding the CloudTrail Processing Library to your project
2. Configuring the CloudTrail Processing Library
3. Implementing the events processor
4. Instantiating and running the processing executor

Adding the CloudTrail Processing Library to your project

To use the CloudTrail Processing Library, add it to your Java project's classpath.

Contents

Adding the library to an Apache Ant project

To add the CloudTrail Processing Library to an Apache Ant project

1. Download or clone the CloudTrail Processing Library source code from GitHub:
   - https://github.com/aws/aws-cloudtrail-processing-library
2. Build the .jar file from source as described in the README:

   ```
   mvn clean install -Dgpg.skip=true
   ```

3. Copy the resulting .jar file into your project and add it to your project's `build.xml` file. For example:

```
<classpath>
  <pathelement path="${classpath}"/>
  <pathelement location="lib/aws-cloudtrail-processing-library-1.6.1.jar"/>
</classpath>
```

Adding the library to an Apache Maven project

The CloudTrail Processing Library is available for Apache Maven. You can add it to your project by writing a single dependency in your project's `pom.xml` file.

To add the CloudTrail Processing Library to a Maven project

- Open your Maven project's `pom.xml` file and add the following dependency:

```
<dependency>
    <groupId>com.amazonaws</groupId>
    <artifactId>aws-cloudtrail-processing-library</artifactId>
    <version>1.6.1</version>
</dependency>
```

Adding the library to an Eclipse project

To add the CloudTrail Processing Library to an Eclipse project

1. Download or clone the CloudTrail Processing Library source code from GitHub:

   - https://github.com/aws/aws-cloudtrail-processing-library

2. Build the .jar file from source as described in the README:

```
mvn clean install -Dgpg.skip=true
```

3. Copy the built aws-cloudtrail-processing-library-1.6.1.jar to a directory in your project (typically `lib`).
4. Right-click your project's name in the Eclipse **Project Explorer**, choose **Build Path**, and then choose **Configure**
5. In the **Java Build Path** window, choose the **Libraries** tab.

6. Choose **Add JARs...** and navigate to the path where you copied aws-cloudtrail-processing-library-1.6.1.jar.
7. Choose **OK** to complete adding the `.jar` to your project.

Adding the library to an IntelliJ project

To add the CloudTrail Processing Library to an IntelliJ project

1. Download or clone the CloudTrail Processing Library source code from GitHub:
   - https://github.com/aws/aws-cloudtrail-processing-library
2. Build the .jar file from source as described in the README:

   ```
   mvn clean install -Dgpg.skip=true
   ```
3. From **File**, choose **Project Structure**.
4. Choose **Modules** and then choose **Dependencies**.
5. Choose **+ JARS or Directories** and then go to the path where you built the `aws-cloudtrail-processing-library-1.6.1.jar`.
6. Choose **Apply** and then choose **OK** to complete adding the `.jar` to your project.

Configuring the CloudTrail Processing Library

You can configure the CloudTrail Processing Library by creating a classpath properties file that is loaded at runtime, or by creating a `ClientConfiguration` object and setting options manually.

Providing a properties file

You can write a classpath properties file that provides configuration options to your application. The following example file shows the options you can set:

```
# AWS access key. (Required)
accessKey = your_access_key

# AWS secret key. (Required)
secretKey = your_secret_key

# The SQS URL used to pull CloudTrail notification from. (Required)
sqsUrl = your_sqs_queue_url
```

```
# The SQS end point specific to a region.
sqsRegion = us-east-1

# A period of time during which Amazon SQS prevents other consuming components
# from receiving and processing that message.
visibilityTimeout = 60

# The S3 region to use.
s3Region = us-east-1

# Number of threads used to download S3 files in parallel. Callbacks can be
# invoked from any thread.
threadCount = 1

# The time allowed, in seconds, for threads to shut down after
# AWSCloudTrailEventProcessingExecutor.stop() is called. If they are still
# running beyond this time, they will be forcibly terminated.
threadTerminationDelaySeconds = 60

# The maximum number of AWSCloudTrailClientEvents sent to a single invocation
# of processEvents().
maxEventsPerEmit = 10

# Whether to include raw event information in CloudTrailDeliveryInfo.
enableRawEventInfo = false

# Whether to delete SQS message when the CloudTrail Processing Library is unable to
 process the notification.
deleteMessageUponFailure = false
```

The following parameters are required:

- `sqsUrl` – Provides the URL from which to pull your CloudTrail notifications. If you don't specify this value, the `AWSCloudTrailProcessingExecutor` throws an `IllegalStateException`.
- `accessKey` – A unique identifier for your account, such as AKIAIOSFODNN7EXAMPLE.
- `secretKey` – A unique identifier for your account, such as wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY.

The `accessKey` and `secretKey` parameters provide your AWS credentials to the library so the library can access AWS on your behalf.

Defaults for the other parameters are set by the library. For more information, see the AWS CloudTrail Processing Library Reference.

Creating a ClientConfiguration

Instead of setting options in the classpath properties, you can provide options to the `AWSCloudTrailProcessingExecutor` by initializing and setting options on a `ClientConfiguration` object, as shown in the following example:

```
ClientConfiguration basicConfig = new ClientConfiguration(
    "http://sqs.us-east-1.amazonaws.com/123456789012/queue2",
    new DefaultAWSCredentialsProviderChain());

basicConfig.setEnableRawEventInfo(true);
basicConfig.setThreadCount(4);
basicConfig.setnEventsPerEmit(20);
```

Implementing the events processor

To process CloudTrail logs, you must implement an `EventsProcessor` that receives the CloudTrail log data. The following is an example implementation:

```
public class SampleEventsProcessor implements EventsProcessor {

    public void process(List<CloudTrailEvent> events) {
        int i = 0;
        for (CloudTrailEvent event : events) {
            System.out.println(String.format("Process event %d : %s", i++,
 event.getEventData()));
        }
    }
}
```

When implementing an `EventsProcessor`, you implement the `process()` callback that the `AWSCloudTrailProcessingExecutor` uses to send you CloudTrail events. Events are provided in a list of `CloudTrailClientEvent` objects.

The `CloudTrailClientEvent` object provides a `CloudTrailEvent` and `CloudTrailEventMetadata` that you can use to read the CloudTrail event and delivery information.

This simple example prints the event information for each event passed to SampleEventsProcessor. In your own implementation, you can process logs as you see fit. The AWSCloudTrailProcessingExecutor continues to send events to your EventsProcessor as long as it has events to send and is still running.

Instantiating and running the processing executor

After you write an EventsProcessor and set configuration values for the CloudTrail Processing Library (either in a properties file or by using the ClientConfiguration class), you can use these elements to initialize and use an AWSCloudTrailProcessingExecutor.

To use AWSCloudTrailProcessingExecutor to process CloudTrail events

1. Instantiate an AWSCloudTrailProcessingExecutor.Builder object. Builder's constructor takes an EventsProcessor object and a classpath properties file name.
2. Call the Builder's build() factory method to configure and obtain an AWSCloudTrailProcessingExecutor object.
3. Use the AWSCloudTrailProcessingExecutor's start() and stop() methods to begin and end CloudTrail event processing.

```
public class SampleApp {
  public static void main(String[] args) throws InterruptedException {
    AWSCloudTrailProcessingExecutor executor = new
      AWSCloudTrailProcessingExecutor.Builder(new SampleEventsProcessor(),
        "/myproject/cloudtrailprocessing.properties").build();

    executor.start();
    Thread.sleep(24 * 60 * 60 * 1000); // let it run for a while (optional)
    executor.stop(); // optional
  }
}
```

Advanced topics

Topics

- Filtering the events to process
- Processing data events
- Reporting progress

- Handling errors

Filtering the events to process

By default, all logs in your Amazon SQS queue's S3 bucket and all events that they contain are sent to your `EventsProcessor`. The CloudTrail Processing Library provides optional interfaces that you can implement to filter the sources used to obtain CloudTrail logs and to filter the events that you are interested in processing.

`SourceFilter`

You can implement the `SourceFilter` interface to choose whether you want to process logs from a provided source. `SourceFilter` declares a single callback method, `filterSource()`, that receives a `CloudTrailSource` object. To keep events from a source from being processed, return `false` from `filterSource()`.

The CloudTrail Processing Library calls the `filterSource()` method after the library polls for logs on the Amazon SQS queue. This occurs before the library starts event filtering or processing for the logs.

The following is an example implementation:

```
public class SampleSourceFilter implements SourceFilter{
  private static final int MAX_RECEIVED_COUNT = 3;

  private static List<String> accountIDs ;
  static {
    accountIDs = new ArrayList<>();
    accountIDs.add("123456789012");
    accountIDs.add("234567890123");
  }

  @Override
  public boolean filterSource(CloudTrailSource source) throws CallbackException {
    source = (SQSBasedSource) source;
    Map<String, String> sourceAttributes = source.getSourceAttributes();

    String accountId = sourceAttributes.get(
      SourceAttributeKeys.ACCOUNT_ID.getAttributeKey());

    String receivedCount = sourceAttributes.get(
      SourceAttributeKeys.APPROXIMATE_RECEIVE_COUNT.getAttributeKey());
    int approximateReceivedCount = Integer.parseInt(receivedCount);

    return approximateReceivedCount <= MAX_RECEIVED_COUNT &&
 accountIDs.contains(accountId);
  }
}
```

If you don't provide your own `SourceFilter`, then `DefaultSourceFilter` is used, which allows all sources to be processed (it always returns `true`).

`EventFilter`

You can implement the `EventFilter` interface to choose whether a CloudTrail event is sent to your `EventsProcessor`. `EventFilter` declares a single callback method, `filterEvent()`, that receives a `CloudTrailEvent` object. To keep the event from being processed, return `false` from `filterEvent()`.

The CloudTrail Processing Library calls the `filterEvent()` method after the library polls for logs on the Amazon SQS queue and after source filtering. This occurs before the library starts event processing for the logs.

See the following example implementation:

```
public class SampleEventFilter implements EventFilter{

  private static final String EC2_EVENTS = "ec2.amazonaws.com";

  @Override
  public boolean filterEvent(CloudTrailClientEvent clientEvent) throws
 CallbackException {
    CloudTrailEvent event = clientEvent.getEvent();

    String eventSource = event.getEventSource();
    String eventName = event.getEventName();

    return eventSource.equals(EC2_EVENTS) && eventName.startsWith("Delete");
  }
}
```

If you don't provide your own `EventFilter`, then `DefaultEventFilter` is used, which allows all events to be processed (it always returns `true`).

Processing data events

When CloudTrail processes data events, it preserves numbers in their original format, whether that is an integer (`int`) or a `float` (a number that contains a decimal). In events that have integers in the fields of a data event, CloudTrail historically processed these numbers as floats. Currently, CloudTrail processes numbers in these fields by keeping their original format.

As a best practice, to avoid breaking your automations, be flexible in any code or automation that you are using to process or filter CloudTrail data events, and allow both `int` and `float` formatted numbers. For best results, use version 1.4.0 or higher of the CloudTrail Processing Library.

The following example snippet shows a `float` formatted number, `2.0`, for the `desiredCount` parameter in the `ResponseParameters` block of a data event.

```
"eventName": "CreateService",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "000.00.00.00",
    "userAgent": "console.amazonaws.com",
    "requestParameters": {
        "clientToken": "EXAMPLE",
        "cluster": "default",
        "desiredCount": 2.0
...
```

The following example snippet shows an `int` formatted number, 2, for the `desiredCount` parameter in the `ResponseParameters` block of a data event.

```
"eventName": "CreateService",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "000.00.00.00",
    "userAgent": "console.amazonaws.com",
    "requestParameters": {
        "clientToken": "EXAMPLE",
        "cluster": "default",
        "desiredCount": 2
...
```

Reporting progress

Implement the `ProgressReporter` interface to customize the reporting of CloudTrail Processing Library progress. `ProgressReporter` declares two methods: `reportStart()` and `reportEnd()`, which are called at the beginning and end of the following operations:

- Polling messages from Amazon SQS
- Parsing messages from Amazon SQS
- Processing an Amazon SQS source for CloudTrail logs
- Deleting messages from Amazon SQS
- Downloading a CloudTrail log file
- Processing a CloudTrail log file

Both methods receive a `ProgressStatus` object that contains information about the operation that was performed. The `progressState` member holds a member of the `ProgressState` enumeration that identifies the current operation. This member can contain additional information in the `progressInfo` member. Additionally, any object that you return from `reportStart()` is passed to `reportEnd()`, so you can provide contextual information such as the time when the event began processing.

The following is an example implementation that provides information about how long an operation took to complete:

```
public class SampleProgressReporter implements ProgressReporter {
  private static final Log logger =
    LogFactory.getLog(DefaultProgressReporter.class);

  @Override
  public Object reportStart(ProgressStatus status) {
    return new Date();
  }

  @Override
  public void reportEnd(ProgressStatus status, Object startDate) {
    System.out.println(status.getProgressState().toString() + " is " +
      status.getProgressInfo().isSuccess() + " , and latency is " +
      Math.abs(((Date) startDate).getTime()-new Date().getTime()) + "
      milliseconds.");
  }
}
```

If you don't implement your own `ProgressReporter`, then `DefaultExceptionHandler`, which prints the name of the state being run, is used instead.

Handling errors

The `ExceptionHandler` interface allows you to provide special handling when an exception occurs during log processing. `ExceptionHandler` declares a single callback method, `handleException()`, which receives a `ProcessingLibraryException` object with context about the exception that occurred.

You can use the passed-in `ProcessingLibraryException`'s `getStatus()` method to find out what operation was executed when the exception occurred and get additional information about the status of the operation. `ProcessingLibraryException` is derived from Java's standard `Exception` class, so you can also retrieve information about the exception by invoking any of the exception methods.

See the following example implementation:

```
public class SampleExceptionHandler implements ExceptionHandler{
  private static final Log logger =
    LogFactory.getLog(DefaultProgressReporter.class);

  @Override
  public void handleException(ProcessingLibraryException exception) {
    ProgressStatus status = exception.getStatus();
    ProgressState state = status.getProgressState();
    ProgressInfo info = status.getProgressInfo();

    System.err.println(String.format(
      "Exception. Progress State: %s. Progress Information: %s.", state, info));
  }
}
```

If you don't provide your own `ExceptionHandler`, then `DefaultExceptionHandler`, which prints a standard error message, is used instead.

Note

If the `deleteMessageUponFailure` parameter is `true`, the CloudTrail Processing Library does not distinguish general exceptions from processing errors and may delete queue messages.

1. For example, you use the `SourceFilter` to filter messages by timestamp.
2. However, you don't have the required permissions to access the S3 bucket that receives the CloudTrail log files. Because you don't have the required permissions, an `AmazonServiceException` is thrown. The CloudTrail Processing Library wraps this in a `CallBackException`.
3. The `DefaultExceptionHandler` logs this as an error, but does not identify the root cause, which is that you don't have the required permissions. The CloudTrail Processing Library considers this a processing error and deletes the message, even if the message includes a valid CloudTrail log file.

If you want to filter messages with `SourceFilter`, verify that your `ExceptionHandler` can distinguish service exceptions from processing errors.

Additional resources

For more information about the CloudTrail Processing Library, see the following:

- CloudTrail Processing Library GitHub project, which includes sample code that demonstrates how to implement a CloudTrail Processing Library application.
- CloudTrail Processing Library Java Package Documentation.

Security in AWS CloudTrail

Cloud security at AWS is the highest priority. As an AWS customer, you benefit from a data center and network architecture that is built to meet the requirements of the most security-sensitive organizations.

Security is a shared responsibility between AWS and you. The shared responsibility model describes this as security *of* the cloud and security *in* the cloud:

- **Security of the cloud** – AWS is responsible for protecting the infrastructure that runs AWS services in the AWS Cloud. AWS also provides you with services that you can use securely. Third-party auditors regularly test and verify the effectiveness of our security as part of the AWS compliance programs. To learn about the compliance programs that apply to AWS CloudTrail, see AWS Services in Scope by Compliance Program.
- **Security in the cloud** – Your responsibility is determined by the AWS service that you use. You are also responsible for other factors including the sensitivity of your data, your company's requirements, and applicable laws and regulations.

This documentation helps you understand how to apply the shared responsibility model when using CloudTrail. The following topics show you how to configure CloudTrail to meet your security and compliance objectives. You also learn how to use other AWS services that help you to monitor and secure your CloudTrail resources.

Topics

Data protection in AWS CloudTrail

The AWS shared responsibility model applies to data protection in AWS CloudTrail. As described in this model, AWS is responsible for protecting the global infrastructure that runs all of the AWS Cloud. You are responsible for maintaining control over your content that is hosted on this infrastructure. You are also responsible for the security configuration and management tasks for the AWS services that you use. For more information about data privacy, see the Data Privacy FAQ. For information about data protection in Europe, see the AWS Shared Responsibility Model and GDPR blog post on the *AWS Security Blog*.

For data protection purposes, we recommend that you protect AWS account credentials and set up individual users with AWS IAM Identity Center or AWS Identity and Access Management (IAM). That way, each user is given only the permissions necessary to fulfill their job duties. We also recommend that you secure your data in the following ways:

- Use multi-factor authentication (MFA) with each account.
- Use SSL/TLS to communicate with AWS resources. We require TLS 1.2 and recommend TLS 1.3.
- Set up API and user activity logging with AWS CloudTrail.
- Use AWS encryption solutions, along with all default security controls within AWS services.
- Use advanced managed security services such as Amazon Macie, which assists in discovering and securing sensitive data that is stored in Amazon S3.
- If you require FIPS 140-2 validated cryptographic modules when accessing AWS through a command line interface or an API, use a FIPS endpoint. For more information about the available FIPS endpoints, see Federal Information Processing Standard (FIPS) 140-2.

We strongly recommend that you never put confidential or sensitive information, such as your customers' email addresses, into tags or free-form text fields such as a **Name** field. This includes when you work with CloudTrail or other AWS services using the console, API, AWS CLI, or AWS SDKs. Any data that you enter into tags or free-form text fields used for names may be used for billing or diagnostic logs. If you provide a URL to an external server, we strongly recommend that you do not include credentials information in the URL to validate your request to that server.

By default, CloudTrail event log files are encrypted using Amazon S3 server-side encryption (SSE). You can also choose to encrypt your log files with an AWS Key Management Service (AWS KMS) key. You can store your log files in your bucket for as long as you want. You can also define Amazon

S3 lifecycle rules to archive or delete log files automatically. If you want notifications about log file delivery and validation, you can set up Amazon SNS notifications.

The following security best practices also address data protection in CloudTrail:

- Encrypting CloudTrail log files with AWS KMS keys (SSE-KMS)
- Amazon S3 bucket policy for CloudTrail
- Validating CloudTrail log file integrity
- Sharing CloudTrail log files between AWS accounts

Because CloudTrail logs files are stored in a bucket or buckets in Amazon S3, you should also review the data protection information in the Amazon Simple Storage Service User Guide. For more information, see Data protection in Amazon S3.

Identity and Access Management for AWS CloudTrail

AWS Identity and Access Management (IAM) is an AWS service that helps an administrator securely control access to AWS resources. IAM administrators control who can be *authenticated* (signed in) and *authorized* (have permissions) to use CloudTrail resources. IAM is an AWS service that you can use with no additional charge.

Topics

- AWS managed policies for AWS CloudTrail

Audience

How you use AWS Identity and Access Management (IAM) differs, depending on the work that you do in CloudTrail.

Service user – If you use the CloudTrail service to do your job, then your administrator provides you with the credentials and permissions that you need. As you use more CloudTrail features to do your work, you might need additional permissions. Understanding how access is managed can help you request the right permissions from your administrator. If you cannot access a feature in CloudTrail, see Troubleshooting AWS CloudTrail identity and access.

Service administrator – If you're in charge of CloudTrail resources at your company, you probably have full access to CloudTrail. It's your job to determine which CloudTrail features and resources your service users should access. You must then submit requests to your IAM administrator to change the permissions of your service users. Review the information on this page to understand the basic concepts of IAM. To learn more about how your company can use IAM with CloudTrail, see How AWS CloudTrail works with IAM.

IAM administrator – If you're an IAM administrator, you might want to learn details about how you can write policies to manage access to CloudTrail. To view example CloudTrail identity-based policies that you can use in IAM, see Identity-based policy examples for AWS CloudTrail.

Authenticating with identities

Authentication is how you sign in to AWS using your identity credentials. You must be *authenticated* (signed in to AWS) as the AWS account root user, as an IAM user, or by assuming an IAM role.

You can sign in to AWS as a federated identity by using credentials provided through an identity source. AWS IAM Identity Center (IAM Identity Center) users, your company's single sign-on authentication, and your Google or Facebook credentials are examples of federated identities. When you sign in as a federated identity, your administrator previously set up identity federation using IAM roles. When you access AWS by using federation, you are indirectly assuming a role.

Depending on the type of user you are, you can sign in to the AWS Management Console or the AWS access portal. For more information about signing in to AWS, see How to sign in to your AWS account in the *AWS Sign-In User Guide*.

If you access AWS programmatically, AWS provides a software development kit (SDK) and a command line interface (CLI) to cryptographically sign your requests by using your credentials. If you don't use AWS tools, you must sign requests yourself. For more information about using the recommended method to sign requests yourself, see Signing AWS API requests in the *IAM User Guide*.

Regardless of the authentication method that you use, you might be required to provide additional security information. For example, AWS recommends that you use multi-factor authentication (MFA) to increase the security of your account. To learn more, see Multi-factor authentication in the *AWS IAM Identity Center User Guide* and Using multi-factor authentication (MFA) in AWS in the *IAM User Guide*.

AWS account root user

When you create an AWS account, you begin with one sign-in identity that has complete access to all AWS services and resources in the account. This identity is called the AWS account *root user* and is accessed by signing in with the email address and password that you used to create the account. We strongly recommend that you don't use the root user for your everyday tasks. Safeguard your root user credentials and use them to perform the tasks that only the root user can perform. For the complete list of tasks that require you to sign in as the root user, see Tasks that require root user credentials in the *IAM User Guide*.

Federated identity

As a best practice, require human users, including users that require administrator access, to use federation with an identity provider to access AWS services by using temporary credentials.

A *federated identity* is a user from your enterprise user directory, a web identity provider, the AWS Directory Service, the Identity Center directory, or any user that accesses AWS services by using credentials provided through an identity source. When federated identities access AWS accounts, they assume roles, and the roles provide temporary credentials.

For centralized access management, we recommend that you use AWS IAM Identity Center. You can create users and groups in IAM Identity Center, or you can connect and synchronize to a set of users and groups in your own identity source for use across all your AWS accounts and applications. For information about IAM Identity Center, see What is IAM Identity Center? in the *AWS IAM Identity Center User Guide*.

IAM users and groups

An *IAM user* is an identity within your AWS account that has specific permissions for a single person or application. Where possible, we recommend relying on temporary credentials instead of creating IAM users who have long-term credentials such as passwords and access keys. However, if you have specific use cases that require long-term credentials with IAM users, we recommend that you rotate access keys. For more information, see Rotate access keys regularly for use cases that require long-term credentials in the *IAM User Guide*.

An *IAM group* is an identity that specifies a collection of IAM users. You can't sign in as a group. You can use groups to specify permissions for multiple users at a time. Groups make permissions easier to manage for large sets of users. For example, you could have a group named *IAMAdmins* and give that group permissions to administer IAM resources.

Users are different from roles. A user is uniquely associated with one person or application, but a role is intended to be assumable by anyone who needs it. Users have permanent long-term credentials, but roles provide temporary credentials. To learn more, see When to create an IAM user (instead of a role) in the *IAM User Guide*.

IAM roles

An *IAM role* is an identity within your AWS account that has specific permissions. It is similar to an IAM user, but is not associated with a specific person. You can temporarily assume an IAM role in the AWS Management Console by switching roles. You can assume a role by calling an AWS CLI or AWS API operation or by using a custom URL. For more information about methods for using roles, see Using IAM roles in the *IAM User Guide*.

IAM roles with temporary credentials are useful in the following situations:

- **Federated user access** – To assign permissions to a federated identity, you create a role and define permissions for the role. When a federated identity authenticates, the identity is associated with the role and is granted the permissions that are defined by the role. For information about roles for federation, see Creating a role for a third-party Identity Provider in the *IAM User Guide*. If you use IAM Identity Center, you configure a permission set. To control what your identities can access after they authenticate, IAM Identity Center correlates the permission set to a role in IAM. For information about permissions sets, see Permission sets in the *AWS IAM Identity Center User Guide*.
- **Temporary IAM user permissions** – An IAM user or role can assume an IAM role to temporarily take on different permissions for a specific task.

- **Cross-account access** – You can use an IAM role to allow someone (a trusted principal) in a different account to access resources in your account. Roles are the primary way to grant cross-account access. However, with some AWS services, you can attach a policy directly to a resource (instead of using a role as a proxy). To learn the difference between roles and resource-based policies for cross-account access, see Cross account resource access in IAM in the *IAM User Guide*.
- **Cross-service access** – Some AWS services use features in other AWS services. For example, when you make a call in a service, it's common for that service to run applications in Amazon EC2 or store objects in Amazon S3. A service might do this using the calling principal's permissions, using a service role, or using a service-linked role.
  - **Forward access sessions (FAS)** – When you use an IAM user or role to perform actions in AWS, you are considered a principal. When you use some services, you might perform an action that then initiates another action in a different service. FAS uses the permissions of the principal calling an AWS service, combined with the requesting AWS service to make requests to downstream services. FAS requests are only made when a service receives a request that requires interactions with other AWS services or resources to complete. In this case, you must have permissions to perform both actions. For policy details when making FAS requests, see Forward access sessions.
  - **Service role** – A service role is an IAM role that a service assumes to perform actions on your behalf. An IAM administrator can create, modify, and delete a service role from within IAM. For more information, see Creating a role to delegate permissions to an AWS service in the *IAM User Guide*.
  - **Service-linked role** – A service-linked role is a type of service role that is linked to an AWS service. The service can assume the role to perform an action on your behalf. Service-linked roles appear in your AWS account and are owned by the service. An IAM administrator can view, but not edit the permissions for service-linked roles.
- **Applications running on Amazon EC2** – You can use an IAM role to manage temporary credentials for applications that are running on an EC2 instance and making AWS CLI or AWS API requests. This is preferable to storing access keys within the EC2 instance. To assign an AWS role to an EC2 instance and make it available to all of its applications, you create an instance profile that is attached to the instance. An instance profile contains the role and enables programs that are running on the EC2 instance to get temporary credentials. For more information, see Using an IAM role to grant permissions to applications running on Amazon EC2 instances in the *IAM User Guide*.

To learn whether to use IAM roles or IAM users, see When to create an IAM role (instead of a user) in the *IAM User Guide*.

Managing access using policies

You control access in AWS by creating policies and attaching them to AWS identities or resources. A policy is an object in AWS that, when associated with an identity or resource, defines their permissions. AWS evaluates these policies when a principal (user, root user, or role session) makes a request. Permissions in the policies determine whether the request is allowed or denied. Most policies are stored in AWS as JSON documents. For more information about the structure and contents of JSON policy documents, see Overview of JSON policies in the *IAM User Guide*.

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

By default, users and roles have no permissions. To grant users permission to perform actions on the resources that they need, an IAM administrator can create IAM policies. The administrator can then add the IAM policies to roles, and users can assume the roles.

IAM policies define permissions for an action regardless of the method that you use to perform the operation. For example, suppose that you have a policy that allows the `iam:GetRole` action. A user with that policy can get role information from the AWS Management Console, the AWS CLI, or the AWS API.

Identity-based policies

Identity-based policies are JSON permissions policy documents that you can attach to an identity, such as an IAM user, group of users, or role. These policies control what actions users and roles can perform, on which resources, and under what conditions. To learn how to create an identity-based policy, see Creating IAM policies in the *IAM User Guide*.

Identity-based policies can be further categorized as *inline policies* or *managed policies*. Inline policies are embedded directly into a single user, group, or role. Managed policies are standalone policies that you can attach to multiple users, groups, and roles in your AWS account. Managed policies include AWS managed policies and customer managed policies. To learn how to choose between a managed policy or an inline policy, see Choosing between managed policies and inline policies in the *IAM User Guide*.

Resource-based policies

Resource-based policies are JSON policy documents that you attach to a resource. Examples of resource-based policies are IAM *role trust policies* and Amazon S3 *bucket policies*. In services that support resource-based policies, service administrators can use them to control access to a specific resource. For the resource where the policy is attached, the policy defines what actions a specified principal can perform on that resource and under what conditions. You must specify a principal in a resource-based policy. Principals can include accounts, users, roles, federated users, or AWS services.

Resource-based policies are inline policies that are located in that service. You can't use AWS managed policies from IAM in a resource-based policy.

Access control lists (ACLs)

Access control lists (ACLs) control which principals (account members, users, or roles) have permissions to access a resource. ACLs are similar to resource-based policies, although they do not use the JSON policy document format.

Amazon S3, AWS WAF, and Amazon VPC are examples of services that support ACLs. To learn more about ACLs, see Access control list (ACL) overview in the *Amazon Simple Storage Service Developer Guide*.

Other policy types

AWS supports additional, less-common policy types. These policy types can set the maximum permissions granted to you by the more common policy types.

- **Permissions boundaries** – A permissions boundary is an advanced feature in which you set the maximum permissions that an identity-based policy can grant to an IAM entity (IAM user or role). You can set a permissions boundary for an entity. The resulting permissions are the intersection of an entity's identity-based policies and its permissions boundaries. Resource-based policies that specify the user or role in the `Principal` field are not limited by the permissions boundary. An explicit deny in any of these policies overrides the allow. For more information about permissions boundaries, see Permissions boundaries for IAM entities in the *IAM User Guide*.
- **Service control policies (SCPs)** – SCPs are JSON policies that specify the maximum permissions for an organization or organizational unit (OU) in AWS Organizations. AWS Organizations is a service for grouping and centrally managing multiple AWS accounts that your business owns. If you enable all features in an organization, then you can apply service control policies (SCPs) to

any or all of your accounts. The SCP limits permissions for entities in member accounts, including each AWS account root user. For more information about Organizations and SCPs, see How SCPs work in the *AWS Organizations User Guide*.

- **Session policies** – Session policies are advanced policies that you pass as a parameter when you programmatically create a temporary session for a role or federated user. The resulting session's permissions are the intersection of the user or role's identity-based policies and the session policies. Permissions can also come from a resource-based policy. An explicit deny in any of these policies overrides the allow. For more information, see Session policies in the *IAM User Guide*.

Multiple policy types

When multiple types of policies apply to a request, the resulting permissions are more complicated to understand. To learn how AWS determines whether to allow a request when multiple policy types are involved, see Policy evaluation logic in the *IAM User Guide*.

How AWS CloudTrail works with IAM

Before you use IAM to manage access to CloudTrail, learn what IAM features are available to use with CloudTrail.

IAM features you can use with AWS CloudTrail

IAM feature	CloudTrail support
Identity-based policies	Yes
Resource-based policies	Partial
Policy actions	Yes
Policy resources	Yes
Policy condition keys (service-specific)	No
ACLs	No
ABAC (tags in policies)	Partial
Temporary credentials	Yes

IAM feature	CloudTrail support
Forward access sessions (FAS)	Yes
Service roles	Yes
Service-linked roles	Yes

To get a high-level view of how CloudTrail and other AWS services work with most IAM features, see AWS services that work with IAM in the *IAM User Guide*.

Identity-based policies for CloudTrail

Supports identity-based policies: Yes

Identity-based policies are JSON permissions policy documents that you can attach to an identity, such as an IAM user, group of users, or role. These policies control what actions users and roles can perform, on which resources, and under what conditions. To learn how to create an identity-based policy, see Creating IAM policies in the *IAM User Guide*.

With IAM identity-based policies, you can specify allowed or denied actions and resources as well as the conditions under which actions are allowed or denied. You can't specify the principal in an identity-based policy because it applies to the user or role to which it is attached. To learn about all of the elements that you can use in a JSON policy, see IAM JSON policy elements reference in the *IAM User Guide*.

Identity-based policy examples for CloudTrail

To view examples of CloudTrail identity-based policies, see Identity-based policy examples for AWS CloudTrail.

Resource-based policies within CloudTrail

Supports resource-based policies: Partial

Resource-based policies are JSON policy documents that you attach to a resource. Examples of resource-based policies are IAM *role trust policies* and Amazon S3 *bucket policies*. In services that support resource-based policies, service administrators can use them to control access to a specific resource. For the resource where the policy is attached, the policy defines what actions a specified principal can perform on that resource and under what conditions. You must specify a principal in a resource-based policy. Principals can include accounts, users, roles, federated users, or AWS services.

To enable cross-account access, you can specify an entire account or IAM entities in another account as the principal in a resource-based policy. Adding a cross-account principal to a resource-based policy is only half of establishing the trust relationship. When the principal and the resource are in different AWS accounts, an IAM administrator in the trusted account must also grant the principal entity (user or role) permission to access the resource. They grant permission by attaching an identity-based policy to the entity. However, if a resource-based policy grants access to a principal in the same account, no additional identity-based policy is required. For more information, see Cross account resource access in IAM in the *IAM User Guide*.

CloudTrail supports resource-based policies on channels used for CloudTrail Lake integrations with event sources outside of AWS. The resource-based policy for the channel defines which principal entities (accounts, users, roles, and federated users) can call `PutAuditEvents` on the channel to deliver events to the destination event data store. For more information about creating integrations with CloudTrail Lake, see Create an integration with an event source outside of AWS.

Examples

To view examples of CloudTrail resource-based policies, see AWS CloudTrail resource-based policy examples.

Policy actions for CloudTrail

Supports policy actions: Yes

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

The `Action` element of a JSON policy describes the actions that you can use to allow or deny access in a policy. Policy actions usually have the same name as the associated AWS API operation. There are some exceptions, such as *permission-only actions* that don't have a matching API operation. There are also some operations that require multiple actions in a policy. These additional actions are called *dependent actions*.

Include actions in a policy to grant permissions to perform the associated operation.

To see a list of CloudTrail actions, see Actions Defined by AWS CloudTrail in the *Service Authorization Reference*.

Policy actions in CloudTrail use the following prefix before the action:

```
cloudtrail
```

For example, to grant someone permission to list tags for a trail with the `ListTags` API operation, you include the `cloudtrail:ListTags` action in their policy. Policy statements must include either an `Action` or `NotAction` element. CloudTrail defines its own set of actions that describe tasks that you can perform with this service.

To specify multiple actions in a single statement, separate them with commas as follows:

```
"Action": [
      "cloudtrail:AddTags",
      "cloudtrail:ListTags",
      "cloudtrail:RemoveTags
```

You can specify multiple actions using wildcards (*). For example, to specify all actions that begin with the word `Get`, include the following action:

```
"Action": "cloudtrail:Get*"
```

Policy resources for CloudTrail

Supports policy resources: Yes

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

The `Resource` JSON policy element specifies the object or objects to which the action applies. Statements must include either a `Resource` or a `NotResource` element. As a best practice, specify a resource using its Amazon Resource Name (ARN). You can do this for actions that support a specific resource type, known as *resource-level permissions*.

For actions that don't support resource-level permissions, such as listing operations, use a wildcard (*) to indicate that the statement applies to all resources.

```
"Resource": "*"
```

To see a list of CloudTrail resource types and their ARNs, see Resources Defined by AWS CloudTrail in the *Service Authorization Reference*. To learn with which actions you can specify the ARN of each resource, see Actions Defined by AWS CloudTrail.

In CloudTrail, there are three resource types: trails, event data stores, and channels. Each resource has a unique Amazon Resource Name (ARN) associated with it. In a policy, you use an ARN to identify the resource that the policy applies to. CloudTrail does not currently support other resource types, which are sometimes referred to as subresources.

The CloudTrail trail resource has the following ARN:

```
arn:${Partition}:cloudtrail:${Region}:${Account}:trail/{TrailName}
```

The CloudTrail event data store resource has the following ARN:

```
arn:${Partition}:cloudtrail:${Region}:${Account}:eventdatastore/{EventDataStoreId}
```

The CloudTrail channel resource has the following ARN:

```
arn:${Partition}:cloudtrail:${Region}:${Account}:channel/{ChannelId}
```

For more information about the format of ARNs, see Amazon Resource Names (ARNs) and AWS Service Namespaces.

For example, for an AWS account with the ID *123456789012*, to specify a trail named *My-Trail* that exists in the US East (Ohio) Region in your statement, use the following ARN:

```
"Resource": "arn:aws:cloudtrail:us-east-2:123456789012:trail/My-Trail"
```

To specify all trails that belong to a specific account in that AWS Region, use the wildcard (*):

```
"Resource": "arn:aws:cloudtrail:us-east-2:123456789012:trail/*"
```

Some CloudTrail actions, such as those for creating resources, can't be performed on a specific resource. In those cases, you must use the wildcard (*).

```
"Resource": "*"
```

Many CloudTrail API actions involve multiple resources. For example, `CreateTrail` requires an Amazon S3 bucket for storing log files, so a user must have permissions to write to the bucket. To specify multiple resources in a single statement, separate the ARNs with commas.

```
"Resource": [
      "resource1",
      "resource2"
```

Policy condition keys for CloudTrail

Supports service-specific policy condition keys: No

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

The `Condition` element (or `Condition` *block*) lets you specify conditions in which a statement is in effect. The `Condition` element is optional. You can create conditional expressions that use condition operators, such as equals or less than, to match the condition in the policy with values in the request.

If you specify multiple `Condition` elements in a statement, or multiple keys in a single `Condition` element, AWS evaluates them using a logical `AND` operation. If you specify multiple values for a single condition key, AWS evaluates the condition using a logical `OR` operation. All of the conditions must be met before the statement's permissions are granted.

You can also use placeholder variables when you specify conditions. For example, you can grant an IAM user permission to access a resource only if it is tagged with their IAM user name. For more information, see IAM policy elements: variables and tags in the *IAM User Guide*.

AWS supports global condition keys and service-specific condition keys. To see all AWS global condition keys, see AWS global condition context keys in the *IAM User Guide*.

CloudTrail doesn't define its own condition keys, but it supports using some global condition keys. To see all AWS global condition keys, see AWS Global Condition Context Keys in the *IAM User Guide*.

To see a list of CloudTrail condition keys, see Condition Keys for AWS CloudTrail in the *Service Authorization Reference*. To learn with which actions and resources you can use a condition key, see Actions Defined by AWS CloudTrail.

ACLs in CloudTrail

Supports ACLs: No

Access control lists (ACLs) control which principals (account members, users, or roles) have permissions to access a resource. ACLs are similar to resource-based policies, although they do not use the JSON policy document format.

ABAC with CloudTrail

Supports ABAC (tags in policies): Partial

Attribute-based access control (ABAC) is an authorization strategy that defines permissions based on attributes. In AWS, these attributes are called *tags*. You can attach tags to IAM entities (users or roles) and to many AWS resources. Tagging entities and resources is the first step of ABAC. Then you design ABAC policies to allow operations when the principal's tag matches the tag on the resource that they are trying to access.

ABAC is helpful in environments that are growing rapidly and helps with situations where policy management becomes cumbersome.

To control access based on tags, you provide tag information in the condition element of a policy using the `aws:ResourceTag/key-name`, `aws:RequestTag/key-name`, or `aws:TagKeys` condition keys.

If a service supports all three condition keys for every resource type, then the value is **Yes** for the service. If a service supports all three condition keys for only some resource types, then the value is **Partial**.

For more information about ABAC, see What is ABAC? in the *IAM User Guide*. To view a tutorial with steps for setting up ABAC, see Use attribute-based access control (ABAC) in the *IAM User Guide*.

Although you can attach tags to CloudTrail resources, CloudTrail only supports controlling access to CloudTrail Lake event data stores and channels based on tags. You cannot control access to trails based on tags.

You can attach tags to CloudTrail resources or pass tags in a request to CloudTrail. For more information about tagging CloudTrail resources, see Creating a trail with the CloudTrail console and Creating, updating, and managing trails with the AWS CLI.

Using temporary credentials with CloudTrail

Supports temporary credentials: Yes

Some AWS services don't work when you sign in using temporary credentials. For additional information, including which AWS services work with temporary credentials, see AWS services that work with IAM in the *IAM User Guide*.

You are using temporary credentials if you sign in to the AWS Management Console using any method except a user name and password. For example, when you access AWS using your company's single sign-on (SSO) link, that process automatically creates temporary credentials. You also automatically create temporary credentials when you sign in to the console as a user and then switch roles. For more information about switching roles, see Switching to a role (console) in the *IAM User Guide*.

You can manually create temporary credentials using the AWS CLI or AWS API. You can then use those temporary credentials to access AWS. AWS recommends that you dynamically generate temporary credentials instead of using long-term access keys. For more information, see Temporary security credentials in IAM.

Forward access sessions for CloudTrail

Supports forward access sessions (FAS): Yes

When you use an IAM user or role to perform actions in AWS, you are considered a principal. When you use some services, you might perform an action that then initiates another action in a different service. FAS uses the permissions of the principal calling an AWS service, combined with the requesting AWS service to make requests to downstream services. FAS requests are only made when a service receives a request that requires interactions with other AWS services or resources to complete. In this case, you must have permissions to perform both actions. For policy details when making FAS requests, see Forward access sessions.

Service roles for CloudTrail

Supports service roles: Yes

A service role is an IAM role that a service assumes to perform actions on your behalf. An IAM administrator can create, modify, and delete a service role from within IAM. For more information, see Creating a role to delegate permissions to an AWS service in the *IAM User Guide*.

⚠ **Warning**

Changing the permissions for a service role might break CloudTrail functionality. Edit service roles only when CloudTrail provides guidance to do so.

Service-linked roles for CloudTrail

Supports service-linked roles: Yes

A service-linked role is a type of service role that is linked to an AWS service. The service can assume the role to perform an action on your behalf. Service-linked roles appear in your AWS account and are owned by the service. An IAM administrator can view, but not edit the permissions for service-linked roles.

CloudTrail supports a service-linked role for integration with AWS Organizations. This role is required for the creation of an organization trail or event data store. Organization trails and event data stores log events for all AWS accounts in an organization. For more information about creating or managing CloudTrail service-linked roles, see Using service-linked roles for AWS CloudTrail.

Identity-based policy examples for AWS CloudTrail

By default, users and roles don't have permission to create or modify CloudTrail resources. They also can't perform tasks by using the AWS Management Console, AWS Command Line Interface (AWS CLI), or AWS API. To grant users permission to perform actions on the resources that they need, an IAM administrator can create IAM policies. The administrator can then add the IAM policies to roles, and users can assume the roles.

To learn how to create an IAM identity-based policy by using these example JSON policy documents, see Creating IAM policies in the *IAM User Guide*.

For details about actions and resource types defined by CloudTrail, including the format of the ARNs for each of the resource types, see Actions, Resources, and Condition Keys for AWS CloudTrail in the *Service Authorization Reference*.

Topics

- Example: Allowing and denying actions for a specified trail
- Examples: Creating and applying policies for actions on specific trails
- Examples: Denying access to create or delete event data stores based on tags
- Using the CloudTrail console
- Allow users to view their own permissions
- Granting custom permissions for CloudTrail users

Policy best practices

Identity-based policies determine whether someone can create, access, or delete CloudTrail resources in your account. These actions can incur costs for your AWS account. When you create or edit identity-based policies, follow these guidelines and recommendations:

- **Get started with AWS managed policies and move toward least-privilege permissions** – To get started granting permissions to your users and workloads, use the *AWS managed policies* that grant permissions for many common use cases. They are available in your AWS account. We recommend that you reduce permissions further by defining AWS customer managed policies that are specific to your use cases. For more information, see AWS managed policies or AWS managed policies for job functions in the *IAM User Guide*.
- **Apply least-privilege permissions** – When you set permissions with IAM policies, grant only the permissions required to perform a task. You do this by defining the actions that can be taken on specific resources under specific conditions, also known as *least-privilege permissions*. For more information about using IAM to apply permissions, see Policies and permissions in IAM in the *IAM User Guide*.
- **Use conditions in IAM policies to further restrict access** – You can add a condition to your policies to limit access to actions and resources. For example, you can write a policy condition to specify that all requests must be sent using SSL. You can also use conditions to grant access to service actions if they are used through a specific AWS service, such as AWS CloudFormation. For more information, see IAM JSON policy elements: Condition in the *IAM User Guide*.
- **Use IAM Access Analyzer to validate your IAM policies to ensure secure and functional permissions** – IAM Access Analyzer validates new and existing policies so that the policies adhere to the IAM policy language (JSON) and IAM best practices. IAM Access Analyzer provides more than 100 policy checks and actionable recommendations to help you author secure and functional policies. For more information, see IAM Access Analyzer policy validation in the *IAM User Guide*.

- **Require multi-factor authentication (MFA)** – If you have a scenario that requires IAM users or a root user in your AWS account, turn on MFA for additional security. To require MFA when API operations are called, add MFA conditions to your policies. For more information, see Configuring MFA-protected API access in the *IAM User Guide*.

For more information about best practices in IAM, see Security best practices in IAM in the *IAM User Guide*.

CloudTrail doesn't have service-specific context keys that you can use in the `Condition` element of policy statements.

Example: Allowing and denying actions for a specified trail

The following example demonstrates a policy that allows users with the policy to view the status and configuration of a trail and start and stop logging for a trail named *`My-First-Trail`*. This trail was created in the US East (Ohio) Region (its home Region) in the AWS account with the ID *`123456789012`*.

```
{
  "Version": "2012-10-17",
  "Statement": [
      {
          "Effect": "Allow",
          "Action": [
              "cloudtrail:StartLogging",
              "cloudtrail:StopLogging",
              "cloudtrail:GetTrail",
              "cloudtrail:GetTrailStatus",
              "cloudtrail:GetEventSelectors"
          ],
          "Resource": [
              "arn:aws:cloudtrail:us-east-2:123456789012:trail/My-First-Trail"
          ]
      }
  ]
}
```

The following example demonstrates a policy that explicitly denies CloudTrail actions for any trail not named *`My-First-Trail`*.

```
{
  "Version": "2012-10-17",
  "Statement": [
      {
          "Effect": "Deny",
          "Action": [
              "cloudtrail:*"
          ],
          "NotResource": [
              "arn:aws:cloudtrail:us-east-2:123456789012:trail/My-First-Trail"
          ]
      }
  ]
}
```

Examples: Creating and applying policies for actions on specific trails

You can use permissions and policies to control a user's ability to perform specific actions on CloudTrail trails.

For example, you don't want users of your company's developer group to start or stop logging on a specific trail. However, you might want to grant them permission to perform the `DescribeTrails` and `GetTrailStatus` actions on the trail. You want the users of the developer group to perform the `StartLogging` or `StopLogging` actions on trails that they manage.

You can create two policy statements and attach them to the developer group you create in IAM. For more information about groups in IAM, see IAM Groups in the *IAM User Guide*.

In the first policy, you deny the `StartLogging` and `StopLogging` actions for the trail ARN that you specify. In the following example, the trail ARN is `arn:aws:cloudtrail:us-east-2:123456789012:trail/Example-Trail`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Stmt1446057698000",
            "Effect": "Deny",
            "Action": [
                "cloudtrail:StartLogging",
                "cloudtrail:StopLogging"
            ],
            "Resource": [
                "arn:aws:cloudtrail:us-east-2:123456789012:trail/Example-Trail"
            ]
        }
    ]
}
```

In the second policy, the `DescribeTrails` and `GetTrailStatus` actions are allowed on all CloudTrail resources:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Stmt1446072643000",
            "Effect": "Allow",
            "Action": [
                "cloudtrail:DescribeTrails",
                "cloudtrail:GetTrail",
                "cloudtrail:GetTrailStatus"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
```

If a user of the developer group tries to start or stop logging on the trail that you specified in the first policy, that user gets an access denied exception. Users of the developer group can start and stop logging on trails that they create and manage.

The following examples show that the configured developer group in an AWS CLI profile named `devgroup`. First, a user of `devgroup` runs the `describe-trails` command.

```
$ aws --profile devgroup cloudtrail describe-trails
```

The command complete successfully with the following output:

```
{
    "trailList": [
        {
            "IncludeGlobalServiceEvents": true,
            "Name": "Default",
            "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/Example-
Trail",
            "IsMultiRegionTrail": false,
            "S3BucketName": "DOC-EXAMPLE-BUCKET",
            "HomeRegion": "us-east-2"
        }
    ]
}
```

The user then runs the `get-trail-status` command on the trail that you specified in the first policy.

```
$ aws --profile devgroup cloudtrail get-trail-status --name Example-Trail
```

The command complete successfully with the following output:

```
{
    "LatestDeliveryTime": 1449517556.256,
    "LatestDeliveryAttemptTime": "2015-12-07T19:45:56Z",
    "LatestNotificationAttemptSucceeded": "",
    "LatestDeliveryAttemptSucceeded": "2015-12-07T19:45:56Z",
    "IsLogging": true,
    "TimeLoggingStarted": "2015-12-07T19:36:27Z",
    "StartLoggingTime": 1449516987.685,
    "StopLoggingTime": 1449516977.332,
    "LatestNotificationAttemptTime": "",
    "TimeLoggingStopped": "2015-12-07T19:36:17Z"
}
```

Next, a user in the `devgroup` group runs the `stop-logging` command on the same trail.

```
$ aws --profile devgroup cloudtrail stop-logging --name Example-Trail
```

The command returns an access denied exception, such as the following:

```
A client error (AccessDeniedException) occurred when calling the StopLogging operation:
 Unknown
```

The user runs the `start-logging` command on the same trail.

```
$ aws --profile devgroup cloudtrail start-logging --name Example-Trail
```

Again the command returns an access denied exception, such as the following:

```
A client error (AccessDeniedException) occurred when calling the StartLogging
 operation: Unknown
```

Examples: Denying access to create or delete event data stores based on tags

In the following policy example, permission to create an event data store with `CreateEventDataStore` is denied if at least one of the following conditions aren't met:

- The event data store doesn't have a tag key of `stage` applied to itself
- The value of the stage tag isn't `alpha`, `beta`, `gamma`, or `prod`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": "cloudtrail:CreateEventDataStore",
            "Resource": "*",
            "Condition": {
                "Null": {
                    "aws:RequestTag/stage": "true"
                }
            }
        },
        {
            "Effect": "Deny",
            "Action": "cloudtrail:CreateEventDataStore",
            "Resource": "*",
            "Condition": {
                "ForAnyValue:StringNotEquals": {
                    "aws:RequestTag/stage": [
                        "alpha",
                        "beta",
                        "gamma",
                        "prod"
                    ]
                }
            }
        }
    ]
}
```

In the following policy example, permission to delete an event data store with `DeleteEventDataStore` is denied is if the event data store has a `stage` tag with a value of `prod`. A policy like this one can help protect an event data store from accidental deletion.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": "cloudtrail:DeleteEventDataStore",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/stage": "prod"
                }
            }
        }
    ]
}
```

Using the CloudTrail console

To access the AWS CloudTrail console, you must have a minimum set of permissions. These permissions must allow you to list and view details about the CloudTrail resources in your AWS account. If you create an identity-based policy that is more restrictive than the minimum required permissions, the console won't function as intended for entities (users or roles) with that policy.

You don't need to allow minimum console permissions for users that are making calls only to the AWS CLI or the AWS API. Instead, allow access to only the actions that match the API operation that they're trying to perform.

Granting permissions for CloudTrail administration

To allow IAM roles or users to administer a CloudTrail resource, such as a trail, event data store, or channel, you must grant explicit permissions to perform the actions associated with

CloudTrail tasks. In most situations, you can use an AWS managed policy that contains predefined permissions.

> **Note**
>
> The permissions you grant to users to perform CloudTrail administration tasks aren't the same as the permissions that CloudTrail requires to deliver log files to Amazon S3 buckets or send notifications to Amazon SNS topics. For more information about those permissions, see Amazon S3 bucket policy for CloudTrail.
> If you configure integration with Amazon CloudWatch Logs, CloudTrail also requires a role that it can assume to deliver events to an Amazon CloudWatch Logs log group. You must create the role that CloudTrail uses. For more information, see Granting permission to view and configure Amazon CloudWatch Logs information on the CloudTrail console and Sending events to CloudWatch Logs.

The following AWS managed policies are available for CloudTrail:

- **AWSCloudTrail_FullAccess** – This policy provides full access to CloudTrail actions on CloudTrail resources, such as trails, event data stores, and channels. This policy provides the required permissions to create, update, and delete CloudTrail trails, event data stores, and channels.

  This policy also provides permissions to manage the Amazon S3 bucket, the log group for CloudWatch Logs, and an Amazon SNS topic for a trail. However, the `AWSCloudTrail_FullAccess` managed policy doesn't provide permissions to delete the Amazon S3 bucket, the log group for CloudWatch Logs, or an Amazon SNS topic. For information about managed policies for other AWS services, see the *AWS Managed Policy Reference Guide*.

  > **Note**
  >
  > The **AWSCloudTrail_FullAccess** policy isn't intended to be shared broadly across your AWS account. Users with this role can turn off or reconfigure the most sensitive and important auditing functions in their AWS accounts. For this reason, you must only apply this policy to account administrators. You must closely control and monitor use of this policy.

- **AWSCloudTrail_ReadOnlyAccess** – This policy grants permissions to view the CloudTrail console, including recent events and event history. This policy also allows you to view existing trails, event data stores, and channels. Roles and users with this policy can download the event history, but they can't create or update trails, event data stores, or channels.

To provide access, add permissions to your users, groups, or roles:

- Users and groups in AWS IAM Identity Center:

  Create a permission set. Follow the instructions in Create a permission set in the *AWS IAM Identity Center User Guide*.
- Users managed in IAM through an identity provider:

  Create a role for identity federation. Follow the instructions in Creating a role for a third-party identity provider (federation) in the *IAM User Guide*.
- IAM users:
  - Create a role that your user can assume. Follow the instructions in Creating a role for an IAM user in the *IAM User Guide*.
  - (Not recommended) Attach a policy directly to a user or add a user to a user group. Follow the instructions in Adding permissions to a user (console) in the *IAM User Guide*.

Additional resources

To learn more about using IAM to give identities, such as users and roles, access to resources in your account, see Getting set up with IAM and Access management for AWS resources in the *IAM User Guide*.

You don't need to allow minimum console permissions for users that are making calls only to the AWS CLI or the AWS API. Instead, allow access to only the actions that match the API operation that you're trying to perform.

Allow users to view their own permissions

This example shows how you might create a policy that allows IAM users to view the inline and managed policies that are attached to their user identity. This policy includes permissions to complete this action on the console or programmatically using the AWS CLI or AWS API.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

Granting custom permissions for CloudTrail users

CloudTrail policies grant permissions to users who work with CloudTrail. If you need to grant different permissions to users, you can attach a CloudTrail policy to an IAM group or to a user. You can edit the policy to include or exclude specific permissions. You can also create your own custom policy. Policies are JSON documents that define the actions a user is allowed to perform and the resources that the user is allowed to perform those actions on. For specific examples, see Example: Allowing and denying actions for a specified trail and Examples: Creating and applying policies for actions on specific trails.

Contents

- Full access
- Granting permission to view AWS Config information on the CloudTrail console
- Granting permission to view and configure Amazon CloudWatch Logs information on the CloudTrail console
- Additional information

Read-only access

The following example shows a policy that grants read-only access to CloudTrail trails. This is equivalent to the managed policy **AWSCloudTrail_ReadOnlyAccess**. It grants users permission to see trail information, but not to create or update trails.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "cloudtrail:Get*",
                "cloudtrail:Describe*",
                "cloudtrail:List*",
                "cloudtrail:LookupEvents"
            ],
            "Resource": "*"
        }
    ]
}
```

In the policy statements, the `Effect` element specifies whether the actions are allowed or denied. The `Action` element lists the specific actions that the user is allowed to perform. The `Resource` element lists the AWS resources the user is allowed to perform those actions on. For policies that control access to CloudTrail actions, the `Resource` element is usually set to `*`, a wildcard that means "all resources."

The values in the `Action` element correspond to the APIs that the services support. The actions are preceded by `cloudtrail:` to indicate that they refer to CloudTrail actions. You can use the `*` wildcard character in the `Action` element , such as in the following examples:

- `"Action": ["cloudtrail:*Logging"]`

This allows all CloudTrail actions that end with "Logging" (StartLogging, StopLogging).

- "Action": ["cloudtrail:*"]

  This allows all CloudTrail actions, but not actions for other AWS services.
- "Action": ["*"]

  This allows all AWS actions. This permission is suitable for a user who acts as an AWS administrator for your account.

The read-only policy doesn't grant user permission for the CreateTrail, UpdateTrail, StartLogging, and StopLogging actions. Users with this policy are not allowed to create trails, update trails, or turn logging on and off. For the list of CloudTrail actions, see the AWS CloudTrail API Reference.

Full access

The following example shows a policy that grants full access to CloudTrail. This is equivalent to the managed policy **AWSCloudTrail_FullAccess**. It grants users the permission to perform all CloudTrail actions. It also lets users log data events in Amazon S3 and AWS Lambda, manage files in Amazon S3 buckets, manage how CloudWatch Logs monitors CloudTrail log events, and manage Amazon SNS topics in the account that the user is associated with.

> **Important**
>
> The **AWSCloudTrail_FullAccess** policy or equivalent permissions are not intended to be shared broadly across your AWS account. Users with this role or equivalent access have the ability to disable or reconfigure the most sensitive and important auditing functions in their AWS accounts. For this reason, this policy should be applied only to account administrators, and use of this policy should be closely controlled and monitored.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "sns:AddPermission",
                "sns:CreateTopic",
                "sns:SetTopicAttributes",
                "sns:GetTopicAttributes"
            ],
            "Resource": [
                "arn:aws:sns:*:*:aws-cloudtrail-logs*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "sns:ListTopics"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:CreateBucket",
                "s3:PutBucketPolicy"
            ],
            "Resource": [
                "arn:aws:s3:::aws-cloudtrail-logs*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListAllMyBuckets",
                "s3:GetBucketLocation",
                "s3:GetBucketPolicy"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "cloudtrail:*",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogGroup"
            ],
            "Resource": [
                "arn:aws:logs:*:*:log-group:aws-cloudtrail-logs*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:ListRoles",
                "iam:GetRolePolicy",
                "iam:GetUser"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:PassRole"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": "cloudtrail.amazonaws.com"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "kms:CreateKey",
                "kms:CreateAlias",
                "kms:ListKeys",
                "kms:ListAliases"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "lambda:ListFunctions"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "dynamodb:ListGlobalTables",
                "dynamodb:ListTables"
            ],
            "Resource": "*"
        }
    ]
}
```

Granting permission to view AWS Config information on the CloudTrail console

You can view event information on the CloudTrail console, including resources that are related to that event. For these resources, you can choose the AWS Config icon to view the timeline for that resource in the AWS Config console. Attach this policy to your users to grant them read-only AWS Config access. The policy doesn't grant them permission to change settings in AWS Config.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": [
            "config:Get*",
            "config:Describe*",
            "config:List*"
        ],
        "Resource": "*"
    }]
}
```

For more information, see Viewing resources referenced with AWS Config.

Granting permission to view and configure Amazon CloudWatch Logs information on the CloudTrail console

You can view and configure delivery of events to CloudWatch Logs in the CloudTrail console if you have sufficient permissions. These are permissions that may be beyond those granted for CloudTrail administrators. Attach this policy to administrators who will configure and manage CloudTrail integration with CloudWatch Logs. The policy doesn't grant them permissions in CloudTrail or in CloudWatch Logs directly, but instead grants the permissions required to create and configure the role CloudTrail will assume to successfully deliver events to your CloudWatch Logs group.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": [
            "iam:CreateRole",
            "iam:PutRolePolicy",
            "iam:AttachRolePolicy",
            "iam:ListRoles",
            "iam:GetRolePolicy",
            "iam:GetUser"
        ],
        "Resource": "*"
    }]
}
```

For more information, see Monitoring CloudTrail Log Files with Amazon CloudWatch Logs.

Additional information

To learn more about using IAM to give identities, such as users and roles, access to resources in your account, see Getting started and Access management for AWS resources in the *IAM User Guide*.

AWS CloudTrail resource-based policy examples

CloudTrail supports resource-based permissions policies for CloudTrail channels used for CloudTrail Lake integrations. For more information about creating integrations with CloudTrail Lake, see Create an integration with an event source outside of AWS.

The information required for the policy is determined by the integration type.

- For a direction integration, CloudTrail requires the policy to contain the partner's AWS account IDs, and requires you to enter the unique external ID provided by the partner. CloudTrail automatically adds the partner's AWS account IDs to the resource policy when you create an integration using the CloudTrail console. Refer to the partner's documentation to learn how to get the AWS account numbers required for the policy.
- For a solution integration, you must specify at least one AWS account ID as principal, and can optionally enter an external ID to prevent against confused deputy.

The following are requirements for the resource-based policy:

- The resource ARN defined in the policy must match the channel ARN the policy is attached to.
- The policy contains only one action: `cloudtrail-data:PutAuditEvents`
- The policy contains at least one statement. The policy can have a maximum of 20 statements.
- Each statement contains at least one principal. A statement can have a maximum of 50 principals.

The channel owner can call the `PutAuditEvents` API on the channel unless the policy denies the owner access to the resource.

Topics

- Example: Providing channel access to principals
- Example: Using an external ID to prevent against confused deputy

Example: Providing channel access to principals

The following example grants permissions to the principals with the ARNs `arn:aws:iam::111122223333:root`, `arn:aws:iam::444455556666:root`, and `arn:aws:iam::123456789012:root` to call the PutAuditEvents API on the CloudTrail channel with the ARN `arn:aws:cloudtrail:us-east-1:777788889999:channel/EXAMPLE-80b5-40a7-ae65-6e099392355b`.

```
{
    "Version": "2012-10-17",
    "Statement":
    [
        {
            "Sid": "ChannelPolicy",
            "Effect": "Allow",
            "Principal":
            {
                "AWS":
                [
                    "arn:aws:iam::111122223333:root",
                    "arn:aws:iam::444455556666:root",
                    "arn:aws:iam::123456789012:root"
                ]
            },
            "Action": "cloudtrail-data:PutAuditEvents",
            "Resource": "arn:aws:cloudtrail:us-east-1:777788889999:channel/
EXAMPLE-80b5-40a7-ae65-6e099392355b"
        }
    ]
}
```

Example: Using an external ID to prevent against confused deputy

The following example uses an external ID to address and prevent against confused deputy. The confused deputy problem is a security issue where an entity that doesn't have permission to perform an action can coerce a more-privileged entity to perform the action.

The integration partner creates the external ID to use in the policy. Then, it provides the external ID to you as part of creating the integration. The value can be any unique string, such as a passphrase or account number.

The example grants permissions to the principals with the ARNs `arn:aws:iam::111122223333:root`, `arn:aws:iam::444455556666:root`, and `arn:aws:iam::123456789012:root` to call the PutAuditEvents API on the CloudTrail channel resource if the call to the `PutAuditEvents` API includes the external ID value defined in the policy.

```
{
    "Version": "2012-10-17",
    "Statement":
    [
        {
            "Sid": "ChannelPolicy",
            "Effect": "Allow",
            "Principal":
            {
                "AWS":
                [
                    "arn:aws:iam::111122223333:root",
                    "arn:aws:iam::444455556666:root",
                    "arn:aws:iam::123456789012:root"
                ]
            },
            "Action": "cloudtrail-data:PutAuditEvents",
            "Resource": "arn:aws:cloudtrail:us-east-1:777788889999:channel/
EXAMPLE-80b5-40a7-ae65-6e099392355b",
            "Condition":
            {
                "StringEquals":
                {
                    "cloudtrail:ExternalId": "uniquePartnerExternalID"
                }
            }
        }
    ]
}
```

Amazon S3 bucket policy for CloudTrail

By default, Amazon S3 buckets and objects are private. Only the resource owner (the AWS account that created the bucket) can access the bucket and objects it contains. The resource owner can grant access permissions to other resources and users by writing an access policy.

To create or modify an Amazon S3 bucket to receive log files for an organization trail, you must change the bucket policy. For more information, see Creating a trail for an organization with the AWS CLI.

To deliver log files to an S3 bucket, CloudTrail must have the required permissions, and it cannot be configured as a Requester Pays bucket.

CloudTrail adds the following fields in the policy for you:

- The allowed SIDs
- The bucket name
- The service principal name for CloudTrail
- The name of the folder where the log files are stored, including the bucket name, a prefix (if you specified one), and your AWS account ID

As a security best practice, add an `aws:SourceArn` condition key to the Amazon S3 bucket policy. The IAM global condition key `aws:SourceArn` helps ensure that CloudTrail writes to the S3 bucket only for a specific trail or trails. The value of `aws:SourceArn` is always the ARN of the trail (or array of trail ARNs) that is using the bucket to store logs. Be sure to add the `aws:SourceArn` condition key to S3 bucket policies for existing trails.

The following policy allows CloudTrail to write log files to the bucket from supported AWS Regions. Replace *`DOC-EXAMPLE-BUCKET`*, *`[optionalPrefix]/`*, *`myAccountID`*, *`region`*, and *`trailName`* with the appropriate values for your configuration.

S3 bucket policy

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AWSCloudTrailAclCheck20150319",
            "Effect": "Allow",
            "Principal": {"Service": "cloudtrail.amazonaws.com"},
            "Action": "s3:GetBucketAcl",
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
            "Condition": {
                "StringEquals": {
                    "aws:SourceArn":
 "arn:aws:cloudtrail:region:myAccountID:trail/trailName"
                }
            }
        },
        {
            "Sid": "AWSCloudTrailWrite20150319",
            "Effect": "Allow",
            "Principal": {"Service": "cloudtrail.amazonaws.com"},
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-
BUCKET/[optionalPrefix]/AWSLogs/myAccountID/*",
            "Condition": {
                "StringEquals": {
                    "s3:x-amz-acl": "bucket-owner-full-control",
                    "aws:SourceArn":
 "arn:aws:cloudtrail:region:myAccountID:trail/trailName"
                }
            }
        }
    ]
}
```

For more information about AWS Regions, see CloudTrail supported Regions.

Contents

Specifying an existing bucket for CloudTrail log delivery

If you specified an existing S3 bucket as the storage location for log file delivery, you must attach a policy to the bucket that allows CloudTrail to write to the bucket.

> **Note**
>
> As a best practice, use a dedicated S3 bucket for CloudTrail logs.

To add the required CloudTrail policy to an Amazon S3 bucket

1. Open the Amazon S3 console at https://console.aws.amazon.com/s3/.
2. Choose the bucket where you want CloudTrail to deliver your log files, and then choose **Permissions**.
3. Choose **Edit**.
4. Copy the S3 bucket policy to the **Bucket Policy Editor** window. Replace the placeholders in italics with the names of your bucket, prefix, and account number. If you specified a prefix when you created your trail, include it here. The prefix is an optional addition to the S3 object key that creates a folder-like organization in your bucket.

   > **Note**
   >
   > If the existing bucket already has one or more policies attached, add the statements for CloudTrail access to that policy or policies. Evaluate the resulting set of permissions to be sure that they are appropriate for the users who will access the bucket.

Receiving log files from other accounts

You can configure CloudTrail to deliver log files from multiple AWS accounts to a single S3 bucket. For more information, see Receiving CloudTrail log files from multiple accounts.

Create or update an Amazon S3 bucket to use to store the log files for an organization trail

You must specify an Amazon S3 bucket to receive the log files for an organization trail. This bucket must have a policy that allows CloudTrail to put the log files for the organization into the bucket.

The following is an example policy for an Amazon S3 bucket named *DOC-EXAMPLE-BUCKET*, which is owned by the organization's management account. Replace *DOC-EXAMPLE-BUCKET*, *region*, *managementAccountID*, *trailName*, and *o-organizationID* with the values for your organization

This bucket policy contains three statements.

- The first statement allows CloudTrail to call the Amazon S3 `GetBucketAcl` action on the Amazon S3 bucket.
- The second statement allows logging in the event the trail is changed from an organization trail to a trail for that account only.
- The third statement allows logging for an organization trail.

The example policy includes an `aws:SourceArn` condition key for the Amazon S3 bucket policy. The IAM global condition key `aws:SourceArn` helps ensure that CloudTrail writes to the S3 bucket only for a specific trail or trails. In an organization trail, the value of `aws:SourceArn` must be a trail ARN that is owned by the management account, and uses the management account ID.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AWSCloudTrailAclCheck20150319",
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "cloudtrail.amazonaws.com"
                ]
            },
            "Action": "s3:GetBucketAcl",
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
            "Condition": {
                "StringEquals": {
                    "aws:SourceArn":
 "arn:aws:cloudtrail:region:managementAccountID:trail/trailName"
                }
            }
        },
        {
            "Sid": "AWSCloudTrailWrite20150319",
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "cloudtrail.amazonaws.com"
                ]
            },
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/AWSLogs/managementAccountID/
*",
            "Condition": {
                "StringEquals": {
                    "s3:x-amz-acl": "bucket-owner-full-control",
                    "aws:SourceArn":
 "arn:aws:cloudtrail:region:managementAccountID:trail/trailName"
                }
            }
        },
        {
            "Sid": "AWSCloudTrailOrganizationWrite20150319",
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "cloudtrail.amazonaws.com"
                ]
            },
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/AWSLogs/o-organizationID/*",
            "Condition": {
                "StringEquals": {
                    "s3:x-amz-acl": "bucket-owner-full-control",
                    "aws:SourceArn":
 "arn:aws:cloudtrail:region:managementAccountID:trail/trailName"
                }
            }
        }
    ]
}
```

This example policy does not allow any users from member accounts to access the log files created for the organization. By default, organization log files are accessible only to the management account. For information about how to allow read access to the Amazon S3 bucket for IAM users in member accounts, see Sharing CloudTrail log files between AWS accounts.

Troubleshooting the Amazon S3 bucket policy

The following sections describe how to troubleshoot the S3 bucket policy.

Common Amazon S3 policy configuration errors

When you create a new bucket as part of creating or updating a trail, CloudTrail attaches the required permissions to your bucket. The bucket policy uses the service principal name, `"cloudtrail.amazonaws.com"`, which allows CloudTrail to deliver logs for all Regions.

If CloudTrail is not delivering logs for a Region, it's possible that your bucket has an older policy that specifies CloudTrail account IDs for each Region. This policy gives CloudTrail permission to deliver logs only for the Regions specified.

As a best practice, update the policy to use a permission with the CloudTrail service principal. To do this, replace the account ID ARNs with the service principal name: `"cloudtrail.amazonaws.com"`. This gives CloudTrail permission to deliver logs for current and new Regions. As a security best practice, add an `aws:SourceArn` or `aws:SourceAccount` condition key to the Amazon S3 bucket policy. This helps prevent unauthorized account access to your S3 bucket. If you have existing trails, be sure to add one or more condition keys. The following example shows a recommended policy configuration. Replace *`DOC-EXAMPLE-BUCKET`*, *`[optionalPrefix]/`*, *`myAccountID`*, *`region`*, and *`trailName`* with the appropriate values for your configuration.

Example Example bucket policy with service principal name

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AWSCloudTrailAclCheck20150319",
            "Effect": "Allow",
            "Principal": {"Service": "cloudtrail.amazonaws.com"},
            "Action": "s3:GetBucketAcl",
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
            "Condition": {
                "StringEquals": {
                    "aws:SourceArn":
 "arn:aws:cloudtrail:region:myAccountID:trail/trailName"
                }
            }
        },
        {
            "Sid": "AWSCloudTrailWrite20150319",
            "Effect": "Allow",
            "Principal": {"Service": "cloudtrail.amazonaws.com"},
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-
BUCKET/[optionalPrefix]/AWSLogs/myAccountID/*",
            "Condition": {"StringEquals": {
                "s3:x-amz-acl": "bucket-owner-full-control",
                "aws:SourceArn":
 "arn:aws:cloudtrail:region:myAccountID:trail/trailName"
                }
            }
        }
    ]
}
```

Changing a prefix for an existing bucket

If you try to add, modify, or remove a log file prefix for an S3 bucket that receives logs from a trail, you might see the error: **There is a problem with the bucket policy**. A bucket policy with an incorrect prefix can prevent your trail from delivering logs to the bucket. To resolve this issue, use the Amazon S3 console to update the prefix in the bucket policy, and then use the CloudTrail console to specify the same prefix for the bucket in the trail.

To update the log file prefix for an Amazon S3 bucket

1. Open the Amazon S3 console at https://console.aws.amazon.com/s3/.
2. Choose the bucket for which you want to modify the prefix, and then choose **Permissions**.
3. Choose **Edit**.

4. In the bucket policy, under the s3:PutObject action, edit the Resource entry to add, modify, or remove the log file *prefix/* as needed.

   ```
   "Action": "s3:PutObject",
         "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/prefix/AWSLogs/myAccountID/*",
   ```

5. Choose **Save**.
6. Open the CloudTrail console at https://console.aws.amazon.com/cloudtrail/.
7. Choose your trail and for **Storage location**, click the pencil icon to edit the settings for your bucket.
8. For **S3 bucket**, choose the bucket with the prefix you are changing.
9. For **Log file prefix**, update the prefix to match the prefix that you entered in the bucket policy.
10. Choose **Save**.

Additional resources

For more information about S3 buckets and policies, see Using bucket policies in the *Amazon Simple Storage Service User Guide*.

Amazon S3 bucket policy for CloudTrail Lake query results

By default, Amazon S3 buckets and objects are private. Only the resource owner (the AWS account that created the bucket) can access the bucket and objects it contains. The resource owner can grant access permissions to other resources and users by writing an access policy.

To deliver CloudTrail Lake query results to an S3 bucket, CloudTrail must have the required permissions, and it cannot be configured as a Requester Pays bucket.

CloudTrail adds the following fields in the policy for you:

- The allowed SIDs
- The bucket name
- The service principal name for CloudTrail

As a security best practice, add an aws:SourceArn condition key to the Amazon S3 bucket policy. The IAM global condition key aws:SourceArn helps ensure that CloudTrail writes to the S3 bucket only for the event data store.

The following policy allows CloudTrail to deliver query results to the bucket from supported AWS Regions. Replace *DOC-EXAMPLE-BUCKET*, *myAccountID*, and *myQueryRunningRegion* with the appropriate values for your configuration. The *myAccountID* is the AWS account ID used for CloudTrail, which may not be the same as the AWS account ID for the S3 bucket.

> **Note**
>
> If your bucket policy includes a statement for a KMS key, we recommend using a fully qualified KMS key ARN. If you use a KMS key alias instead, AWS KMS resolves the key within the requester's account. This behavior can result in data that's encrypted with a KMS key that belongs to the requester, and not the bucket owner.
> If this is an organization event data store, the event data store ARN must include the AWS account ID for the management account. This is because the management account maintains ownership of all organization resources.

S3 bucket policy

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AWSCloudTrailLake1",
            "Effect": "Allow",
            "Principal": {"Service": "cloudtrail.amazonaws.com"},
            "Action": [
                "s3:PutObject*",
                "s3:Abort*"
            ],
            "Resource": [
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
            ],
            "Condition": {
                "StringLike": {
                    "aws:sourceAccount": "myAccountID",
                    "aws:sourceArn":
 "arn:aws:cloudtrail:myQueryRunningRegion:myAccountID:eventdatastore/*"
                }
            }
        },
        {
            "Sid": "AWSCloudTrailLake2",
            "Effect": "Allow",
            "Principal": {"Service":"cloudtrail.amazonaws.com"},
            "Action": "s3:GetBucketAcl",
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
            "Condition": {
                "StringLike": {
                    "aws:sourceAccount": "myAccountID",
                    "aws:sourceArn":
 "arn:aws:cloudtrail:myQueryRunningRegion:myAccountID:eventdatastore/*"
                }
            }
        }
    ]
}
```

Contents

Specifying an existing bucket for CloudTrail Lake query results

If you specified an existing S3 bucket as the storage location for CloudTrail Lake query results delivery, you must attach a policy to the bucket that allows CloudTrail to deliver the query results to the bucket.

> **Note**
>
> As a best practice, use a dedicated S3 bucket for CloudTrail Lake query results.

To add the required CloudTrail policy to an Amazon S3 bucket

1. Open the Amazon S3 console at https://console.aws.amazon.com/s3/.
2. Choose the bucket where you want CloudTrail to deliver your Lake query results, and then choose **Permissions**.
3. Choose **Edit**.

4. Copy the S3 bucket policy for query results to the **Bucket Policy Editor** window. Replace the placeholders in italics with the names of your bucket, Region, and account ID.

> **Note**
>
> If the existing bucket already has one or more policies attached, add the statements for CloudTrail access to that policy or policies. Evaluate the resulting set of permissions to be sure that they are appropriate for the users who access the bucket.

Additional resources

For more information about S3 buckets and policies, see Using bucket policies in the *Amazon Simple Storage Service User Guide*.

Amazon SNS topic policy for CloudTrail

To send notifications to an SNS topic, CloudTrail must have the required permissions. CloudTrail automatically attaches the required permissions to the topic when you create an Amazon SNS topic as part of creating or updating a trail in the CloudTrail console.

> **Important**
>
> As a security best practice, to restrict access to your SNS topic, we strongly recommend that after you create or update a trail to send SNS notifications, you manually edit the IAM policy that is attached to the SNS topic to add condition keys. For more information, see the section called "Security best practice for SNS topic policy" in this topic.

CloudTrail adds the following statement to the policy for you with the following fields:

- The allowed SIDs.
- The service principal name for CloudTrail.
- The SNS topic, including Region, account ID, and topic name.

The following policy allows CloudTrail to send notifications about log file delivery from supported Regions. For more information, see CloudTrail supported Regions. This is the default policy that is attached to a new or existing SNS topic policy when you create or update a trail, and choose to enable SNS notifications.

SNS topic policy

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AWSCloudTrailSNSPolicy20131101",
            "Effect": "Allow",
            "Principal": {
                "Service": "cloudtrail.amazonaws.com"
            },
            "Action": "SNS:Publish",
            "Resource": "arn:aws:sns:region:SNSTopicOwnerAccountId:SNSTopicName"
        }
    ]
}
```

To use an AWS KMS-encrypted Amazon SNS topic to send notifications, you must also enable compatibility between the event source (CloudTrail) and the encrypted topic by adding the following statement to the policy of the AWS KMS key.

KMS key policy

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "cloudtrail.amazonaws.com"
            },
            "Action": [
                "kms:GenerateDataKey*",
                "kms:Decrypt"
            ],
            "Resource": "*"
        }
    ]
}
```

For more information, see Enable Compatibility between Event Sources from AWS Services and Encrypted Topics.

Contents

Security best practice for SNS topic policy

By default, the IAM policy statement that CloudTrail attaches to your Amazon SNS topic allows the CloudTrail service principal to publish to an SNS topic, identified by an ARN. To help prevent an attacker from gaining access to your SNS topic, and sending notifications on behalf of CloudTrail to topic recipients, manually edit your CloudTrail SNS topic policy to add an `aws:SourceArn` condition key to the policy statement attached by CloudTrail. The value of this key is the ARN of the trail, or an array of trail ARNs that are using the SNS topic. Because it includes both the specific trail ID and the ID of the account that owns the trail, it restricts SNS topic access to only those accounts that have permission to manage the trail. Before you add condition keys to your SNS topic policy, get the SNS topic name from your trail's settings in the CloudTrail console.

The `aws:SourceAccount` condition key is also supported, but is not recommended.

To add the `aws:SourceArn` condition key to your SNS topic policy

1. Open the Amazon SNS console at https://console.aws.amazon.com/sns/v3/home.
2. In the navigation pane, choose **Topics**.
3. Choose the SNS topic that is shown in your trail settings, and then choose **Edit**.
4. Expand **Access policy**.
5. In the **Access policy** JSON editor, look for a block that resembles the following example.

```
    {
      "Sid": "AWSCloudTrailSNSPolicy20150319",
      "Effect": "Allow",
      "Principal": {
        "Service": "cloudtrail.amazonaws.com"
      },
      "Action": "SNS:Publish",
      "Resource": "arn:aws:sns:us-west-2:111122223333:aws-cloudtrail-
logs-111122223333-61bbe496"
    }
```

6. Add a new block for a condition, `aws:SourceArn`, as shown in the following example. The value of `aws:SourceArn` is the ARN of the trail about which you are sending notifications to SNS.

```
      {
        "Sid": "AWSCloudTrailSNSPolicy20150319",
        "Effect": "Allow",
        "Principal": {
          "Service": "cloudtrail.amazonaws.com"
        },
        "Action": "SNS:Publish",
        "Resource": "arn:aws:sns:us-west-2:111122223333:aws-cloudtrail-
logs-111122223333-61bbe496",
        "Condition": {
          "StringEquals": {
            "aws:SourceArn": "arn:aws:cloudtrail:us-west-2:123456789012:trail/Trail3"
          }
        }
      }
```

7. When you are finished editing the SNS topic policy, choose **Save changes**.

To add the `aws:SourceAccount` condition key to your SNS topic policy

1. Open the Amazon SNS console at https://console.aws.amazon.com/sns/v3/home.
2. In the navigation pane, choose **Topics**.
3. Choose the SNS topic that is shown in your trail settings, and then choose **Edit**.
4. Expand **Access policy**.
5. In the **Access policy** JSON editor, look for a block that resembles the following example.

```
      {
        "Sid": "AWSCloudTrailSNSPolicy20150319",
        "Effect": "Allow",
        "Principal": {
          "Service": "cloudtrail.amazonaws.com"
        },
        "Action": "SNS:Publish",
        "Resource": "arn:aws:sns:us-west-2:111122223333:aws-cloudtrail-
  logs-111122223333-61bbe496"
      }
```

6. Add a new block for a condition, `aws:SourceAccount`, as shown in the following example. The value of `aws:SourceAccount` is the ID of the account that owns the CloudTrail trail. This example restricts access to the SNS topic to only those users who can sign in to the AWS account 123456789012.

```
    {
      "Sid": "AWSCloudTrailSNSPolicy20150319",
      "Effect": "Allow",
      "Principal": {
        "Service": "cloudtrail.amazonaws.com"
      },
      "Action": "SNS:Publish",
      "Resource": "arn:aws:sns:us-west-2:111122223333:aws-cloudtrail-
logs-111122223333-61bbe496",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "123456789012"
        }
      }
    }
```

7. When you are finished editing the SNS topic policy, choose **Save changes**.

Specifying an existing topic for sending notifications

You can manually add the permissions for an Amazon SNS topic to your topic policy in the Amazon SNS console and then specify the topic in the CloudTrail console.

To manually update an SNS topic policy

1. Open the Amazon SNS console at https://console.aws.amazon.com/sns/v3/home.
2. Choose **Topics** and then choose the topic.
3. Choose **Edit** and then scroll down to **Access policy**.

4. Add the statement from SNS topic policy with the appropriate values for the Region, account ID, and topic name.
5. If your topic is an encrypted topic, you must allow CloudTrail to have `kms:GenerateDataKey*` and the `kms:Decrypt` permissions. For more information, see Encrypted SNS topic KMS key policy.
6. Choose **Save changes**.
7. Return to the CloudTrail console and specify the topic for the trail.

Troubleshooting the SNS topic policy

The following sections describe how to troubleshoot the SNS topic policy.

Scenarios:

- CloudTrail is not sending notifications for a Region
- CloudTrail is not sending notifications for a member account in an organization

CloudTrail is not sending notifications for a Region

When you create a new topic as part of creating or updating a trail, CloudTrail attaches the required permissions to your topic. The topic policy uses the service principal name, `"cloudtrail.amazonaws.com"`, which allows CloudTrail to send notifications for all Regions.

If CloudTrail is not sending notifications for a Region, it's possible that your topic has an older policy that specifies CloudTrail account IDs for each Region. This policy gives CloudTrail permission to send notifications only for the Regions specified.

The following topic policy allows CloudTrail to send notifications for the specified nine Regions only:

Example topic policy with account IDs

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Sid": "AWSCloudTrailSNSPolicy20131101",
        "Effect": "Allow",
        "Principal": {"AWS": [
            "arn:aws:iam::903692715234:root",
            "arn:aws:iam::035351147821:root",
            "arn:aws:iam::859597730677:root",
            "arn:aws:iam::814480443879:root",
            "arn:aws:iam::216624486486:root",
            "arn:aws:iam::086441151436:root",
            "arn:aws:iam::388731089494:root",
            "arn:aws:iam::284668455005:root",
            "arn:aws:iam::113285607260:root"
        ]},
        "Action": "SNS:Publish",
        "Resource": "aws:arn:sns:us-east-1:123456789012:myTopic"
    }]
}
```

This policy uses a permission based on individual CloudTrail account IDs. To deliver logs for a new Region, you must manually update the policy to include the CloudTrail account ID for that Region. For example, because CloudTrail added support for the US East (Ohio) Region, you must update the policy to add the account ID ARN for that Region: `"arn:aws:iam::475085895292:root"`.

As a best practice, update the policy to use a permission with the CloudTrail service principal. To do this, replace the account ID ARNs with the service principal name: `"cloudtrail.amazonaws.com"`.

This gives CloudTrail permission to send notifications for current and new Regions. The following is an updated version of the previous policy:

Example topic policy with service principal name

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Sid": "AWSCloudTrailSNSPolicy20131101",
        "Effect": "Allow",
        "Principal": {"Service": "cloudtrail.amazonaws.com"},
        "Action": "SNS:Publish",
        "Resource": "arn:aws:sns:us-west-2:123456789012:myTopic"
    }]
}
```

Verify that the policy has the correct values:

- In the `Resource` field, specify the account number of the topic owner. For topics that you create, specify your account number.
- Specify the appropriate values for the Region and SNS topic name.

CloudTrail is not sending notifications for a member account in an organization

When a member account with an AWS Organizations organization trail is not sending Amazon SNS notifications, there could be an issue with the configuration of the SNS topic policy. CloudTrail creates organization trails in member accounts even if a resource validation fails, for example, the organization trail's SNS topic does not include all member account IDs. If the SNS topic policy is incorrect, an authorization failure occurs.

To check whether a trail's SNS topic policy has an authorization failure:

- From the CloudTrail console, check the trail's details page. If there's an authorization failure, the details page includes a warning `SNS authorization failed` and indicates to fix the SNS topic policy.
- From the AWS CLI, run the get-trail-status command. If there's an authorization failure, the command output includes the `LastNotificationError` field with a value of `AuthorizationError`.

Additional resources

For more information about SNS topics and subscribing to them, see the Amazon Simple Notification Service Developer Guide.

Troubleshooting AWS CloudTrail identity and access

Use the following information to help you diagnose and fix common issues that you might encounter when working with CloudTrail and IAM.

Topics

- I am getting a NoManagementAccountSLRExistsException exception when I try to create an organization trail or event data store

I am not authorized to perform an action in CloudTrail

If you receive an error that you're not authorized to perform an action, your policies must be updated to allow you to perform the action.

The following example error occurs when the `mateojackson` IAM user tries to use the console to view details about a fictional *`my-example-widget`* resource but doesn't have the fictional `cloudtrail:`*`GetWidget`* permissions.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 cloudtrail:GetWidget on resource: my-example-widget
```

In this case, the policy for the `mateojackson` user must be updated to allow access to the *`my-example-widget`* resource by using the `cloudtrail:`*`GetWidget`* action.

If you need help, contact your AWS administrator. Your administrator is the person who provided you with your sign-in credentials.

If the AWS Management Console tells you that you're not authorized to perform an action, then you must contact your administrator for assistance. Your administrator is the person that provided you with your sign-in credentials.

The following example error occurs when the `mateojackson` IAM user tries to use the console to view details about a trail but doesn't have either the appropriate CloudTrail managed policy (**AWSCloudTrail_FullAccess** or **AWSCloudTrail_ReadOnlyAccess**) or the equivalent permissions applied to his account.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 cloudtrail:GetTrailStatus on resource: My-Trail
```

In this case, Mateo asks his administrator to update his policies to allow him to access trail information and status in the console.

If you sign in with an IAM user or role that has the **AWSCloudTrail_FullAccess** managed policy or its equivalent permissions, and you can't configure AWS Config or Amazon CloudWatch Logs integration with a trail, you might be missing the required permissions for integration with those services. For more information, see Granting permission to view AWS Config information on the CloudTrail console and Granting permission to view and configure Amazon CloudWatch Logs information on the CloudTrail console.

I am not authorized to perform `iam:PassRole`

If you receive an error that you're not authorized to perform the `iam:PassRole` action, your policies must be updated to allow you to pass a role to CloudTrail.

Some AWS services allow you to pass an existing role to that service instead of creating a new service role or service-linked role. To do this, you must have permissions to pass the role to the service.

The following example error occurs when an IAM user named `marymajor` tries to use the console to perform an action in CloudTrail. However, the action requires the service to have permissions that are granted by a service role. Mary does not have permissions to pass the role to the service.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

In this case, Mary's policies must be updated to allow her to perform the `iam:PassRole` action.

If you need help, contact your AWS administrator. Your administrator is the person who provided you with your sign-in credentials.

I want to allow people outside of my AWS account to access my CloudTrail resources

You can create a role and share CloudTrail information between multiple AWS accounts. For more information, see Sharing CloudTrail log files between AWS accounts.

You can create a role that users in other accounts or people outside of your organization can use to access your resources. You can specify who is trusted to assume the role. For services that support resource-based policies or access control lists (ACLs), you can use those policies to grant people access to your resources.

To learn more, consult the following:

- To learn whether CloudTrail supports these features, see How AWS CloudTrail works with IAM.

- To learn how to provide access to your resources across AWS accounts that you own, see Providing access to an IAM user in another AWS account that you own in the *IAM User Guide*.
- To learn how to provide access to your resources to third-party AWS accounts, see Providing access to AWS accounts owned by third parties in the *IAM User Guide*.
- To learn how to provide access through identity federation, see Providing access to externally authenticated users (identity federation) in the *IAM User Guide*.
- To learn the difference between using roles and resource-based policies for cross-account access, see Cross account resource access in IAM in the *IAM User Guide*.

I am not authorized to perform `iam:PassRole`

If you receive an error that you're not authorized to perform the `iam:PassRole` action, your policies must be updated to allow you to pass a role to CloudTrail.

Some AWS services allow you to pass an existing role to that service instead of creating a new service role or service-linked role. To do this, you must have permissions to pass the role to the service.

The following example error occurs when an IAM user named `marymajor` tries to use the console to perform an action in CloudTrail. However, the action requires the service to have permissions that are granted by a service role. Mary does not have permissions to pass the role to the service.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

In this case, Mary's policies must be updated to allow her to perform the `iam:PassRole` action.

If you need help, contact your AWS administrator. Your administrator is the person who provided you with your sign-in credentials.

I am getting a `NoManagementAccountSLRExistsException` exception when I try to create an organization trail or event data store

The `NoManagementAccountSLRExistsException` exception is thrown when the management account does not have a service-linked role. When you add a delegated administrator using the AWS Organizations AWS CLI or API operation, the service-linked role doesn't get created if it does not exist.

When you use your organization's management account to add a delegated administrator or create an organization trail or event data store in the CloudTrail console, or by using the AWS CLI or CloudTrail API, CloudTrail automatically creates a service-linked role for your management account if one does not already exist.

If you haven't added a delegated administrator, use the CloudTrail console, AWS CLI or CloudTrail API to add the delegated administrator. For more information about adding a delegated administrator, see Add a CloudTrail delegated administrator and RegisterOrganizationDelegatedAdmin (API).

If you've already added the delegated administrator, use the management account to create the organization trail or event data store in the CloudTrail console, or by using the AWS CLI or CloudTrail API. For more information about creating an organization trail, see Creating a trail for your organization in the console, Creating a trail for an organization with the AWS CLI, and CreateTrail (API).

Using service-linked roles for AWS CloudTrail

AWS CloudTrail uses AWS Identity and Access Management (IAM) service-linked roles. A service-linked role is a unique type of IAM role that is linked directly to CloudTrail. Service-linked roles are predefined by CloudTrail and include all the permissions that the service requires to call other AWS services on your behalf.

A service-linked role makes setting up CloudTrail easier because you don’t have to manually add the necessary permissions. CloudTrail defines the permissions of its service-linked roles, and unless defined otherwise, only CloudTrail can assume its roles. The defined permissions include the trust policy and the permissions policy, and that permissions policy cannot be attached to any other IAM entity.

For information about other services that support service-linked roles, see AWS Services That Work with IAM and look for the services that have **Yes** in the **Service-Linked Role** column. Choose a **Yes** with a link to view the service-linked role documentation for that service.

Service-linked role permissions for CloudTrail

CloudTrail uses the service-linked role named **AWSServiceRoleForCloudTrail** – This service linked role is used for supporting organization trails and organization event data stores.

The AWSServiceRoleForCloudTrail service-linked role trusts the following services to assume the role:

- `cloudtrail.amazonaws.com`

This role is used to support the creation and management of CloudTrail organization trails and CloudTrail Lake organization event data stores in CloudTrail. For more information, see Creating a trail for an organization.

The CloudTrailServiceRolePolicy policy attached to the role allows CloudTrail to complete the following actions on the specified resources:

- Actions on all CloudTrail resources:
  - `All`
- Actions on all AWS Organizations resources:
  - `organizations:DescribeAccount`
  - `organizations:DescribeOrganization`
  - `organizations:ListAccounts`
  - `organizations:ListAWSServiceAccessForOrganization`
- Actions on all Organizations resources for the CloudTrail service principal to list the delegated administrators for the organization:
  - `organizations:ListDelegatedAdministrators`
- Actions for disabling Lake federation on an organization event data store:
  - `glue:DeleteTable`
  - `lakeformation:DeRegisterResource`

You must configure permissions to allow an IAM entity (such as a user, group, or role) to create, edit, or delete a service-linked role. For more information, see Service-Linked Role Permissions in the *IAM User Guide*.

Creating a service-linked role for CloudTrail

You don't need to manually create a service-linked role. When you create an organization trail or organization event data store, or add a delegated administrator in the CloudTrail console, or by using the AWS CLI or API operation, CloudTrail creates the service-linked role for you if it does not already exist.

If you delete this service-linked role, and then need to create it again, you can use the same process to re-create the role in your account. When you create an organization trail or organization event data store, or add a delegated administrator, CloudTrail creates the service-linked role for you again.

Editing a service-linked role for CloudTrail

CloudTrail does not allow you to edit the AWSServiceRoleForCloudTrail service-linked role. After you create a service-linked role, you cannot change the name of the role because various entities might reference the role. However, you can edit the description of the role using IAM. For more information, see Editing a Service-Linked Role in the *IAM User Guide*.

Deleting a service-linked role for CloudTrail

You don't need to manually delete the AWSServiceRoleForCloudTrail role. If an AWS account is removed from an Organizations organization, the AWSServiceRoleForCloudTrail role is automatically removed from that AWS account. You cannot detach or remove policies from the AWSServiceRoleForCloudTrail service-linked role in an organization management account without removing the account from the organization.

You can also use the IAM console, the AWS CLI or the AWS API to manually delete the service-linked role. To do this, you must first manually clean up the resources for your service-linked role, and then you can manually delete it.

> **Note**
>
> If the CloudTrail service is using the role when you try to delete the resources, then deletion might fail. If that happens, wait for a few minutes and try the operation again.

To remove a resource being used by the AWSServiceRoleForCloudTrail role, you can do one of the following:

- Remove the AWS account from the organization in Organizations.
- Update the trail so that it is no longer an organization trail. For more information, see Updating a trail with the CloudTrail console.
- Update the event data store so that it is no longer an organization event data store. For more information, see Update an event data store with the console.
- Delete the trail. For more information, see Deleting a trail with the CloudTrail console.
- Delete the event data store. For more information, see Delete an event data store with the console.

To manually delete the service-linked role using IAM

Use the IAM console, the AWS CLI, or the AWS API to delete the AWSServiceRoleForCloudTrail service-linked role. For more information, see Deleting a service-linked role in the *IAM User Guide*.

Supported Regions for CloudTrail service-linked roles

CloudTrail supports using service-linked roles in all of the AWS Regions where CloudTrail and Organizations are both available. For more information, see AWS service endpoints in the *AWS General Reference*.

AWS managed policies for AWS CloudTrail

To add permissions to users, groups, and roles, it is easier to use AWS managed policies than to write policies yourself. It takes time and expertise to create IAM customer managed policies that provide your team with only the permissions they need. To get started quickly, you can use AWS managed policies. These policies cover common use cases and are available in your AWS account. For more information about AWS managed policies, see AWS managed policies in the *IAM User Guide*.

AWS services maintain and update AWS managed policies. You can't change the permissions in AWS managed policies. Services occasionally add additional permissions to an AWS managed policy to support new features. This type of update affects all identities (users, groups, and roles) where the policy is attached. Services are most likely to update an AWS managed policy when a new feature is launched or when new operations become available. Services do not remove permissions from an AWS managed policy, so policy updates won't break your existing permissions.

Additionally, AWS supports managed policies for job functions that span multiple services. For example, the **ReadOnlyAccess** AWS managed policy provides read-only access to all AWS services and resources. When a service launches a new feature, AWS adds read-only permissions for new operations and resources. For a list and descriptions of job function policies, see AWS managed policies for job functions in the *IAM User Guide*.

AWS managed policy: `AWSCloudTrail_ReadOnlyAccess`

A user identity that has the AWSCloudTrail_ReadOnlyAccess policy attached to its role can perform read-only actions in CloudTrail, such as `Get*`, `List*`, and `Describe*` actions on trails, CloudTrail Lake event data stores, or Lake queries.

AWS managed policy: `AWSServiceRoleForCloudTrail`

The CloudTrailServiceRolePolicy policy allows AWS CloudTrail to perform actions on organization trails and organization event data stores on your behalf. The policy includes required AWS Organizations permissions for describing and listing the organization accounts and delegated administrators in an AWS Organizations organization.

This policy additionally includes the required AWS Glue and AWS Lake Formation permissions to disable Lake federation on an organization event data store.

This policy is attached to the **AWSServiceRoleForCloudTrail** service-linked role that allows CloudTrail to perform actions on your behalf. You cannot attach this policy to your users, groups, or roles.

CloudTrail updates to AWS managed policies

View details about updates to AWS managed policies for CloudTrail. For automatic alerts about changes to this page, subscribe to the RSS feed on the CloudTrail Document history page.

Change	Description	Date
`CloudTrailServiceRolePolicy` – Update to an existing policy	Updated policy to allow the following actions on an organization event data store when federation is disabled: • `glue:DeleteTable` • `lakeformation:DeregisterResource`	November 26, 2023
`AWSCloudTrail_ReadOnlyAccess` – Update to an existing policy	CloudTrail changed the name of the `AWSCloudTrailReadOnlyAccess` policy to `AWSCloudTrail_ReadOnlyAccess`. Also, the scope of permissions in the policy has been reduced to CloudTrail actions. It no longer includes Amazon	June 6, 2022

Change	Description	Date
	S3, AWS KMS, or AWS Lambda action permissions.	
CloudTrail started tracking changes	CloudTrail started tracking changes for its AWS managed policies.	June 6, 2022

Compliance validation for AWS CloudTrail

Third-party auditors assess the security and compliance of AWS CloudTrail as part of multiple AWS compliance programs. These include SOC, PCI, FedRAMP, HIPAA, and others.

To learn whether an AWS service is within the scope of specific compliance programs, see AWS services in Scope by Compliance Program and choose the compliance program that you are interested in. For general information, see AWS Compliance Programs.

You can download third-party audit reports using AWS Artifact. For more information, see Downloading Reports in AWS Artifact.

Your compliance responsibility when using AWS services is determined by the sensitivity of your data, your company's compliance objectives, and applicable laws and regulations. AWS provides the following resources to help with compliance:

- Security and Compliance Quick Start Guides – These deployment guides discuss architectural considerations and provide steps for deploying baseline environments on AWS that are security and compliance focused.
- Architecting for HIPAA Security and Compliance on Amazon Web Services – This whitepaper describes how companies can use AWS to create HIPAA-eligible applications.

  > **Note**
  >
  > Not all AWS services are HIPAA eligible. For more information, see the HIPAA Eligible Services Reference.

- AWS Compliance Resources – This collection of workbooks and guides might apply to your industry and location.

- AWS Customer Compliance Guides – Understand the shared responsibility model through the lens of compliance. The guides summarize the best practices for securing AWS services and map the guidance to security controls across multiple frameworks (including National Institute of Standards and Technology (NIST), Payment Card Industry Security Standards Council (PCI), and International Organization for Standardization (ISO)).
- Evaluating Resources with Rules in the *AWS Config Developer Guide* – The AWS Config service assesses how well your resource configurations comply with internal practices, industry guidelines, and regulations.
- AWS Security Hub – This AWS service provides a comprehensive view of your security state within AWS. Security Hub uses security controls to evaluate your AWS resources and to check your compliance against security industry standards and best practices. For a list of supported services and controls, see Security Hub controls reference.
- Amazon GuardDuty – This AWS service detects potential threats to your AWS accounts, workloads, containers, and data by monitoring your environment for suspicious and malicious activities. GuardDuty can help you address various compliance requirements, like PCI DSS, by meeting intrusion detection requirements mandated by certain compliance frameworks.
- AWS Audit Manager – This AWS service helps you continuously audit your AWS usage to simplify how you manage risk and compliance with regulations and industry standards.

Resilience in AWS CloudTrail

The AWS global infrastructure is built around AWS Regions and Availability Zones. AWS Regions provide multiple physically separated and isolated Availability Zones, which are connected with low-latency, high-throughput, and highly redundant networking. With Availability Zones, you can design and operate applications and databases that automatically fail over between Availability Zones without interruption. Availability Zones are more highly available, fault tolerant, and scalable than traditional single or multiple data center infrastructures. If you specifically need to replicate your CloudTrail log files over greater geographic distances, you can use Cross-Region Replication for your trail Amazon S3 buckets, which enables automatic, asynchronous copying of objects across buckets in different AWS Regions.

For more information about AWS Regions and Availability Zones, see AWS Global Infrastructure.

In addition to the AWS global infrastructure, CloudTrail offers several features to help support your data resiliency and backup needs.

Trails and event data stores that log events in all AWS Regions

When you apply a trail to all AWS Regions, CloudTrail creates trails with identical configurations in all other AWS Regions in the AWS partition in which you are working. When AWS adds a new Region, that trail configuration is automatically created in the new Region.

When you create a multi-Region event data store, CloudTrail collects events that occur in all AWS Regions in your account.

Versioning, lifecycle configuration, and object lock protection for CloudTrail log data

Because CloudTrail uses Amazon S3 buckets to store log files, you can also use the features provided by Amazon S3 to help support your data resiliency and backup needs. For more information, see Resilience in Amazon S3.

Infrastructure security in AWS CloudTrail

As a managed service, AWS CloudTrail is protected by AWS global network security. For information about AWS security services and how AWS protects infrastructure, see AWS Cloud Security. To design your AWS environment using the best practices for infrastructure security, see Infrastructure Protection in *Security Pillar AWS Well‐Architected Framework*.

You use AWS published API calls to access CloudTrail through the network. Clients must support the following:

- Transport Layer Security (TLS). We require TLS 1.2 and recommend TLS 1.3.
- Cipher suites with perfect forward secrecy (PFS) such as DHE (Ephemeral Diffie-Hellman) or ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Most modern systems such as Java 7 and later support these modes.

Additionally, requests must be signed by using an access key ID and a secret access key that is associated with an IAM principal. Or you can use the AWS Security Token Service (AWS STS) to generate temporary security credentials to sign requests.

The following security best practices also address infrastructure security in CloudTrail:

- Consider Amazon VPC endpoints for trail access.
- Consider Amazon VPC endpoints for Amazon S3 bucket access. For more information, see Controlling access from VPC endpoints with bucket policies.
- Identify and audit all Amazon S3 buckets that contain CloudTrail log files. Consider using tags to help identify both your CloudTrail trails and the Amazon S3 buckets that contain CloudTrail log

files. You can then use resource groups for your CloudTrail resources. For more information, see AWS Resource Groups.

Cross-service confused deputy prevention

The confused deputy problem is a security issue where an entity that doesn't have permission to perform an action can coerce a more-privileged entity to perform the action. In AWS, cross-service impersonation can result in the confused deputy problem. Cross-service impersonation can occur when one service (the *calling service*) calls another service (the *called service*). The calling service can be manipulated to use its permissions to act on another customer's resources in a way it should not otherwise have permission to access. To prevent this, AWS provides tools that help you protect your data for all services with service principals that have been given access to resources in your account.

We recommend using the `aws:SourceArn` and `aws:SourceAccount` global condition context keys in resource policies to limit the permissions that AWS CloudTrail gives another service to the resource. Use `aws:SourceArn` if you want only one resource to be associated with the cross-service access. Use `aws:SourceAccount` if you want to allow any resource in that account to be associated with the cross-service use.

The most effective way to protect against the confused deputy problem is to use the `aws:SourceArn` global condition context key with the full ARN of the resource. If you don't know the full ARN of the resource or if you are specifying multiple resources, use the `aws:SourceArn` global context condition key with wildcards (*) for the unknown portions of the ARN. For example, `"arn:aws:cloudtrail:*:`*`AccountID`*`:trail/*"`. When you include a wildcard, you must also use the `StringLike` condition operator.

The value of `aws:SourceArn` must be the ARN of the trail, event data store, or channel that is using the resource.

The following example shows how you can use the `aws:SourceArn` and `aws:SourceAccount` global condition context keys in CloudTrail to prevent the confused deputy problem: Amazon S3 bucket policy for CloudTrail Lake query results.

Security best practices in AWS CloudTrail

AWS CloudTrail provides a number of security features to consider as you develop and implement your own security policies. The following best practices are general guidelines and don’t represent a complete security solution. Because these best practices might not be appropriate or sufficient for your environment, treat them as helpful considerations rather than prescriptions.

Topics

- CloudTrail detective security best practices
- CloudTrail preventative security best practices

CloudTrail detective security best practices

Create a trail

For an ongoing record of events in your AWS account, you must create a trail. Although CloudTrail provides 90 days of event history information for management events in the CloudTrail console without creating a trail, it is not a permanent record, and it does not provide information about all possible types of events. For an ongoing record, and for a record that contains all the event types you specify, you must create a trail, which delivers log files to an Amazon S3 bucket that you specify.

To help manage your CloudTrail data, consider creating one trail that logs management events in all AWS Regions, and then creating additional trails that log specific event types for resources, such as Amazon S3 bucket activity or AWS Lambda functions.

The following are some steps you can take:

- Create a trail for your AWS account.
- Create a trail for an organization.

Apply trails to all AWS Regions

To obtain a complete record of events taken by an IAM identity, or service in your AWS account, each trail should be configured to log events in all AWS Regions. By logging events in all AWS Regions, you ensure that all events that occur in your AWS account are logged, regardless of which AWS Region where they occurred. This includes logging global service events, which are logged to an AWS Region specific to that service. When you create a trail that applies to all Regions, CloudTrail records events in each Region and delivers the CloudTrail event log files to an S3 bucket that you specify. If an AWS Region is added after you create a trail that applies to all Regions, that new Region is automatically included, and events in that Region are logged. This is the default option when you create a trail in the CloudTrail console.

The following are some steps you can take:

- Create a trail for your AWS account.
- Update an existing trail to log events in all AWS Regions.
- Implement ongoing detective controls to help ensure all trails created are logging events in all AWS Regions by using the multi-region-cloud-trail-enabled rule in AWS Config.

Enable CloudTrail log file integrity

Validated log files are especially valuable in security and forensic investigations. For example, a validated log file enables you to assert positively that the log file itself has not changed, or that particular IAM identity credentials performed specific API activity. The CloudTrail log file integrity validation process also lets you know if a log file has been deleted or changed, or assert positively that no log files were delivered to your account during a given period of time. CloudTrail log file integrity validation uses industry standard algorithms: SHA-256 for hashing and SHA-256 with RSA for digital signing. This makes it computationally unfeasible to modify, delete or forge CloudTrail log files without detection. For more information, see Enabling validation and validating files.

Integrate with Amazon CloudWatch Logs

CloudWatch Logs allows you to monitor and receive alerts for specific events captured by CloudTrail. The events sent to CloudWatch Logs are those configured to be logged by your trail, so make sure you have configured your trail or trails to log the event types (management events and/or data events) that you are interested in monitoring.

For example, you can monitor key security and network-related management events, such as failed AWS Management Console sign-in events.

The following are some steps you can take:

- Review example CloudWatch Logs integrations for CloudTrail.
- Configure your trail to send events to CloudWatch Logs.
- Consider implementing ongoing detective controls to help ensure all trails are sending events to CloudWatch Logs for monitoring by using the cloud-trail-cloud-watch-logs-enabled rule in AWS Config.

Use Amazon GuardDuty

Amazon GuardDuty is a threat detection service that helps you protect your accounts, containers, workloads, and the data within your AWS environment. By using machine learning (ML) models, and anomaly and threat detection capabilities, GuardDuty continuously monitors different log sources to identify, and prioritize potential security risks and malicious activities in your environment.

For example, GuardDuty will detect potential credential exfiltration in case it detects credentials that were created exclusively for an Amazon EC2 instance through an instance launch role but are being used from another account within AWS. For more information, see the *Amazon GuardDuty User Guide*.

Use AWS Security Hub

Monitor your usage of CloudTrail as it relates to security best practices by using AWS Security Hub. Security Hub uses detective *security controls* to evaluate resource configurations and *security standards* to help you comply with various compliance frameworks. For more information about using Security Hub to evaluate CloudTrail resources, see AWS CloudTrail controls in the *AWS Security Hub User Guide*.

CloudTrail preventative security best practices

The following best practices for CloudTrail can help prevent security incidents.

Log to a dedicated and centralized Amazon S3 bucket

CloudTrail log files are an audit log of actions taken by an IAM identity or an AWS service. The integrity, completeness and availability of these logs is crucial for forensic and auditing purposes. By logging to a dedicated and centralized Amazon S3 bucket, you can enforce strict security controls, access, and segregation of duties.

The following are some steps you can take:

- Create a separate AWS account as a log archive account. If you use AWS Organizations, enroll this account in the organization, and consider creating an organization trail to log data for all AWS accounts in your organization.
- If you do not use Organizations but want to log data for multiple AWS accounts, create a trail to log activity in this log archive account. Restrict access to this account to only trusted administrative users who should have access to account and auditing data.
- As part of creating a trail, whether it is an organization trail or a trail for a single AWS account, create a dedicated Amazon S3 bucket to store log files for this trail.

- If you want to log activity for more than one AWS account, modify the bucket policy to allow logging and storing log files for all AWS accounts that you want to log AWS account activity.
- If you are not using an organization trail, create trails in all of your AWS accounts, specifying the Amazon S3 bucket in the log archive account.

Use server-side encryption with AWS KMS managed keys

By default, the log files delivered by CloudTrail to your S3 bucket are encrypted by using server-side encryption with a KMS key (SSE-KMS). To use SSE-KMS with CloudTrail, you create and manage an AWS KMS key, also known as a KMS key.

> **Note**
>
> If you use SSE-KMS and log file validation, and you have modified your Amazon S3 bucket policy to only allow SSE-KMS encrypted files, you will not be able to create trails that utilize that bucket unless you modify your bucket policy to specifically allow AES256 encryption, as shown in the following example policy line.
>
> ```
> "StringNotEquals": { "s3:x-amz-server-side-encryption": ["aws:kms", "AES256"] }
> ```

The following are some steps you can take:

- Review the advantages of encrypting your log files with SSE-KMS.
- Create a KMS key to use for encrypting log files.
- Configure log file encryption for your trails.
- Consider implementing ongoing detective controls to help ensure all trails are encrypting log files with SSE-KMS by using the cloud-trail-encryption-enabled rule in AWS Config.

Add a condition key to the default Amazon SNS topic policy

When you configure a trail to send notifications to Amazon SNS, CloudTrail adds a policy statement to your SNS topic access policy that allows CloudTrail to send content to an SNS topic. As a security best practice, we recommend adding an `aws:SourceArn` (or optionally `aws:SourceAccount`) condition key to the Amazon SNS topic policy statement. This helps prevent unauthorized account access to your SNS topic. For more information, see Amazon SNS topic policy for CloudTrail.

Implement least privilege access to Amazon S3 buckets where you store log files

CloudTrail trails log events to an Amazon S3 bucket that you specify. These log files contain an audit log of actions taken by IAM identities and AWS services. The integrity and completeness of these log files are crucial for auditing and forensic purposes. In order to help ensure that integrity, you should adhere to the principle of least privilege when creating or modifying access to any Amazon S3 bucket used for storing CloudTrail log files.

Take the following steps:

- Review the Amazon S3 bucket policy for any and all buckets where you store log files and adjust it if necessary to remove any unnecessary access. This bucket policy will be generated for you if you create a trail using the CloudTrail console, but can also be created and managed manually.
- As a security best practice, be sure to manually add a `aws:SourceArn` condition key to the bucket policy. For more information, see Amazon S3 bucket policy for CloudTrail.
- If you are using the same Amazon S3 bucket to store log files for multiple AWS accounts, follow the guidance for receiving log files for multiple accounts.
- If you are using an organization trail, make sure you follow the guidance for organization trails, and review the example policy for an Amazon S3 bucket for an organization trail in Creating a trail for an organization with the AWS CLI.
- Review the Amazon S3 security documentation and the example walkthrough for securing a bucket.

Enable MFA delete on the Amazon S3 bucket where you store log files

When you configure multi-factor authentication (MFA), attempts to change the versioning state of bucket, or delete an object version in a bucket, require additional authentication. This way, even if a user acquires the password of an IAM user with permissions to permanently delete Amazon S3 objects, you can still prevent operations that could compromise your log files.

The following are some steps you can take:

- Review the MFA delete guidance in the *Amazon Simple Storage Service User Guide*.
- Add an Amazon S3 bucket policy to require MFA.

Note

You cannot use MFA delete with lifecycle configurations. For more information about lifecycle configurations and how they interact with other configurations, see Lifecycle and other bucket configurations in the *Amazon Simple Storage Service User Guide*.

Configure object lifecycle management on the Amazon S3 bucket where you store log files

The CloudTrail trail default is to store log files indefinitely in the Amazon S3 bucket configured for the trail. You can use the Amazon S3 object lifecycle management rules to define your own retention policy to better meet your business and auditing needs. For example, you might want to archive log files that are more than a year old to Amazon Glacier, or delete log files after a certain amount of time has passed.

Note

Lifecycle configuration on multi-factor authentication (MFA)-enabled buckets is not supported.

Limit access to the AWSCloudTrail_FullAccess policy

Users with the AWSCloudTrail_FullAccess policy have the ability to disable or reconfigure the most sensitive and important auditing functions in their AWS accounts. This policy is not intended to be shared or applied broadly to IAM identities in your AWS account. Limit application of this policy to as few individuals as possible, those you expect to act as AWS account administrators.

Encrypting CloudTrail log files with AWS KMS keys (SSE-KMS)

By default, the log files delivered by CloudTrail to your bucket are encrypted by using server-side encryption with a KMS key (SSE-KMS). If you don't enable SSE-KMS encryption, your logs are encrypted using SSE-S3 encryption.

Note

Enabling server-side encryption encrypts the log files but not the digest files with SSE-KMS. Digest files are encrypted with Amazon S3-managed encryption keys (SSE-S3).

If you are using an existing S3 bucket with an S3 bucket Key, CloudTrail must be allowed permission in the key policy to use the AWS KMS actions `GenerateDataKey` and `DescribeKey`. If `cloudtrail.amazonaws.com` is not granted those permissions in the key policy, you cannot create or update a trail.

To use SSE-KMS with CloudTrail, you create and manage a KMS key, also known as an AWS KMS key. You attach a policy to the key that determines which users can use the key for encrypting and decrypting CloudTrail log files. The decryption is seamless through S3. When authorized users of the key read CloudTrail log files, S3 manages the decryption, and the authorized users are able to read log files in unencrypted form.

This approach has the following advantages:

- You can create and manage the KMS key encryption keys yourself.
- You can use a single KMS key to encrypt and decrypt log files for multiple accounts across all Regions.
- You have control over who can use your key for encrypting and decrypting CloudTrail log files. You can assign permissions for the key to the users in your organization according to your requirements.
- You have enhanced security. With this feature, to read log files, the following permissions are required:
  - A user must have S3 read permissions for the bucket that contains the log files.
  - A user must also have a policy or role applied that allows decrypt permissions by the KMS key policy.
- Because S3 automatically decrypts the log files for requests from users authorized to use the KMS key, SSE-KMS encryption for CloudTrail log files is backward-compatible with applications that read CloudTrail log data.

Note

The KMS key that you choose must be created in the same AWS Region as the Amazon S3 bucket that receives your log files. For example, if the log files will be stored in a bucket in the US East (Ohio) Region, you must create or choose a KMS key that was created in that

Region. To verify the Region for an Amazon S3 bucket, inspect its properties in the Amazon S3 console.

Enabling log file encryption

Note

If you create a KMS key in the CloudTrail console, CloudTrail adds the required KMS key policy sections for you. Follow these procedures if you created a key in the IAM console or AWS CLI and you need to manually add the required policy sections.

To enable SSE-KMS encryption for CloudTrail log files, perform the following high-level steps:

1. Create a KMS key.

   - For information about creating a KMS key with the AWS Management Console, see Creating Keys in the *AWS Key Management Service Developer Guide*.
   - For information about creating a KMS key with the AWS CLI, see create-key.

   Note

   The KMS key that you choose must be in the same Region as the S3 bucket that receives your log files. To verify the Region for an S3 bucket, inspect the bucket's properties in the S3 console.

2. Add policy sections to the key that enable CloudTrail to encrypt and users to decrypt log files.

   - For information about what to include in the policy, see Configure AWS KMS key policies for CloudTrail.

     Warning

     Be sure to include decrypt permissions in the policy for all users who need to read log files. If you do not perform this step before adding the key to your trail configuration, users without decrypt permissions cannot read encrypted files until you grant them those permissions.

- For information about editing a policy with the IAM console, see Editing a Key Policy in the *AWS Key Management Service Developer Guide*.
- For information about attaching a policy to a KMS key with the AWS CLI, see put-key-policy.

3. Update your trail to use the KMS key whose policy you modified for CloudTrail.

   - To update your trail configuration by using the CloudTrail console, see Updating a resource to use your KMS key with the console.
   - To update your trail configuration by using the AWS CLI, see Enabling and disabling CloudTrail log file encryption with the AWS CLI.

CloudTrail also supports AWS KMS multi-Region keys. For more information about multi-Region keys, see Using multi-Region keys in the *AWS Key Management Service Developer Guide*.

The next section describes the policy sections that your KMS key policy requires for use with CloudTrail.

Granting permissions to create a KMS key

You can grant users permission to create an AWS KMS key with the `AWSKeyManagementServicePowerUser` policy.

To grant permission to create a KMS key

1. Open the IAM console at https://console.aws.amazon.com/iam/.
2. Choose the group or user that you want to give permission.
3. Choose **Permissions**, and then choose **Attach Policy**.
4. Search for **AWSKeyManagementServicePowerUser**, choose the policy, and then choose **Attach policy**.

   The user now has permission to create a KMS key. For more information about creating policies, see Creating IAM policies in the *IAM User Guide*.

Configure AWS KMS key policies for CloudTrail

You can create an AWS KMS key in three ways:

- The CloudTrail console
- The AWS Management console
- The AWS CLI

> **Note**
>
> If you create a KMS key in the CloudTrail console, CloudTrail adds the required KMS key policy for you. You do not need to manually add the policy statements. See Default KMS key policy created in CloudTrail console.

If you create a KMS key in the AWS Management or the AWS CLI, you must add policy sections to the key so that you can use it with CloudTrail. The policy must allow CloudTrail to use the key to encrypt your log files and event data stores, and allow the users you specify to read log files in unencrypted form.

See the following resources:

- To create a KMS key with the AWS CLI, see create-key.
- To edit a KMS key policy for CloudTrail, see Editing a Key Policy in the *AWS Key Management Service Developer Guide*.
- For technical details on how CloudTrail uses AWS KMS, see How AWS CloudTrail Uses AWS KMS in the *AWS Key Management Service Developer Guide*.

Required KMS key policy sections for use with CloudTrail

If you created a KMS key with the AWS Management console or the AWS CLI, then you must, at minimum, add the following statements to your KMS key policy for it to work with CloudTrail.

Topics

Required KMS key policy elements for trails

1. Enable CloudTrail log encrypt permissions. See Granting encrypt permissions.
2. Enable CloudTrail log decrypt permissions. See Granting decrypt permissions. If you are using an existing S3 bucket with an S3 Bucket Key, `kms:Decrypt` permissions are required to create or update a trail with SSE-KMS encryption enabled.
3. Enable CloudTrail to describe KMS key properties. See Enable CloudTrail to describe KMS key properties.

As a security best practice, add an `aws:SourceArn` condition key to the KMS key policy. The IAM global condition key `aws:SourceArn` helps ensure that CloudTrail uses the KMS key only for a specific trail or trails. The value of `aws:SourceArn` is always the trail ARN (or array of trail ARNs) that is using the KMS key. Be sure to add the `aws:SourceArn` condition key to KMS key policies for existing trails.

The `aws:SourceAccount` condition key is also supported, but not recommended. The value of `aws:SourceAccount` is the account ID of the trail owner, or for organization trails, the management account ID.

> ⚠ **Important**
>
> When you add the new sections to your KMS key policy, do not change any existing sections in the policy.
> If encryption is enabled on a trail, and the KMS key is disabled, or the KMS key policy is not correctly configured for CloudTrail, CloudTrail cannot deliver logs.

Required KMS key policy elements for event data stores

1. Enable CloudTrail log encrypt permissions. See Granting encrypt permissions.
2. Enable CloudTrail log decrypt permissions. See Granting decrypt permissions.
3. Grant users and roles permission to encrypt and decrypt event data store data with the KMS key.

   When you create an event data store and encrypt it with a KMS key, or run queries on an event data store that you're encrypting with a KMS key, you should have write access to the KMS key. The KMS key policy must have access to CloudTrail, and the KMS key should be manageable by users who run operations (such as queries) on the event data store.

4. Enable CloudTrail to describe KMS key properties. See [Enable CloudTrail to describe KMS key properties](#).

The `aws:SourceArn` and `aws:SourceAccount` condition keys are not supported in KMS key policies for event data stores.

> **Important**
>
> When you add the new sections to your KMS key policy, do not change any existing sections in the policy.
> If encryption is enabled on an event data store, and the KMS key is disabled or deleted, or the KMS key policy is not correctly configured for CloudTrail, CloudTrail cannot deliver events to your event data store.

Granting encrypt permissions

Example Allow CloudTrail to encrypt logs on behalf of specific accounts

CloudTrail needs explicit permission to use the KMS key to encrypt logs on behalf of specific accounts. To specify an account, add the following required statement to your KMS key policy and replace *account-id*, *region*, and *trailName* with the appropriate values for your configuration. You can add additional account IDs to the `EncryptionContext` section to enable those accounts to use CloudTrail to use your KMS key to encrypt log files.

As a security best practice, add an `aws:SourceArn` condition key to the KMS key policy for a trail. The IAM global condition key `aws:SourceArn` helps ensure that CloudTrail uses the KMS key only for a specific trail or trails.

```
{
   "Sid": "Allow CloudTrail to encrypt logs",
   "Effect": "Allow",
   "Principal": {
       "Service": "cloudtrail.amazonaws.com"
    },
    "Action": "kms:GenerateDataKey*",
    "Resource": "*",
    "Condition": {
        "StringEquals": {
            "aws:SourceArn": "arn:aws:cloudtrail:region:account-id:trail/trail-name"
         },
         "StringLike": {
             "kms:EncryptionContext:aws:cloudtrail:arn": "arn:aws:cloudtrail:*:account-
id:trail/*"
         }
    }
}
```

A policy for a KMS key used to encrypt CloudTrail Lake event data store logs cannot use the condition keys `aws:SourceArn` or `aws:SourceAccount`. The following is an example of a KMS key policy for an event data store.

```
{
    "Sid": "Allow CloudTrail to encrypt event data store",
    "Effect": "Allow",
    "Principal": {
        "Service": "cloudtrail.amazonaws.com"
     },
     "Action": [
        "kms:GenerateDataKey",
        "kms:Decrypt"
      ],
      "Resource": "*"
}
```

Example

The following example policy statement illustrates how another account can use your KMS key to encrypt CloudTrail logs.

Scenario

- Your KMS key is in account *111111111111*.
- Both you and account *222222222222* will encrypt logs.

In the policy, you add one or more accounts that encrypt with your key to the CloudTrail **EncryptionContext**. This restricts CloudTrail to using your key to encrypt logs only for the accounts that you specify. When you give the root of account *222222222222* permission to encrypt logs, it delegates permission to the account administrator to encrypt the necessary permissions to other users in that account. The account administrator does this by changing the policies associated with those IAM users.

As a security best practice, add an `aws:SourceArn` condition key to the KMS key policy. The IAM global condition key `aws:SourceArn` helps ensure that CloudTrail uses the KMS key only for the specified trails. This condition isn't supported in KMS key policies for event data stores.

KMS key policy statement:

```
{
  "Sid": "Enable CloudTrail encrypt permissions",
  "Effect": "Allow",
  "Principal": {
    "Service": "cloudtrail.amazonaws.com"
  },
  "Action": "kms:GenerateDataKey*",
  "Resource": "*",
  "Condition": {
    "StringLike": {
      "kms:EncryptionContext:aws:cloudtrail:arn": [
        "arn:aws:cloudtrail:*:111111111111:trail/*",
        "arn:aws:cloudtrail:*:222222222222:trail/*"
      ]
    },
    "StringEquals": {
        "aws:SourceArn": "arn:aws:cloudtrail:region:account-id:trail/trail-name"
    }
  }
}
```

For more information about editing a KMS key policy for use with CloudTrail, see Editing a key policy in the AWS Key Management Service Developer Guide.

Granting decrypt permissions

Before you add your KMS key to your CloudTrail configuration, it is important to give decrypt permissions to all users who require them. Users who have encrypt permissions but no decrypt permissions cannot read encrypted logs. If you are using an existing S3 bucket with an S3 Bucket Key, `kms:Decrypt` permissions are required to create or update a trail with SSE-KMS encryption enabled.

Enable CloudTrail log decrypt permissions

Users of your key must be given explicit permissions to read the log files that CloudTrail has encrypted. To enable users to read encrypted logs, add the following required statement to your KMS key policy, modifying the `Principal` section to add a line for every principal that you want to be able decrypt by using your KMS key.

```
{
  "Sid": "Enable CloudTrail log decrypt permissions",
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::account-id:user/username"
  },
  "Action": "kms:Decrypt",
  "Resource": "*",
  "Condition": {
    "Null": {
      "kms:EncryptionContext:aws:cloudtrail:arn": "false"
    }
  }
}
```

The following is an example policy that is required to allow the CloudTrail service principal to decrypt trail logs.

```
{
      "Sid": "Allow CloudTrail to decrypt a trail",
      "Effect": "Allow",
      "Principal": {
          "Service": "cloudtrail.amazonaws.com"
        },
      "Action": "kms:Decrypt",
      "Resource": "*"
}
```

A decrypt policy for a KMS key that is used with a CloudTrail Lake event data store is similar to the following. The user or role ARNs specified as values for `Principal` need decrypt permissions to create or update event data stores, run queries, or get query results.

```
{
      "Sid": "Enable user key permissions for event data stores"
      "Effect": "Allow",
      "Principal": {
          "AWS": "arn:aws:iam::account-id:user/username"
      },
      "Action": [
          "kms:Decrypt",
          "kms:GenerateDataKey"
      ],
      "Resource": "*"
  }
```

The following is an example policy that is required to allow the CloudTrail service principal to decrypt event data store logs.

```
{
      "Sid": "Allow CloudTrail to decrypt an event data store",
      "Effect": "Allow",
      "Principal": {
          "Service": "cloudtrail.amazonaws.com"
        },
      "Action": "kms:Decrypt",
      "Resource": "*"
}
```

Allow users in your account to decrypt trail logs with your KMS key

Example

This policy statement illustrates how to allow a user or role in your account to use your key to read the encrypted logs in your account's S3 bucket.

Example Scenario

- Your KMS key, S3 bucket, and IAM user Bob are in account *111111111111*.
- You give IAM user Bob permission to decrypt CloudTrail logs in the S3 bucket.

In the key policy, you enable CloudTrail log decrypt permissions for IAM user Bob.

KMS key policy statement:

```
{
  "Sid": "Enable CloudTrail log decrypt permissions",
  "Effect": "Allow",
  "Principal": {
    "AWS": "arn:aws:iam::111111111111:user/Bob"
  },
  "Action": "kms:Decrypt",
  "Resource": "arn:aws:kms:region:account-id:key/key-id",
  "Condition": {
    "Null": {
      "kms:EncryptionContext:aws:cloudtrail:arn": "false"
    }
  }
}
```

Allow users in other accounts to decrypt trail logs with your KMS key

You can allow users in other accounts to use your KMS key to decrypt trail logs, but not event data store logs. The changes required to your key policy depend on whether the S3 bucket is in your account or in another account.

Allow users of a bucket in a different account to decrypt logs

Example

This policy statement illustrates how to allow an IAM user or role in another account to use your key to read encrypted logs from an S3 bucket in the other account.

Scenario

- Your KMS key is in account *111111111111*.
- The IAM user Alice and S3 bucket are in account *222222222222*.

In this case, you give CloudTrail permission to decrypt logs under account *222222222222*, and you give Alice's IAM user policy permission to use your key *KeyA*, which is in account *111111111111*.

KMS key policy statement:

```
{
  "Sid": "Enable encrypted CloudTrail log read access",
  "Effect": "Allow",
  "Principal": {
    "AWS": [
      "arn:aws:iam::222222222222:root"
    ]
  },
  "Action": "kms:Decrypt",
  "Resource": "arn:aws:kms:region:account-id:key/key-id",
  "Condition": {
    "Null": {
      "kms:EncryptionContext:aws:cloudtrail:arn": "false"
    }
  }
}
```

Alice's IAM user policy statement:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "kms:Decrypt",
      "Resource": "arn:aws:kms:us-west-2:111111111111:key/KeyA"
    }
  ]
}
```

Allow users in a different account to decrypt trail logs from your bucket

Example

This policy illustrates how another account can use your key to read encrypted logs from your S3 bucket.

Example Scenario

- Your KMS key and S3 bucket are in account *111111111111*.
- The user who reads logs from your bucket is in account *222222222222*.

To enable this scenario, you enable decrypt permissions for the IAM role **CloudTrailReadRole** in your account, and then give the other account permission to assume that role.

KMS key policy statement:

```
{
  "Sid": "Enable encrypted CloudTrail log read access",
  "Effect": "Allow",
  "Principal": {
    "AWS": [
      "arn:aws:iam::111111111111:role/CloudTrailReadRole"
    ]
  },
  "Action": "kms:Decrypt",
  "Resource": "arn:aws:kms:region:account-id:key/key-id",
  "Condition": {
    "Null": {
      "kms:EncryptionContext:aws:cloudtrail:arn": "false"
    }
  }
}
```

CloudTrailReadRole trust entity policy statement:

```
{
 "Version": "2012-10-17",
 "Statement": [
   {
     "Sid": "Allow CloudTrail access",
     "Effect": "Allow",
     "Principal": {
       "AWS": "arn:aws:iam::222222222222:root"
     },
     "Action": "sts:AssumeRole"
    }
  ]
 }
```

For information about editing a KMS key policy for use with CloudTrail, see Editing a Key Policy in the *AWS Key Management Service Developer Guide*.

Enable CloudTrail to describe KMS key properties

CloudTrail requires the ability to describe the properties of the KMS key. To enable this functionality, add the following required statement as is to your KMS key policy. This statement does not grant CloudTrail any permissions beyond the other permissions that you specify.

As a security best practice, add an `aws:SourceArn` condition key to the KMS key policy. The IAM global condition key `aws:SourceArn` helps ensure that CloudTrail uses the KMS key only for a specific trail or trails.

```
{
  "Sid": "Allow CloudTrail access",
  "Effect": "Allow",
  "Principal": {
    "Service": "cloudtrail.amazonaws.com"
  },
  "Action": "kms:DescribeKey",
  "Resource": "arn:aws:kms:region:account-id:key/key-id",
  "Condition": {
    "StringEquals": {
        "aws:SourceArn": "arn:aws:cloudtrail:region:account-id:trail/trail-name"
    }
  }
}
```

For more information about editing KMS key policies, see Editing a Key Policy in the *AWS Key Management Service Developer Guide*.

Default KMS key policy created in CloudTrail console

If you create an AWS KMS key in the CloudTrail console, the following policies are automatically created for you. The policy allows these permissions:

- Allows AWS account (root) permissions for the KMS key.
- Allows CloudTrail to encrypt log files under the KMS key and describe the KMS key.
- Allows all users in the specified accounts to decrypt log files.
- Allows all users in the specified account to create a KMS alias for the KMS key.
- Enables cross-account log decryption for the account ID of the account that created the trail.

Topics

Default KMS key policy for CloudTrail Lake event data stores

The following is the default policy created for a AWS KMS key that you use with an event data store in CloudTrail Lake.

```
{
      "Version": "2012-10-17",
      "Id": "Key policy created by CloudTrail",
      "Statement": [
        {
          "Sid": "The key created by CloudTrail to encrypt event data stores. Created
 ${new Date().toUTCString()}",
          "Effect": "Allow",
          "Principal": {
            "Service": "cloudtrail.amazonaws.com"
          },
          "Action": [
            "kms:GenerateDataKey",
            "kms:Decrypt"
          ],
          "Resource": "*"
        },
        {
          "Sid": "Enable IAM user permissions",
          "Effect": "Allow",
          "Principal": {
            "AWS": "arn:aws:iam::account-id:root"
          },
          "Action": "kms:*",
          "Resource": "*"
        },
        {
          "Sid": "Enable user to have permissions",
          "Effect": "Allow",
          "Principal": {
            "AWS" : "arn:aws:sts::account-id:role-arn"
          },
          "Action": [
            "kms:Decrypt",
            "kms:GenerateDataKey"
           ],
          "Resource": "*"
        }
      ]
    }
```

Default KMS key policy for trails

The following is the default policy created for a AWS KMS key that you use with a trail.

> **Note**
>
> The policy includes a statement to allow cross accounts to decrypt log files with the KMS key.

```
{
    "Version": "2012-10-17",
    "Id": "Key policy created by CloudTrail",
    "Statement": [
        {
            "Sid": "Enable IAM user permissions",
            "Effect": "Allow",
            "Principal": {
                "AWS": [
                    "arn:aws:iam::account-id:root",
                    "arn:aws:iam::account-id:user/username"
                ]
            },
            "Action": "kms:*",
            "Resource": "*"
        },
        {
            "Sid": "Allow CloudTrail to encrypt logs",
            "Effect": "Allow",
            "Principal": {
                "Service": "cloudtrail.amazonaws.com"
             },
            "Action": "kms:GenerateDataKey*",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:SourceArn": "arn:aws:cloudtrail:region:account-id:trail/trail-
name"
                },
                "StringLike": {
                    "kms:EncryptionContext:aws:cloudtrail:arn":
 "arn:aws:cloudtrail:*:account-id:trail/*"
                }
            }
        },
        {
            "Sid": "Allow CloudTrail to describe key",
            "Effect": "Allow",
            "Principal": {
                "Service": "cloudtrail.amazonaws.com"
             },
            "Action": "kms:DescribeKey",
            "Resource": "*"
        },
        {
            "Sid": "Allow principals in the account to decrypt log files",
            "Effect": "Allow",
            "Principal": {
                "AWS": "*"
             },
            "Action": [
                "kms:Decrypt",
                "kms:ReEncryptFrom"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "kms:CallerAccount": "account-id"
                },
                "StringLike": {
                    "kms:EncryptionContext:aws:cloudtrail:arn":
 "arn:aws:cloudtrail:*:account-id:trail/*"
                }
            }
        },
        {
            "Sid": "Allow alias creation during setup",
            "Effect": "Allow",
            "Principal": {
                "AWS": "*"
             },
            "Action": "kms:CreateAlias",
            "Resource": "arn:aws:kms:region:account-id:key/key-id",
            "Condition": {
                "StringEquals": {
                    "kms:ViaService": "ec2.region.amazonaws.com",
                    "kms:CallerAccount": "account-id"
                }
            }
        },
        {
            "Sid": "Enable cross account log decryption",
            "Effect": "Allow",
            "Principal": {
                "AWS": "*"
            },
            "Action": [
                "kms:Decrypt",
                "kms:ReEncryptFrom"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "kms:CallerAccount": "account-id"
                },
                "StringLike": {
                    "kms:EncryptionContext:aws:cloudtrail:arn":
 "arn:aws:cloudtrail:*:account-id:trail/*"
                }
            }
        }
    ]
}
```

Updating a resource to use your KMS key with the console

In the AWS CloudTrail console, update a trail or an event data store to use an AWS Key Management Service key. Be aware that using your own KMS key incurs AWS KMS costs for encryption and decryption. For more information, see AWS Key Management Service Pricing.

Topics

- Update a trail to use a KMS key
- Update an event data store to use a KMS key

Update a trail to use a KMS key

To update a trail to use the AWS KMS key that you modified for CloudTrail, complete the following steps in the CloudTrail console.

> **Note**
>
> Updating a trail with the following procedure encrypts the log files but not the digest files with SSE-KMS. Digest files are encrypted with Amazon S3-managed encryption keys (SSE-S3).
>
> If you are using an existing S3 bucket with an S3 Bucket Key, CloudTrail must be allowed permission in the key policy to use the AWS KMS actions `GenerateDataKey` and `DescribeKey`. If `cloudtrail.amazonaws.com` is not granted those permissions in the key policy, you cannot create or update a trail.

To update a trail using the AWS CLI, see Enabling and disabling CloudTrail log file encryption with the AWS CLI.

To update a trail to use your KMS key

1. Sign in to the AWS Management Console and open the CloudTrail console at https://console.aws.amazon.com/cloudtrail/.
2. Choose **Trails** and then choose a trail name.
3. In **General details**, choose **Edit**.
4. For **Log file SSE-KMS encryption**, choose **Enabled** if you want to encrypt your log files using SSE-KMS encryption instead of SSE-S3 encryption. The default is **Enabled**. If you don't enable SSE-KMS encryption, your logs are encrypted using SSE-S3 encryption. For more information about SSE-KMS encryption, see Using server-side encryption with AWS Key Management Service (SSE-KMS). For more information about SSE-S3 encryption, see Using Server-Side Encryption with Amazon S3-Managed Encryption Keys (SSE-S3).

   Choose **Existing** to update your trail with your AWS KMS key. Choose a KMS key that is in the same Region as the S3 bucket that receives your log files. To verify the Region for an S3 bucket, view its properties in the S3 console.

   > **Note**
   >
   > You can also type the ARN of a key from another account. For more information, see Updating a resource to use your KMS key with the console. The key policy must allow CloudTrail to use the key to encrypt your log files, and allow the users you specify to

read log files in unencrypted form. For information about manually editing the key policy, see Configure AWS KMS key policies for CloudTrail.

In **AWS KMS Alias**, specify the alias for which you changed the policy for use with CloudTrail, in the format `alias/MyAliasName`. For more information, see Updating a resource to use your KMS key with the console.

You can type the alias name, ARN, or the globally unique key ID. If the KMS key belongs to another account, verify that the key policy has permissions that enable you to use it. The value can be one of the following formats:

- **Alias Name**: `alias/MyAliasName`
- **Alias ARN**: `arn:aws:kms:region:123456789012:alias/MyAliasName`
- **Key ARN**: `arn:aws:kms:region:123456789012:key/12345678-1234-1234-1234-123456789012`
- **Globally unique key ID**: `12345678-1234-1234-1234-123456789012`

5. Choose **Update trail**.

> **Note**
>
> If the KMS key that you chose is disabled or is pending deletion, you cannot save the trail with that KMS key. You can enable the KMS key or choose another one. For more information, see Key state: Effect on your KMS key in the *AWS Key Management Service Developer Guide*.

Update an event data store to use a KMS key

To update an event data store to use the AWS KMS key that you modified for CloudTrail, complete the following steps in the CloudTrail console.

To update an event data store by using the AWS CLI, see Update an event data store with the AWS CLI.

> **Important**
>
> Disabling or deleting the KMS key, or removing CloudTrail permissions on the key, prevents CloudTrail from ingesting events into the event data store, and prevents users from querying data in the event data store that was encrypted with the key. After you associate an event data store with a KMS key, the KMS key cannot be removed or changed. Before you disable or delete a KMS key that you are using with an event data store, delete or back up your event data store.

To update an event data store to use your KMS key

1. Sign in to the AWS Management Console and open the CloudTrail console at https://console.aws.amazon.com/cloudtrail/.
2. In the navigation pane, choose **Event data stores** in **Lake**. Choose an event data store to update.
3. In **General details**, choose **Edit**.
4. For **Encryption**, if it is not already enabled, choose **Use my own AWS KMS key** to encrypt your log files with your own KMS key.

   Choose **Existing** to update your event data store with your KMS key. Choose a KMS key that is in the same Region as the event data store. A key from another account is not supported.

   In **Enter AWS KMS Alias**, specify the alias for which you changed the policy for use with CloudTrail, in the format `alias/`*`MyAliasName`*. For more information, see Updating a resource to use your KMS key with the console.

   You can choose an alias, or use the globally unique key ID. The value can be one of the following formats:

   - **Alias Name**: `alias/`*`MyAliasName`*
   - **Alias ARN**: `arn:aws:kms:`*`region`*`:123456789012:alias/`*`MyAliasName`*
   - **Key ARN**: `arn:aws:kms:`*`region`*`:123456789012:key/12345678-1234-1234-1234-123456789012`
   - **Globally unique key ID**: `12345678-1234-1234-1234-123456789012`
5. Choose **Save changes**.

> **Note**
>
> If the KMS key that you chose is disabled or is pending deletion, you cannot save the event data store configuration with that KMS key. You can enable the KMS key, or choose a different key. For more information, see Key state: Effect on your KMS key in the *AWS Key Management Service Developer Guide*.

Enabling and disabling CloudTrail log file encryption with the AWS CLI

This topic describes how to enable and disable SSE-KMS log file encryption for CloudTrail by using the AWS CLI. For background information, see Encrypting CloudTrail log files with AWS KMS keys (SSE-KMS).

Topics

Enabling CloudTrail log file encryption by using the AWS CLI

Enable log file encryption for a trail

1. Create a key with the AWS CLI. The key that you create must be in the same Region as the S3 bucket that receives your CloudTrail log files. For this step, you use the AWS KMS **create-key** command.
2. Get the existing key policy so that you can modify it for use with CloudTrail. You can retrieve the key policy with the AWS KMS **get-key-policy** command.
3. Add required sections to the key policy so that CloudTrail can encrypt and users can decrypt your log files. Be sure that all users who read the log files are granted decrypt permissions. Do not change existing sections of the policy. For information about the policy sections to include, see Configure AWS KMS key policies for CloudTrail.

4. Attach the modified JSON policy file to the key by using the AWS KMS **put-key-policy** command.

5. Run the CloudTrail `create-trail` or `update-trail` command with the `--kms-key-id` parameter. This command enables log encryption.

   ```
   aws cloudtrail update-trail --name Default --kms-key-id alias/MyKmsKey
   ```

   The `--kms-key-id` parameter specifies the key whose policy you modified for CloudTrail. It can be any one of the following formats:

   - **Alias Name**. Example: `alias/MyAliasName`
   - **Alias ARN**. Example: `arn:aws:kms:us-east-2:123456789012:alias/MyAliasName`
   - **Key ARN**. Example: `arn:aws:kms:us-east-2:123456789012:key/12345678-1234-1234-1234-123456789012`
   - **Globally unique key ID.** Example: `12345678-1234-1234-1234-123456789012`

   The following is an example response:

   ```
   {
       "IncludeGlobalServiceEvents": true,
       "Name": "Default",
       "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/Default",
       "LogFileValidationEnabled": false,
       "KmsKeyId": "arn:aws:kms:us-
   east-2:123456789012:key/12345678-1234-1234-1234-123456789012",
       "S3BucketName": "DOC-EXAMPLE-BUCKET"
   }
   ```

   The presence of the `KmsKeyId` element indicates that log file encryption has been enabled. The encrypted log files should appear in your bucket in about 5 minutes.

Enable log file encryption for an event data store

1. Create a key with the AWS CLI. The key that you create must be in the same Region as the event data store. For this step, run the AWS KMS **create-key** command.

2. Get the existing key policy to edit for use with CloudTrail. You can get the key policy by running the AWS KMS **get-key-policy** command.
3. Add required sections to the key policy so that CloudTrail can encrypt and users can decrypt your log files. Be sure that all users who read the log files are granted decrypt permissions. Do not change existing sections of the policy. For information about the policy sections to include, see Configure AWS KMS key policies for CloudTrail.
4. Attach the edited JSON policy file to the key by running the AWS KMS put-key-policy command.
5. Run the CloudTrail `create-event-data-store` or `update-event-data-store` command, and add the `--kms-key-id` parameter. This command enables log encryption.

   ```
   aws cloudtrail update-event-data-store --name my-event-data-store --kms-key-id
    alias/MyKmsKey
   ```

   The `--kms-key-id` parameter specifies the key whose policy you modified for CloudTrail. It can be any one of the following four formats:

   - **Alias Name**. Example: `alias/MyAliasName`
   - **Alias ARN**. Example: `arn:aws:kms:us-east-2:123456789012:alias/MyAliasName`
   - **Key ARN**. Example: `arn:aws:kms:us-east-1:123456789012:key/12345678-1234-1234-1234-123456789012`
   - **Globally unique key ID.** Example: `12345678-1234-1234-1234-123456789012`

   The following is an example response:

   ```
   {
       "Name": "my-event-data-store",
       "ARN": "arn:aws:cloudtrail:us-east-1:12345678910:eventdatastore/
   EXAMPLEf852-4e8f-8bd1-bcf6cEXAMPLE",
       "RetentionPeriod": "90",
       "KmsKeyId": "arn:aws:kms:us-
   east-1:123456789012:key/12345678-1234-1234-1234-123456789012"
       "MultiRegionEnabled": false,
       "OrganizationEnabled": false,
       "TerminationProtectionEnabled": true,
       "AdvancedEventSelectors": [{
           "Name": "Select all external events",
           "FieldSelectors": [{
   ```

```
                "Field": "eventCategory",
                "Equals": [
                    "ActivityAuditLog"
                ]
            }]
        }]
    }
```

The presence of the KmsKeyId element indicates that log file encryption has been enabled. The encrypted log files should appear in your event data store in about 5 minutes.

Disabling CloudTrail log file encryption by using the AWS CLI

To stop encrypting logs on a trail, run update-trail and pass an empty string to the kms-key-id parameter:

```
aws cloudtrail update-trail --name my-test-trail --kms-key-id ""
```

The following is an example response:

```
{
    "IncludeGlobalServiceEvents": true,
    "Name": "Default",
    "TrailARN": "arn:aws:cloudtrail:us-east-2:123456789012:trail/Default",
    "LogFileValidationEnabled": false,
    "S3BucketName": "DOC-EXAMPLE-BUCKET"
}
```

The absence of the KmsKeyId value indicates that log file encryption is no longer enabled.

Important

You cannot stop log file encryption on an event data store.

Document history

The following table describes the important changes to the documentation for AWS CloudTrail. For notification about updates to this documentation, you can subscribe to an RSS feed.

- **API version**: 2013-11-01
- **Latest documentation update**: 2024-07-09

Change	Description	Date
Added functionality	You can now log CloudTrail data events on Amazon Bedrock flow aliases and guardrails, and Amazon S3 object-level API activity on directory buckets by using advanced event selectors. For more information, see Data events.	July 9, 2024
Added functionality	You can now log CloudTrail data events on AWS Payment Cryptography keys and aliases by using advanced event selectors. For more information, see Data events.	July 5, 2024
Added functionality	Introducing a preview feature for CloudTrail Lake queries that uses generative artificial intelligence (generative AI) capabilities to produce a SQL query from an English language prompt. For more information, see Create	June 11, 2024

	CloudTrail Lake queries from English language prompts.	
Added functionality	You can now log CloudTrail data events on Amazon CloudWatch metrics, Amazon Machine Learning ML models, and AWS Private CA by using advanced event selectors. For more information, see Data events.	June 5, 2024
Updated documentation	Added section to describe how to filter data events by using advanced event selectors. For more information, see Filtering data events by using advanced event selectors.	May 29, 2024
Added functionality	You can now log CloudTrail data events on Amazon Kinesis Data Streams streams and stream consumers by using advanced event selectors. For more information, see Data events.	May 21, 2024
Updated documentation	Updated the CloudTrail Lake supported Regions page to add the Asia Pacific (Hyderabad) Region (ap-south-2), the Europe (Zurich) Region (eu-central-2), and the Israel (Tel Aviv) Region (il-central-1).	May 16, 2024

Added functionality	You can now log CloudTrail data events on AWS Step Functions state machines by using advanced event selectors. For more information, see Data events.	May 16, 2024
Updated documentation	Added section about viewing CloudTrail cost and usage using AWS Cost Explorer. For more information, see Viewing your CloudTrail cost and usage with AWS Cost Explorer.	May 14, 2024
Added functionality	You can now log CloudTrail data events on Amazon Q Apps by using advanced event selectors. For more information, see Data events.	May 1, 2024

Updated documentation	General organizational improvements to user guide sections and page titles, which includes the following: Changed title of CloudTrail log event reference page to Understanding CloudTrail events and added descriptions of management events, data events, and Insights events. Changed title of Settings page to Configure CloudTrail settings. Moved Logging data events, Logging management events, and Logging Insights events pages to the Understanding CloudTrail events section. Moved CloudTrail log file examples page to the CloudTrail log files section. Added separate pages to list the AWS CLI commands for CloudTrail Lake event data stores, queries, and integrations.	April 10, 2024
Updated documentation	Updated the CloudTrail Lake supported Regions page to add the Europe (Spain) Region (eu-south-2).	April 10, 2024

Added service support	This release supports AWS Control Catalog. For more information, see AWS service topics for CloudTrail and Logging AWS Control Catalog API calls using AWS CloudTrail.	April 8, 2024
Added service support	This release supports AWS Deadline Cloud. For more information, see AWS service topics for CloudTrail.	April 2, 2024
Added functionality	The AWS CloudTrail event version is now 1.10. For more information, see CloudTrail record contents.	March 26, 2024
Added service support	This release supports AWS Billing Conductor. For more information, see AWS service topics for CloudTrail and Logging AWS Billing Conductor API calls using AWS CloudTrail.	March 12, 2024
Added functionality	You can now log CloudTrail data events on AWS X-Ray traces and AWS Systems Manager managed nodes by using advanced event selectors. For more information, see Data events.	March 7, 2024

Added functionality	You can now log CloudTrail data events on Amazon Simple Workflow Service (Amazon SWF) domains by using advanced event selectors. For more information, see Data events.	February 14, 2024
Added functionality	CloudTrail added the `ListInsightsMetricData` API. The `ListInsightsMetricData` API returns Insights metrics data for trails that have enabled Insights. For more information, see ListInsightsMetricData in the *AWS CloudTrail API Reference*.	February 6, 2024
Added functionality	You can now log CloudTrail data events for AWS IoT, AWS IoT SiteWise, and AWS AppConfig by using advanced event selectors. For more information, see Data events.	January 4, 2024
Added functionality	You can now log CloudTrail data events for AWS IoT Greengrass by using advanced event selectors. For more information, see Data events.	December 22, 2023
New Region support	CloudTrail expanded support to a new Region, the Canada West (Calgary) Region. For more information, see CloudTrail supported Regions.	December 20, 2023

Added functionality	You can now log CloudTrail data events for Amazon Keyspaces (for Apache Cassandra), AWS IoT TwinMaker, Amazon RDS, and AWS Supply Chain by using advanced event selectors. For more information, see Data events.	December 20, 2023
Updated AWS managed policy	Updated the CloudTrailServiceRolePolicy managed policy to allow the following actions on an organization event data store when federation is disabled: `glue:DeleteTable` and `lakeformation:DeregisterResource` .	November 26, 2023
Added functionality	You can now can federate a CloudTrail Lake event data store to see the metadata associated with the event data store in the AWS Glue Data Catalog and run SQL queries on the event data using Amazon Athena. The table metadata stored in the AWS Glue Data Catalog lets the Athena query engine know how to find, read, and process the data that you want to query. For more information, see Federate an event data store.	November 26, 2023

Added functionality	You can now log CloudTrail data events for AWS Cloud Map by using advanced event selectors. For more information, see Logging data events.	November 16, 2023
Added functionality	You can now log CloudTrail data events on Amazon SQS messages by using advanced event selectors. For more information, see Logging data events.	November 16, 2023

Added functionality	CloudTrail Lake now offers two pricing options for event data stores: one-year extendable retention pricing and seven-year retention pricing. The pricing option determines the cost for ingesting and storing events, and the default and maximum retention period for the event data store. Before this release, all event data stores used the seven-year retention pricing option. You can switch an event data store from using the seven-year retention pricing option to using the one-year extendable retention pricing by using the CloudTrail console, AWS CLI, or the UpdateEventDataStore API operation. For more information about pricing options, see AWS CloudTrail Pricing and Event data store pricing options.	November 15, 2023

Added functionality	You can now collect Insights events in CloudTrail Lake. AWS CloudTrail Insights help AWS users identify and respond to unusual activity associated with API calls and API error rates by continuously analyzing CloudTrail management events. To collect Insights events in CloudTrail Lake, you need a source event data store that logs management events and enables Insights and a destination event data store that collects Insights events based upon unusual management event activity in the source event data store. For more information, see Create an event data store for CloudTrail Insights events and Logging Insights events.	November 9, 2023
Added service support	This release supports AWS Launch Wizard. For more information, see AWS service topics for CloudTrail and Logging AWS Launch Wizard API calls using AWS CloudTrail.	November 8, 2023

Added service support	This release supports Amazon Bedrock. For more information, see AWS service topics for CloudTrail and Log Amazon Bedrock API calls using AWS CloudTrail.	October 23, 2023
Added functionality	You can now log CloudTrail data events on Amazon CodeWhisperer customizations by using advanced event selectors. For more information, see Logging data events.	October 18, 2023
Added functionality	You can now log CloudTrail data events on Amazon Timestream databases and tables by using advanced event selectors. For more information, see Logging data events.	September 28, 2023
Added functionality	You can now log CloudTrail data events on Amazon SNS topics and platform endpoints by using advanced event selectors. For more information, see Logging data events.	September 28, 2023

Updated documentation	Added table to show the tasks that the management account, delegated administrator accounts, and member accounts within an AWS Organizations organization can perform in CloudTrail. For more information, see Organization delegated administrator.	September 25, 2023
Added service support	This release supports AWS Marketplace Agreements. For more information, see AWS service topics for CloudTrail and Logging Agreements API Calls using AWS CloudTrail.	September 1, 2023
Added functionality	You can now log CloudTrail data events on Amazon Kinesis video streams and Amazon SageMaker endpoints by using advanced event selectors. For more information, see Logging data events.	August 31, 2023
Added service support	This release supports AWS Application Transformation Service. AWS Application Transformation Service is a backend service used by services like AWS Microservice Extractor for .NET. For more information, see CloudTrail supported services and integrations.	August 26, 2023

Added functionality	You can now log CloudTrail data events on AWS Private CA Connector for Active Directory by using advanced event selectors. For more information, see Logging data events.	August 24, 2023
Updated documentation	Added new CloudTrail Lake scenarios to show how to create event data stores, view CloudTrail Lake dashboards, copy trail events to an event data store, view and run sample queries, and save query results to an Amazon S3 bucket using the AWS Management Console. For more information, see Scenarios for CloudTrail Lake	August 16, 2023
New Region support	CloudTrail expanded support to a new Region, the Israel (Tel Aviv) Region. For more information, see CloudTrail supported Regions.	August 1, 2023
Added service support	This release supports AWS HealthImaging. For more information, see CloudTrail supported services and integrations and Logging AWS HealthImaging API calls using AWS CloudTrail.	July 26, 2023

Added functionality	You can now log CloudTrail data events on AWS HealthImaging data stores by using advanced event selectors. For more information, see Logging data events.	July 26, 2023
Added functionality	You can now log CloudTrail data events on AWS Systems Manager control channels and Amazon Managed Blockchain networks by using advanced event selectors. For more information, see Logging data events.	June 21, 2023
Added functionality	You can now verify your CloudTrail Lake saved query results using the **aws cloudtrail verify-query-results** command. For more information, see Validate saved query results with the AWS CLI.	June 21, 2023
Added service support	This release supports Amazon Verified Permissions. For more information, see CloudTrail supported services and integrations and Logging Amazon Verified Permissions API calls using AWS CloudTrail.	June 13, 2023

Added functionality	You can now use CloudTrail Lake dashboards to visualize the events in an event data store. For more information, see View Lake dashboards.	June 13, 2023
Added functionality	You can now log CloudTrail data events on Amazon Verified Permissions policy stores by using advanced event selectors. For more information, see Logging data events.	June 13, 2023
Added functionality	You can now log CloudTrail data events on an Amazon CodeWhisperer profile by using advanced event selectors. For more information, see Logging data events.	June 6, 2023
Added functionality	You can now start and stop event ingestion on CloudTrail event data stores. For information about stopping event ingestion using the console, see Stop an event data store from ingesting events. For information about stopping event ingestion using the AWS CLI, see Stop ingestion on an event data store.	June 2, 2023

Added functionality	You can now log CloudTrail data events on an Amazon EMR write-ahead log workspace by using advanced event selectors. For more information, see Logging data events.	May 31, 2023
Added service support	This release supports Amazon Security Lake. For more information, see CloudTrail supported services and integrations and Logging Amazon Security Lake API calls using AWS CloudTrail.	May 30, 2023
Updated documentation	Updated CloudTrail userIdentity element topic to include an example and field descriptions for a request made on behalf of an IAM Identity Center user. For more information, see CloudTrail userIdentity element.	May 23, 2023
Updated documentation	This update supports the following patch release for the CloudTrail Processing Library: aws-cloudtrail-processing-library-1.6.1.jar. For more information, see Using the CloudTrail Processing Library and the CloudTrail Processing Library on GitHub.	May 23, 2023

Added functionality	CloudTrail Lake now supports all Presto functions and operators. For more information, see CloudTrail Lake SQL constraints.	May 9, 2023
Added functionality	You can now log CloudTrail data events on an Amazon GuardDuty detector by using advanced event selectors. For more information, see Logging data events and Logging Amazon GuardDuty API calls with AWS CloudTrail.	March 30, 2023
Updated documentation	Added new section about creating user-defined cost allocation tags for event data stores. For more information, see Creating user-defined cost allocation tags for CloudTrail Lake event data stores.	March 24, 2023
Added service support	This release supports AWS Telco Network Builder (AWS TNB). For more information, see CloudTrail supported services and integrations and Logging AWS Telco Network Builder API calls using AWS CloudTrail.	February 21, 2023

Added functionality	You can now log CloudTrail data events on Amazon Cognito identity pools by using advanced event selectors. For more information, see Logging data events.	February 15, 2023
Updated documentation	Added new section about the learning resources available for CloudTrail Lake. For more information, see Learning resources.	February 9, 2023
Added functionality	You can now create CloudTrail Lake integrations with event sources outside of AWS. You can log and store user activity data from any source in your hybrid environments, such as in-house or SaaS applications hosted on-premises or in the cloud, virtual machines, or containers. For more information, see Create an integration with an event source outside of AWS.	January 31, 2023
Added functionality	You can now log CloudTrail data events on CloudTrail `PutAuditEvents` activity on a CloudTrail Lake channel by using advanced event selectors. For more information, see Logging data events.	January 31, 2023

New Region support	CloudTrail expanded support to a new Region, the Asia Pacific (Melbourne) Region. For more information, see CloudTrail supported Regions.	January 24, 2023
Updated documentation	Added new section about managing data consistency in CloudTrail, see Managing data consistency in CloudTrail.	January 18, 2023
Added functionality	You can now log CloudTrai l data events on Amazon SageMaker feature stores by using advanced event selectors. For more informati on, see Logging data events.	December 27, 2022
Added service support	This release supports AWS Marketplace Discovery. See AWS CloudTrail Supported Services and Integrations.	December 15, 2022
Added functionality	You can now log CloudTrai l data events on Amazon SageMaker metrics experimen t trial components by using advanced event selectors. For more information, see Logging data events.	December 15, 2022

Added functionality	You can now create an event data store to include AWS Config configuration items, and use the event data store to investigate non-compliant changes to your production environments. For more information, see Create an event data store for AWS Config configuration items.	November 28, 2022
New Region support	CloudTrail expanded support to a new Region, the Asia Pacific (Hyderabad) Region. For more information, see CloudTrail supported Regions.	November 22, 2022
Added functionality	You can now log CloudTrail data events on Amazon FinSpace environments by using advanced event selectors. For more information, see Logging data events.	November 18, 2022
New Region support	CloudTrail expanded support to a new Region, the Europe (Spain) Region. For more information, see CloudTrail supported Regions.	November 16, 2022
New Region support	CloudTrail expanded support to a new Region, the Europe (Zurich) Region. For more information, see CloudTrail supported Regions.	November 9, 2022

Added functionality	The management account for an AWS Organizations organization can now add a delegated administrator to manage the organization's CloudTrail trails and event data stores. For more information, see Organization delegated administrator.	November 7, 2022
Added functionality	You can now enable AWS Key Management Service encryption for a CloudTrail Lake event data store. For more information, see Create an event data store.	November 7, 2022
Added functionality	You can now save CloudTrail Lake query results to an Amazon S3 bucket when you run a query. For more information about running a query, see Run a query and save query results. For more information about downloading query results, see Get and download saved query results.	October 21, 2022
Added functionality	You can now copy CloudTrail trail events to a CloudTrail Lake event data store. For more information, see Copying trail events to CloudTrail Lake.	September 19, 2022

Updated documentation	Added list of supported Amazon CloudWatch metrics for CloudTrail Lake. For more information, see Supported CloudWatch metrics.	September 16, 2022
Added functionality	You can now view CloudTrail service-linked channels using the AWS CLI. For more information, see Viewing service-linked channels for CloudTrail by using the AWS CLI.	September 9, 2022
New Region support	CloudTrail expanded support to a new Region, the Middle East (UAE) Region. For more information, see CloudTrail supported Regions.	August 30, 2022
Changed functionality	CloudTrail has changed the name of the managed policy `AWSCloudTrailReadOnlyAccess` to `AWSCloudTrail_ReadOnlyAccess`. Permissions in this policy have been scoped down. By default, the policy no longer grants permission to list all Amazon S3 buckets, AWS Lambda functions, or AWS KMS aliases. For more information, see Read-only access.	June 6, 2022

Changed functionality	As a security best practice, you can now add an `aws:SourceArn` or `aws:SourceAccount` condition key to an `s3:GetBucketAcl` ACL checking block in Amazon S3 bucket policies. For more information, see Configure Amazon S3 bucket policies for CloudTrail.	May 11, 2022
Changed functionality	Starting Feb 24, 2022, AWS CloudTrail began changing the `userAgent` and `sourceIPAddress` field values in any event that originated from an AWS Management Console session where a proxy client was used. For these events, CloudTrail replaces the values of the `userAgent` and `sourceIPAddress` fields with `AWS Internal`. CloudTrail made this change to standardize how it logs information for service actions across all AWS services. For more information, see CloudTrail record contents.	April 12, 2022
Added service support	This release supports Amazon GameSparks. See AWS CloudTrail Supported Services and Integrations.	March 24, 2022

Added service support	This release supports AWS App Mesh Envoy Management Service. See AWS CloudTrail Supported Services and Integrations.	March 18, 2022
Updated documentation	New query examples have been added for CloudTrail Lake, a new feature that lets you run fine-grained, multiple-field SQL queries on your events. Also, a new field, `BytesScanned` , has been added to the query metadata results of `DescribeQuery` and `GetQueryResults` operations. For more information, see Working with CloudTrail Lake.	March 4, 2022

Changed functionality	CloudTrail now removes the account ID of the Amazon S3 bucket owner in the `resources` block of a data event if both of the following conditions are met: the data event API call is from a different AWS account than the Amazon S3 bucket owner, and the API caller received an `AccessDenied` error that was only for the caller account. For more information, see Redacting bucket owner account IDs for data events called by other accounts.	March 3, 2022
Updated documentation	This update supports the following release for the CloudTrail Processing Library: Added support for implementing a custom S3 manager, event logging to log file parsing-related exceptions, support for parsing an optional `errorCode` field in `insightDetails` , and updated the account ID parsing regex to accept non-numerical values. For more information, see Using the CloudTrail Processing Library and the CloudTrail Processing Library on GitHub.	January 28, 2022

Added functionality	CloudTrail introduces CloudTrail Lake, a new feature that lets you run fine-grained, multiple-field SQL queries on your events. Events are aggregated into event data stores, which are immutable collections of events based on criteria that you select by applying advanced event selectors. For more information, see Working with CloudTrail Lake.	January 5, 2022
New Region support	CloudTrail expanded support to a new Region, the Asia Pacific (Jakarta) Region. For more information, see CloudTrail supported Regions.	December 13, 2021
Added service support	This release supports Amazon WorkSpaces Web. See AWS CloudTrail Supported Services and Integrations.	December 3, 2021
Added functionality	You can now log CloudTrail data events on AWS Glue tables created by Lake Formation by using advanced event selectors. For more information, see Logging data events.	November 30, 2021

Changed functionality	As a security best practice, you can now add an `aws:SourceArn` or `aws:SourceAccount` condition key to AWS KMS key policies and Amazon S3 bucket policies. For more information, see Configure AWS KMS key policies for CloudTrail and Configure Amazon S3 bucket policies for CloudTrail.	November 15, 2021
Added service support	This release supports AWS Resilience Hub. See AWS CloudTrail Supported Services and Integrations.	November 10, 2021
Added functionality	A new CloudTrail Insights event type is available: error rate Insights events. An error rate Insights event captures unusual activity on an error that occurs on APIs called in your account. For more information, see Logging Insights events for trails.	November 10, 2021
Added functionality	You can now log CloudTrai l data events on DynamoDB streams by using advanced event selectors. For more information, see Logging data events.	September 22, 2021

Added functionality	You can now log data events on Amazon S3 access points. You can log Amazon S3 access point data events by using advanced event selectors. For more information, see Logging data events.	August 24, 2021
Changed functionality	When you configure a trail to send notifications to Amazon SNS, CloudTrail adds a policy statement to your SNS topic access policy that allows CloudTrail to send content to an SNS topic. As a security best practice, we recommend adding an `aws:SourceArn` or `aws:SourceAccount` condition key to the CloudTrail policy statement. For more information, see Amazon SNS topic policy for CloudTrail.	August 16, 2021
Added service support	This release supports Amazon Route 53 Application Recovery Controller. See AWS CloudTrail Supported Services and Integrations.	July 27, 2021
Added functionality	You can now log data events on Amazon EBS direct APIs run on EBS snapshots. You can log Amazon EBS direct API data events by using advanced event selectors. For more information, see Logging data events.	July 27, 2021

Changed functionality	When CloudTrail processes data events, it preserves numbers in their original format, whether that is an integer (`int`) or a `float`. In events that have integers in the fields of a data event, CloudTrail historically processed these numbers as floats. Now, CloudTrail keeps the original format of integers in data events. For more information, see Using the CloudTrail Processing Library.	July 13, 2021
Added functionality	You can now exclude Amazon RDS Data API management events from your trails. For more information, see Logging management events for trails.	July 1, 2021
Added service support	This release supports AWS BugBust. See AWS CloudTrail Supported Services and Integrations.	June 24, 2021
Added service support	This release supports Amazon Managed Grafana and Amazon Managed Service for Prometheus. See AWS CloudTrail Supported Services and Integrations.	June 2, 2021

Added service support	This release supports AWS App Runner. See AWS CloudTrail Supported Services and Integrations.	May 18, 2021
Added service support	This release supports AWS Systems Manager Incident Manager. See AWS CloudTrail Supported Services and Integrations.	May 10, 2021
Updated documentation	This update describes data event logging requirements for AWS Config conformance packs, especially for compliance frameworks such as HIPAA or FedRAMP. For more information, see Logging data events.	May 7, 2021
Added service support	This release supports Service Quotas and Amazon EBS direct APIs. See AWS CloudTrail Supported Services and Integrations.	April 13, 2021
Added functionality	After an IAM administrator configures AWS STS, CloudTrail logs `sourceIdentity` information in events when users assume an IAM role, or perform any actions with the assumed role. For more information, see CloudTrail userIdentity Element.	April 13, 2021

Updated documentation	This update documents limits, in kilobytes (KB), for content in some CloudTrail event record fields. For more information, see CloudTrail record contents.	April 8, 2021
Added functionality	After an IAM administrator configures AWS STS, CloudTrail logs `sourceIdentity` information in events when users assume an IAM role, or perform any actions with the assumed role. For more information, see CloudTrail userIdentity Element.	April 6, 2021
Added functionality	You can now log data events on Amazon DynamoDB tables. You can log DynamoDB data events by using either event selectors or advanced event selectors. For more information, see Logging data events.	March 23, 2021
Added service support	This release supports Amazon Managed Workflows for Apache Airflow. See AWS CloudTrail Supported Services and Integrations.	March 22, 2021

Added functionality	You can now log data events on S3 Object Lambda access points if you have opted in to use advanced event selectors. For more information, see Logging data events.	March 18, 2021
Added service support	This release supports AWS Fault Injection Simulator. See AWS CloudTrail Supported Services and Integrations.	March 15, 2021
Added functionality	You can now log data events on Ethereum nodes in Amazon Managed Blockchain if you have opted in to use advanced event selectors. For more information, see Logging data events.	March 1, 2021
Added service support	This release supports Amazon Managed Blockchain and the preview of Ethereum for Managed Blockchain. See AWS CloudTrail Supported Services and Integrations.	February 4, 2021
Added service support	This release supports AWS Amplify. See AWS CloudTrail Supported Services and Integrations.	February 3, 2021
Added service support	This release supports Amazon Lookout for Metrics. See AWS CloudTrail Supported Services and Integrations.	February 1, 2021

Updated documentation	This update supports the following patch release for the CloudTrail Processing Library: Update the .jar file references in the user guide to use the latest version, aws-cloudtrail-processing-library-1.4.0.jar. For more information, see Using the CloudTrail Processing Library and the CloudTrail Processing Library on GitHub.	January 12, 2021
Added functionality	You can now log data events on Amazon S3 on AWS Outposts. For more informati on, see Logging data events.	December 21, 2020
Added service support	This release supports Amazon Lookout for Equipment, AWS Well-Architected Tool, and Amazon Location Service. See AWS CloudTrail Supported Services and Integrations.	December 16, 2020
Added service support	This release supports AWS IoT Greengrass V2. See AWS CloudTrail Supported Services and Integrations.	December 15, 2020
Added service support	This release supports Amazon EMR on EKS. See AWS CloudTrail Supported Services and Integrations.	December 10, 2020

Added service support	This release supports AWS Audit Manager and Amazon HealthLake. See AWS CloudTrail Supported Services and Integrations.	December 8, 2020
Added service support	This release supports Amazon Lookout for Vision. See AWS CloudTrail Supported Services and Integrations.	December 1, 2020
Added functionality	The AWS CloudTrail event version is now 1.08. Version 1.08 introduces new fields for CloudTrail. For more information, see CloudTrail record contents.	November 24, 2020
Added functionality	AWS CloudTrail introduces advanced event selectors for data events. Advanced event selectors allow finer-grained control over the data events that you log to your trail. You can include or exclude data events for specific AWS resources, and select specific APIs on those resources to log to your trail. For more information, see Logging data events.	November 24, 2020
Added service support	This release supports AWS Network Firewall. See AWS CloudTrail Supported Services and Integrations.	November 17, 2020

Added service support	This release supports AWS Trusted Advisor. See AWS CloudTrail Supported Services and Integrations.	October 22, 2020
Updated documentation	Added two new examples of event records for root user sign-in events. For more information, see AWS Console sign-in events.	October 13, 2020
Changed functionality	Permissions in the `AWSCloudTrail_FullAccess` policy have been narrowed. This policy no longer allows you to delete Amazon SNS topics or Amazon S3 buckets, and the `getObject` action has been removed. For more information, see Granting custom permissions for CloudTrail users.	September 29, 2020
Updated documentation	This update supports the following patch release for the CloudTrail Processing Library: Update the .jar file references in the user guide to use the latest version, aws-cloudtrail-processing-library-1.3.0.jar. For more information, see Using the CloudTrail Processing Library and the CloudTrail Processing Library on GitHub.	August 28, 2020

Added service support	This release supports AWS Outposts. See AWS CloudTrail Supported Services and Integrations.	August 28, 2020
Added functionality	AWS CloudTrail Insights introduces attribution fields for CloudTrail Insights events. Attribution fields show the top user identities, user agents, and error codes that are associated with the anomalous activity that triggers Insights events. For comparison, attribution fields also show the top user identities, user agents, and error codes associated with normal, or baseline, activity. For more information, see Logging Insights events.	August 13, 2020
Added functionality	The AWS CloudTrail console has a new look that's designed to make it easier to use. The AWS CloudTrail User Guide has been updated with changes to procedures for how to perform tasks in the console, such as creating trails, updating trails, and downloading event history.	August 13, 2020
Added service support	This release supports Amazon Interactive Video Service. See AWS CloudTrail Supported Services and Integrations.	July 15, 2020

Added service support	This release supports Amazon Honeycode. See AWS CloudTrail Supported Services and Integrations.	June 24, 2020
Added service support	This release supports Amazon Macie. See AWS CloudTrail Supported Services and Integrations.	May 19, 2020
Added service support	This release supports Amazon Kendra. See AWS CloudTrail Supported Services and Integrations.	May 13, 2020
Added service support	This release supports AWS IoT SiteWise. See AWS CloudTrail Supported Services and Integrations.	April 29, 2020
Added Region support	This release supports an additional Region: Europe (Milan). See AWS CloudTrail Supported Regions.	April 28, 2020
Added service and Region support	This release supports Amazon AppFlow. See AWS CloudTrail Supported Services and Integrations. Support has also been added for the Africa (Cape Town) Region. See AWS CloudTrail Supported Regions.	April 22, 2020

Added functionality	High-volume AWS KMS actions such as `Encrypt`, `Decrypt`, and `GenerateDataKey` are now logged as **Read** events. If you choose to log all AWS KMS events on your trail, and also choose to log **Write** management events, your trail logs relevant AWS KMS actions like `Disable`, `Delete` and `ScheduleKey`.	April 7, 2020
Added service support	This release supports Amazon CodeGuru Reviewer. See AWS CloudTrail Supported Services and Integrations.	February 7, 2020
Added service support	This release supports Amazon Managed Apache Cassandra Service. See AWS CloudTrail Supported Services and Integrations.	January 17, 2020
Added service support	This release supports Amazon Connect. See AWS CloudTrail Supported Services and Integrations.	December 13, 2019

Updated documentation	This update supports the following patch release for the CloudTrail Processing Library: Update the .jar file references in the user guide to use the latest version, aws-cloudtrail-processing-library-1.2.0.jar. For more information, see Using the CloudTrail Processing Library and the CloudTrail Processing Library on GitHub.	November 21, 2019
Added functionality	This release supports AWS CloudTrail Insights for helping you detect unusual activity in your account. See Logging Insights events for Trails.	November 20, 2019
Added functionality	This release adds an option for filtering AWS Key Management Service events out of a trail. See Creating a Trail.	November 20, 2019
Added service support	This release supports AWS CodeStar Notifications. See AWS CloudTrail Supported Services and Integrations.	November 7, 2019
Added functionality	This release supports adding tags when you create a trail in CloudTrail, whether you use the CloudTrail console or API. This release adds two new APIs, `GetTrail` and `ListTrails` .	November 1, 2019

Added service support	This release supports AWS App Mesh. See AWS CloudTrail Supported Services and Integrations.	October 17, 2019
Added service support	This release supports Amazon Translate. See AWS CloudTrail Supported Services and Integrations.	October 17, 2019
Documentation update	The Unsupported Services topic has been restored and updated to include only those AWS services that do not currently log events in CloudTrail. See CloudTrail Unsupported Services.	October 7, 2019
Documentation update	The documentation has been updated with changes to the `AWSCloudTrailFullAccess` policy. A policy example that shows equivalent permissions to `AWSCloudTrailFullAccess` has been updated to restrict the resources on which the `iam:PassRole` action can act to those matching the following condition statement: `"iam:PassedToService": "cloudtrail.amazonaws.com"`. See AWS CloudTrail Identity-Based Policy Examples.	September 24, 2019

Documentation update	The documentation has been updated with a new topic, Managing CloudTrail Costs, to help you get the log data you need out of CloudTrail while staying within a budget.	September 3, 2019
Added service support	This release supports AWS Control Tower. See AWS CloudTrail Supported Services and Integrations.	August 13, 2019
Added Region support	This release supports an additional Region: Middle East (Bahrain). See AWS CloudTrail Supported Regions.	July 29, 2019
Documentation update	The documentation has been updated with information about security for CloudTrail. See Security in AWS CloudTrail.	July 3, 2019
Added service support	This release supports AWS Ground Station. See AWS CloudTrail Supported Services and Integrations.	June 6, 2019
Added service support	This release supports AWS IoT Things Graph. See AWS CloudTrail Supported Services and Integrations.	June 4, 2019
Added service support	This release supports Amazon AppStream 2.0. See AWS CloudTrail Supported Services and Integrations.	April 25, 2019

Added Region support	This release supports an additional Region: Asia Pacific (Hong Kong). See AWS CloudTrail Supported Regions.	April 24, 2019
Added service support	This release supports Amazon Managed Service for Apache Flink. See AWS CloudTrail Supported Services and Integrations.	March 22, 2019
Added service support	This release supports AWS Backup. See AWS CloudTrail Supported Services and Integrations.	February 4, 2019
Added service support	This release supports Amazon WorkLink. See AWS CloudTrail Supported Services and Integrations.	January 23, 2019
Added service support	This release supports AWS Cloud9. See AWS CloudTrail Supported Services and Integrations.	January 21, 2019
Added service support	This release supports AWS Elemental MediaLive. See AWS CloudTrail Supported Services and Integrations.	January 19, 2019
Added service support	This release supports Amazon Comprehend. See AWS CloudTrail Supported Services and Integrations.	January 18, 2019

Added service support	This release supports AWS Elemental MediaPackage. See AWS CloudTrail Supported Services and Integrations.	December 21, 2018
Added Region support	This release supports an additional Region: EU (Stockholm). See AWS CloudTrail Supported Regions.	December 11, 2018
Documentation update	The documentation has been updated with information about supported and unsupported services. See AWS CloudTrail Supported Services and Integrations.	December 3, 2018
Added service support	This release supports AWS Resource Access Manager (AWS RAM). See AWS CloudTrail Supported Services and Integrations.	November 20, 2018
Updated functionality	This release supports creating a trail in CloudTrail that logs events for all AWS accounts in an organization in AWS Organizations. See Creating a Trail for an Organization.	November 19, 2018
Added service support	This release supports Amazon Pinpoint SMS and Voice API. See AWS CloudTrail Supported Services and Integrations.	November 16, 2018

Added service support	This release supports AWS IoT Greengrass. See AWS CloudTrail Supported Services and Integrations.	October 29, 2018
Updated documentation	This update supports the following patch release for the CloudTrail Processing Library: Update the .jar file references in the user guide to use the latest version, aws-cloudtrail-processing-library-1.1.3.jar. For more information, see Using the CloudTrail Processing Library and the CloudTrail Processing Library on GitHub.	October 18, 2018
Added functionality	This release supports using additional filters in **Event history**. See Viewing CloudTrail Events in the CloudTrail Console.	October 18, 2018
Added functionality	This release supports using Amazon Virtual Private Cloud (Amazon VPC) to establish a private connection between your VPC and AWS CloudTrail. See Using AWS CloudTrail with Interface VPC Endpoints.	August 9, 2018
Added service support	This release supports Amazon Data Lifecycle Manager. See AWS CloudTrail Supported Services and Integrations.	July 24, 2018

Added service support	This release supports Amazon MQ. See AWS CloudTrail Supported Services and Integrations.	July 19, 2018
Added service support	This release supports AWS Mobile CLI. See AWS CloudTrail Supported Services and Integrations.	June 29, 2018
AWS CloudTrail documentation history notification available through RSS feed	You can now receive notification about updates to the AWS CloudTrail documentation by subscribing to an RSS feed.	June 29, 2018

Earlier updates

The following table describes the documentation release history of AWS CloudTrail prior to June 29, 2018.

Change	Description	Release Date
Added service support	This release supports Amazon RDS Performance Insights. For more information, see CloudTrail Supported Services and Integrations.	June 21, 2018
Added functionality	This release supports logging all CloudTrail management events in Event history. For more information, see Working with CloudTrail Event history.	June 14, 2018
Added service support	This release supports AWS Billing and Cost Management. See CloudTrail supported services and integrations.	June 7, 2018

Change	Description	Release Date
Added service support	This release supports Amazon Elastic Container Service for Kubernetes (Amazon EKS). See CloudTrail supported services and integrations.	June 5, 2018
Updated documentation	This update supports the following patch release for the CloudTrail Processing Library: - Update the .jar file references in the user guide to use the latest version, aws-cloudtrail-processing-library-1.1.2.jar. For more information, see Using the CloudTrail Processing Library and the CloudTrail Processing Library on GitHub.	May 16, 2018
Added service support	This release supports AWS Billing and Cost Management. See CloudTrail supported services and integrations.	June 7, 2018
Added service support	This release supports Amazon Elastic Container Service for Kubernetes (Amazon EKS). See CloudTrail supported services and integrations.	June 5, 2018
Updated documentation	This update supports the following patch release for the CloudTrail Processing Library: - Update the .jar file references in the user guide to use the latest version, aws-cloudtrail-processing-library-1.1.2.jar. For more information, see Using the CloudTrail Processing Library and the CloudTrail Processing Library on GitHub.	May 16, 2018

Change	Description	Release Date
Added service support	This release supports AWS X-Ray. See CloudTrail supported services and integrations.	April 25, 2018
Added service support	This release supports AWS IoT Analytics. See CloudTrail supported services and integrations.	April 23, 2018
Added service support	This release supports Secrets Manager. See CloudTrail supported services and integrations.	April 10, 2018
Added service support	This release supports Amazon Rekognition. See CloudTrail supported services and integrations.	April 6, 2018
Added service support	This release supports AWS Private Certificate Authority (PCA). See CloudTrail supported services and integrations.	April 4, 2018
Added functionality	This release supports making it easier to search CloudTrail log files with Amazon Athena. You can automatically create tables for querying logs directly from the CloudTrail console, and use those tables to run queries in Athena. For more information, see CloudTrail supported services and integrations and Creating a Table for CloudTrail Logs in the CloudTrail Console.	March 15, 2018
Added service support	This release supports AWS AppSync. See CloudTrail supported services and integrations.	February 13, 2018
Added Region support	This release supports an additional Region: Asia Pacific (Osaka) (ap-northeast-3). See CloudTrail supported Regions.	February 12, 2018
Added service support	This release supports AWS Shield. See CloudTrail supported services and integrations.	February 12, 2018

Change	Description	Release Date
Added service support	This release supports Amazon SageMaker. See CloudTrail supported services and integrations.	January 11, 2018
Added service support	This release supports AWS Batch. See CloudTrail supported services and integrations.	January 10, 2018
Added functionality	This release supports extending the amount of account activity that is available in CloudTrail event history to 90 days. You can also customize the display of columns to improve the view of your CloudTrai l events. For more information, see Working with CloudTrail Event history.	December 12, 2017
Added service support	This release supports Amazon WorkMail. See CloudTrail supported services and integrations.	December 12, 2017
Added service support	This release supports Alexa for Business, AWS Elemental MediaConvert, and AWS Elemental MediaStore. See CloudTrail supported services and integrations.	December 1, 2017
Added functionality and documentation	This release supports logging data events for AWS Lambda functions. For more information, see Logging data events.	November 30, 2017
Added functionality and documentation	This release supports logging data events for AWS Lambda functions. For more information, see Logging data events.	November 30, 2017

Change	Description	Release Date
Added functionality and documentation	This release supports the following updates to the CloudTrail Processing Library: - Add support for Boolean identification of management events. - Update the CloudTrail event version to 1.06. For more information, see Using the CloudTrail Processing Library and the CloudTrail Processing Library on GitHub.	November 30, 2017
Added service support	This release supports AWS Glue. See CloudTrail supported services and integrations.	November 7, 2017
New documentation	This release adds a new topic, Quotas in AWS CloudTrail.	October 19, 2017
Updated documentation	This release updates the documentation of APIs supported in CloudTrail event history for Amazon Athena, AWS CodeBuild, Amazon Elastic Container Registry, and AWS Migration Hub.	October 13, 2017
Added service support	This release supports Amazon Chime. See CloudTrail supported services and integrations.	September 27, 2017
Added functionality and documentation	This release supports configuring data event logging for all Amazon S3 buckets in your AWS account. See Logging data events.	September 20, 2017
Added service support	This release supports Amazon Lex. See CloudTrail supported services and integrations.	August 15, 2017
Added service support	This release supports AWS Migration Hub. See CloudTrail supported services and integrations.	August 14, 2017

Change	Description	Release Date
Added functionality and documentation	This release supports CloudTrail being enabled by default for all AWS accounts. The past seven days of account activity are available in CloudTrail event history, and the most recent events appear on the console dashboard. The feature formerly known as **API activity history** has been replaced by **Event history**.	August 14, 2017
Added functionality and documentation	This release supports downloading events from the CloudTrail console on the API activity history page. You can download events in JSON or CSV format. For more information, see Downloading events.	July 27, 2017
Added functionality	This release supports logging Amazon S3 object level API operations in two additional Regions, Europe (London) and Canada (Central). For more information, see Working with CloudTrail log files.	July 19, 2017
Added service support	This release supports looking up APIs for Amazon CloudWatch Events in the CloudTrail API activity history feature.	June 27, 2017

Change	Description	Release Date
Added functionality and documentation	This release supports additional APIs in the CloudTrail API activity history feature for the following services: • AWS CloudHSM • Amazon Cognito • Amazon DynamoDB • Amazon EC2 • Kinesis • AWS Storage Gateway	June 27, 2017
Added service support	This release supports AWS CodeStar. See CloudTrail supported services and integrations.	June 14, 2017

Change	Description	Release Date
Added functionality and documentation	This release supports the following updates to the CloudTrail Processing Library: • Add support for different formats for SQS messages from the same SQS queue to identify CloudTrail log files. The following formats are supported: ◦ Notifications that CloudTrail sends to an SNS topic ◦ Notifications that Amazon S3 sends to an SNS topic ◦ Notifications that Amazon S3 sends directly to an SQS queue • Add support for the `deleteMessageUponFailure` property, which you can use to delete messages that can't be processed. For more information, see Using the CloudTrail Processing Library and the CloudTrail Processing Library on GitHub.	June 1, 2017
Added service support	This release supports Amazon Athena. See CloudTrail supported services and integrations.	May 19, 2017
Added functionality	This release supports sending data events to Amazon CloudWatch Logs. For more information about configuring your trail to log data events, see Data events. For more information about sending events to CloudWatch Logs, see Monitoring CloudTrail Log Files with Amazon CloudWatch Logs.	May 9, 2017

Change	Description	Release Date
Added service support	This release supports the AWS Marketplace Metering Service. See CloudTrail supported services and integrations.	May 2, 2017
Added service support	This release supports Amazon QuickSight. See CloudTrail supported services and integrations.	April 28, 2017
Added functionality and documentation	This release supports an updated console experience for creating new trails. You can now configure a new trail to log management and data events. For more information, see Creating a trail with the CloudTrail console.	April 11, 2017
Added documentation	If CloudTrail is not delivering logs to your S3 bucket or sending SNS notifications from some Regions in your account, you may need to update the policies. To learn more about updating your S3 bucket policy, see Common Amazon S3 policy configuration errors. To learn more about updating your SNS topic policy, see CloudTrail is not sending notifications for a Region.	March 31, 2017
Added service support	This release supports AWS Organizations. See CloudTrail supported services and integrations.	February 27, 2017
Added functionality and documentation	This release supports an updated console experienc e for configuring trails for logging management and data events. For more information, see Working with CloudTrail log files.	February 10, 2017
Added service support	This release supports Amazon Cloud Directory. See CloudTrail supported services and integrations.	January 26, 2017

Change	Description	Release Date
Added functionality and documentation	This release supports looking up APIs for AWS CodeCommit, Amazon GameLift, and AWS Managed Services in the CloudTrail API activity history.	January 26, 2017
Added functionality	This release supports integration with the AWS Health Dashboard. You can use the AWS Health Dashboard to identify if your trails are unable to deliver logs to an SNS topic or S3 bucket. This can occur when there is an issue with the policy for the S3 bucket or SNS topic. AWS Health Dashboard notifies you about the affected trails and recommends ways to fix the policy. For more information, see the AWS Health User Guide.	January 24, 2017
Added functionality and documentation	This release supports filtering by event source in the CloudTrail console. Event source shows the AWS service to which the request was made. For more information, see Viewing recent management events with the console.	January 12, 2017
Added service support	This release supports AWS CodeCommit. See CloudTrail supported services and integrations.	January 11, 2017
Added service support	This release supports Amazon Lightsail. See CloudTrail supported services and integrations.	December 23, 2016
Added service support	This release supports AWS Managed Services. See CloudTrail supported services and integrations.	December 21, 2016
Added Region support	This release supports the Europe (London) Region. See CloudTrail supported Regions.	December 13, 2016

Change	Description	Release Date
Added Region support	This release supports the Canada (Central) Region. See CloudTrail supported Regions.	December 8, 2016
Added service support	This release supports AWS CodeBuild See CloudTrail supported services and integrations. This release supports AWS Health. See CloudTrail supported services and integrations. This release supports AWS Step Functions. See CloudTrail supported services and integrations.	December 1, 2016
Added service support	This release supports Amazon Polly. See CloudTrail supported services and integrations.	November 30, 2016
Added service support	This release supports AWS OpsWorks for Chef Automate. See CloudTrail supported services and integrations.	November 23, 2016
Added functionality and documentation	This release supports configuring your trail to log read-only, write-only, or all events. CloudTrail supports logging Amazon S3 object level API operations such as `GetObject` , `PutObject` , and `DeleteObject` . You can configure your trails to log object level API operations. For more information, see Working with CloudTrail log files.	November 21, 2016
Added functionality and documentation	This release supports additional values for the `type` field in the `userIdentity` element: `AWSAccount` and `AWSService` . For more information, see the Fields for `userIdentity` .	November 16, 2016

Change	Description	Release Date
Added service support	This release supports Application Auto Scaling. See CloudTrail supported services and integrations.	October 31, 2016
Added Region support	This release supports the US East (Ohio) Region. See CloudTrail supported Regions.	October 17, 2016
Added functionality and documentation	This release supports logging non-API AWS service events. For more information, see AWS service events.	September 23, 2016
Added functionality and documentation	This release supports using the CloudTrail console to view resource types that are supported by AWS Config. For more information, see Viewing resources referenced with AWS Config.	July 7, 2016
Added service support	This release supports AWS Service Catalog. See CloudTrail supported services and integrations.	July 6, 2016
Added service support	This release supports Amazon Elastic File System (Amazon EFS). See CloudTrail supported services and integrations.	June 28, 2016
Added Region support	This release supports one additional Region: ap-south-1 (Asia Pacific (Mumbai)). See CloudTrail supported Regions.	June 27, 2016
Added service support	This release supports AWS Application Discovery Service. See CloudTrail supported services and integrations.	May 12, 2016
Added service support	This release supports CloudWatch Logs in the South America (São Paulo) Region. For more information, see Monitoring CloudTrail Log Files with Amazon CloudWatch Logs.	May 6, 2016
Added service support	This release supports AWS WAF. See CloudTrail supported services and integrations.	April 28, 2016

Change	Description	Release Date
Added service support	This release supports AWS Support. See CloudTrail supported services and integrations.	April 21, 2016
Added service support	This release supports Amazon Inspector. See CloudTrail supported services and integrations.	April 20, 2016
Added service support	This release supports AWS IoT. See CloudTrail supported services and integrations.	April 11, 2016
Added functionality and documentation	This release supports logging AWS Security Token Service (AWS STS) API calls made with Security Assertion Markup Language (SAML) and web identity federation. For more information, see Values for AWS STS APIs with SAML and web identity federation.	March 28, 2016
Added service support	This release supports AWS Certificate Manager. See CloudTrail supported services and integrations.	March 25, 2016
Added service support	This release supports Amazon Data Firehose. See CloudTrail supported services and integrations.	March 17, 2016
Added service support	This release supports Amazon CloudWatch Logs. See CloudTrail supported services and integrations.	March 10, 2016
Added service support	This release supports Amazon Cognito. See CloudTrail supported services and integrations.	February 18, 2016
Added service support	This release supports AWS Database Migration Service. See CloudTrail supported services and integrations.	February 4, 2016
Added service support	This release supports Amazon GameLift (Amazon GameLift). See CloudTrail supported services and integrations.	January 27, 2016

Change	Description	Release Date
Added service support	This release supports Amazon CloudWatch Events. See CloudTrail supported services and integrations.	January 16, 2016
Added Region support	This release supports one additional Region: ap-northeast-2 (Asia Pacific (Seoul)). See CloudTrail supported Regions.	January 6, 2016
Added service support	This release supports Amazon Elastic Container Registry (Amazon ECR). See CloudTrail supported services and integrations.	December 21, 2015
Added functionality and documentation	This release supports turning on CloudTrail across all Regions and support for multiple trails per Region. For more information, see Working with CloudTrail trails.	December 17, 2015
Added service support	This release supports Amazon Machine Learning. See CloudTrail supported services and integrations.	December 10, 2015
Added functionality and documentation	This release supports log file encryption, log file integrity validation, and tagging. For more informati on, see Encrypting CloudTrail log files with AWS KMS keys (SSE-KMS), Validating CloudTrail log file integrity, and Updating a trail with the CloudTrail console.	October 1, 2015
Added service support	This release supports Amazon OpenSearch Service. See CloudTrail supported services and integrations.	October 1, 2015
Added service support	This release supports Amazon S3 bucket level events. See CloudTrail supported services and integrations.	September 1, 2015
Added service support	This release supports AWS Device Farm. See CloudTrai l supported services and integrations.	July 13, 2015

Change	Description	Release Date
Added service support	This release supports Amazon API Gateway. See CloudTrail supported services and integrations.	July 9, 2015
Added service support	This release supports CodePipeline. See CloudTrail supported services and integrations.	July 9, 2015
Added service support	This release supports Amazon DynamoDB. See CloudTrail supported services and integrations.	May 28, 2015
Added service support	This release supports CloudWatch Logs in the US West (N. California) Region. For more informati on about CloudTrail support for CloudWatch Logs monitoring, see Monitoring CloudTrail Log Files with Amazon CloudWatch Logs.	May 19, 2015
Added service support	This release supports AWS Directory Service. See CloudTrail supported services and integrations.	May 14, 2015
Added service support	This release supports Amazon Simple Email Service (Amazon SES). See CloudTrail supported services and integrations.	May 7, 2015
Added service support	This release supports Amazon Elastic Container Service See CloudTrail supported services and integrations.	April 9, 2015
Added service support	This release supports AWS Lambda. See CloudTrail supported services and integrations.	April 9, 2015
Added service support	This release supports Amazon WorkSpaces. See CloudTrail supported services and integrations.	April 9, 2015

Change	Description	Release Date
	This release supports the lookup of AWS activity captured by CloudTrail (CloudTrail events). You can look up and filter events in your account related to creation, modification, or deletion. To look up these events, you can use the CloudTrail console, the AWS Command Line Interface (AWS CLI), or the AWS SDK. For more information, see Working with CloudTrail Event history.	March 12, 2015
Added service support and new documentation	This release supports Amazon CloudWatch Logs in the Asia Pacific (Singapore), Asia Pacific (Sydney), Asia Pacific (Tokyo), and Europe (Frankfurt) Regions. For more information, see Sending events to CloudWatch Logs.	March 5, 2015
New documentation	A new section that describes CloudTrail support for AWS Security Token Service (AWS STS) regional endpoints has been added to the CloudTrail Concepts page.	February 17, 2015
Added service support	This release supports Amazon Route 53. See CloudTrail supported services and integrations.	February 11, 2015
Added service support	This release supports AWS Config. See CloudTrail supported services and integrations.	February 10, 2015
Added service support	This release supports AWS CloudHSM. See CloudTrail supported services and integrations.	January 8, 2015
Added service support	This release supports AWS CodeDeploy. See CloudTrail supported services and integrations.	December 17, 2014
Added service support	This release supports AWS Storage Gateway. See CloudTrail supported services and integrations.	December 16, 2014

Change	Description	Release Date
Added Region support	This release supports one additional Region: us-gov-west-1 (AWS GovCloud (US-West)). See CloudTrail supported Regions.	December 16, 2014
Added service support	This release supports Amazon S3 Glacier. See CloudTrail supported services and integrations.	December 11, 2014
Added service support	This release supports AWS Data Pipeline. See CloudTrail supported services and integrations.	December 2, 2014
Added service support	This release supports AWS Key Management Service. See CloudTrail supported services and integrations.	November 12, 2014
New documentation	A new section, Monitoring CloudTrail Log Files with Amazon CloudWatch Logs, has been added to the guide. It describes how to use Amazon CloudWatch Logs to monitor CloudTrail log events.	November 10, 2014
New documentation	A new section, Using the CloudTrail Processing Library, has been added to the guide. It provides information about how to write a CloudTrail log processor in Java using the AWS CloudTrail Processing Library.	November 5, 2014
Added service support	This release supports Amazon Elastic Transcoder. See CloudTrail supported services and integrations.	October 27, 2014
Added Region support	This release supports one additional region: eu-central-1 (Europe (Frankfurt)). See CloudTrail supported Regions.	October 23, 2014
Added service support	This release supports Amazon CloudSearch. See CloudTrail supported services and integrations.	October 16, 2014

Change	Description	Release Date
Added service support	This release supports Amazon Simple Notificat ion Service. See CloudTrail supported services and integrations.	October 09, 2014
Added service support	This release supports Amazon ElastiCache. See CloudTrail supported services and integrations.	September 15, 2014
Added service support	This release supports Amazon WorkDocs. See CloudTrail supported services and integrations.	August 27, 2014
Added new content	This release includes a topic that discusses logging sign-in events. See AWS Management Console sign-in events.	July 24, 2014
Added new content	The **eventVersion** element for this release has been upgraded to version 1.02 and three new fields have been added. See CloudTrail record contents.	July 18, 2014
Added service support	This release supports Auto Scaling (see CloudTrail supported services and integrations).	July 17, 2014
Added Region support	This release supports three additional Regions: ap-southeast-1 (Asia Pacific (Singapore)), ap-northeast-1 (Asia Pacific (Tokyo)), sa-east-1 (South America (São Paulo)). See CloudTrail supported Regions.	June 30, 2014
Additional service support	This release supports Amazon Redshift. See CloudTrai l supported services and integrations.	June 10, 2014
Added service support	This release supports AWS OpsWorks. See CloudTrail supported services and integrations.	June 5, 2014
Added service support	This release supports Amazon CloudFront. See CloudTrail supported services and integrations.	May 28, 2014

Change	Description	Release Date
Added Region support	This release supports three additional Regions: us-west-1 (US West (N. California)), eu-west-1 (Europe (Ireland)), ap-southeast-2 (Asia Pacific (Sydney)). See CloudTrail supported Regions.	May 13, 2014
Added service support	This release supports Amazon Simple Workflow Service. See CloudTrail supported services and integrations.	May 9, 2014
Added new content	This release includes topics that discuss sharing log files between accounts. See Sharing CloudTrail log files between AWS accounts.	May 2, 2014
Added service support	This release supports Amazon CloudWatch. See CloudTrail supported services and integrations.	April 28, 2014
Added service support	This release supports Amazon Kinesis. See CloudTrail supported services and integrations.	April 22, 2014
Added service support	This release supports AWS Direct Connect. See CloudTrail supported services and integrations.	April 11, 2014
Added service support	This release supports Amazon EMR. See CloudTrail supported services and integrations.	April 4, 2014
Added service support	This release supports Elastic Beanstalk. See CloudTrail supported services and integrations.	April 2, 2014
Additional service support	This release supports AWS CloudFormation. See CloudTrail supported services and integrations.	March 7, 2014
New guide	This release introduces AWS CloudTrail.	November 13, 2013