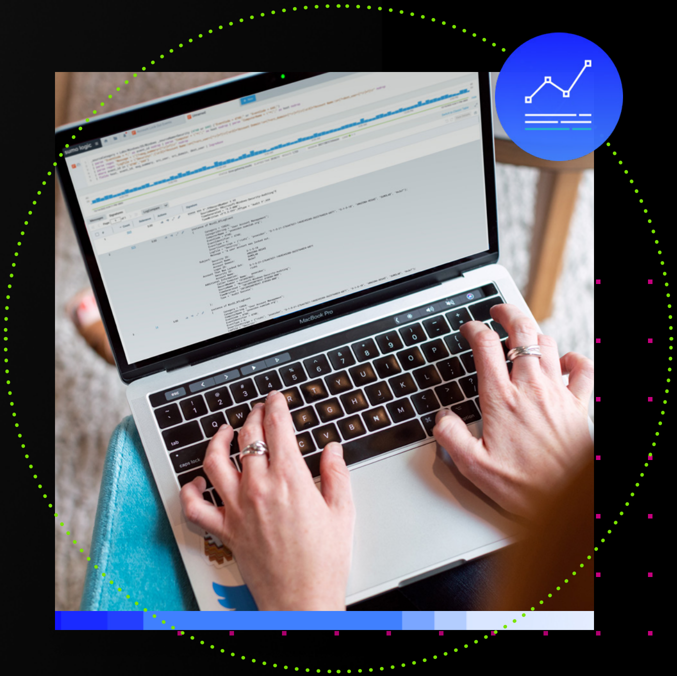
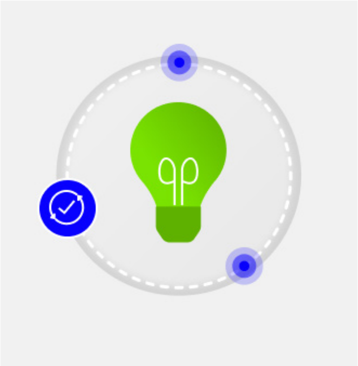
Log management best practices

How to turn your log data into useful log analytics

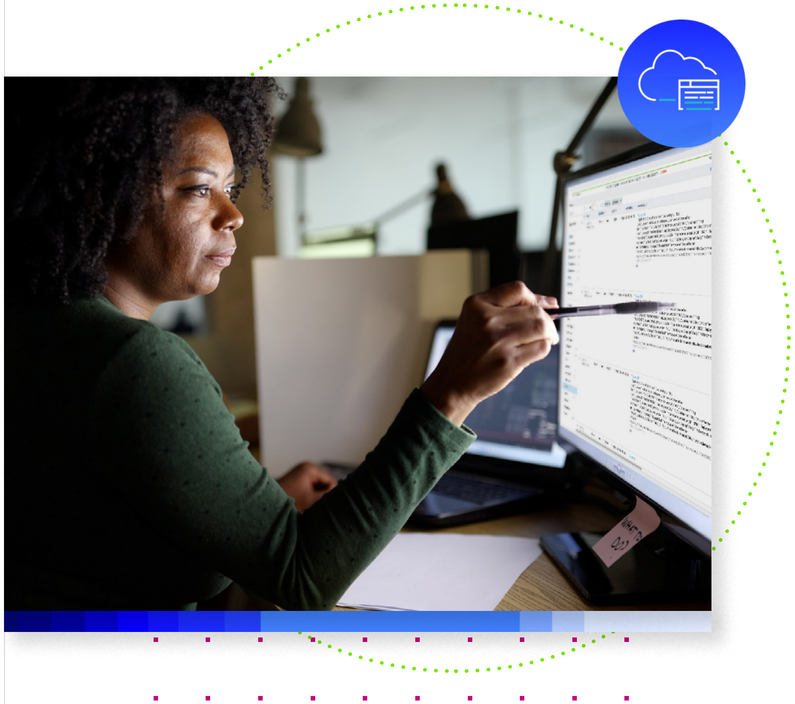


sumo logic

What's inside



Turn your log data into opportunity



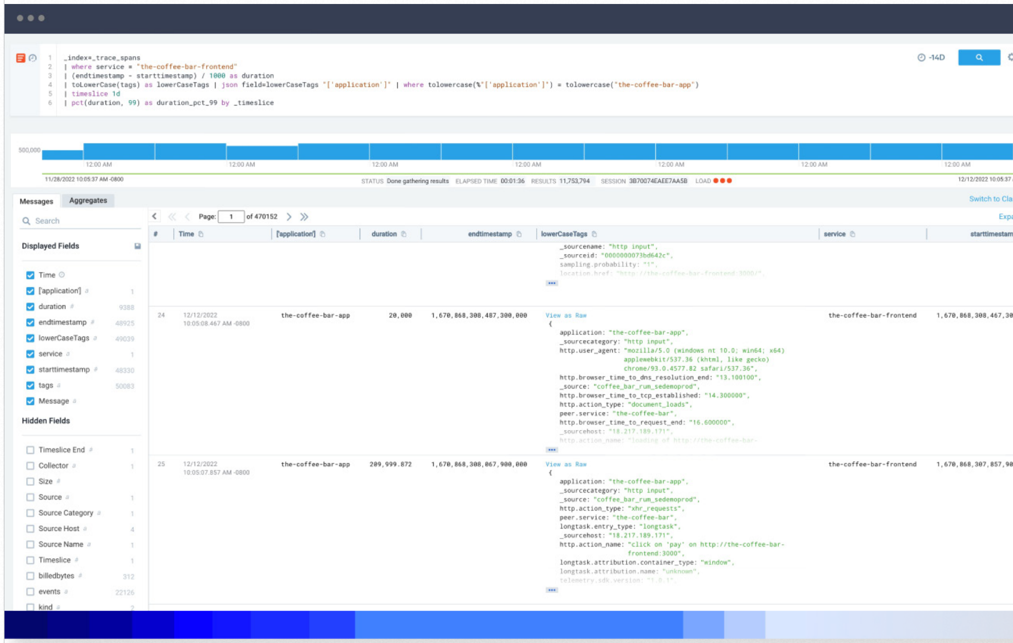
Maintaining log management best practices requires constant awareness of how technology advances.

Your cloud app doesn't exist in isolation. Log types like authentications, connections, access and errors provide essential data that can inform new operations efficiencies and the best security approach. But these benefits manifest when you manage logs appropriately, store and analyze them on a single platform and have insights visible on a single screen. Today's log management best practices take the effort off DevOps and security teams and bring the insights to you.

This ebook will help teams gather and draw insights from log data that benefit your business. Uncover hidden opportunities to move faster, work smarter and collaborate more efficiently.

Logging and monitoring is unique to each company

Centralizing your logs from across the network is an accepted best practice for networks of all types — cloud, multi-cloud, hybrid or on-premises. Converting **logs to a structured format** to analyze by machine learning is a second emerging best practice helping companies across industries claim a competitive advantage. In the following sections, we'll cover both of these and some application-specific details.



```
_index=_trace_spans
| where service = "the-coffee-bar-frontend"
| (endtimestamp - starttimestamp) / 1000 as duration
| toLowerCase(tags) as lowerCaseTags | json field=lowerCaseTags "['application']" | where tolowercase(%"['application']") = tolowercase("the-coffee-bar-app")
| timeslice 1d
| pct(duration, 99) as duration_pct_99 by _timeslice
```

STATUS Done gathering results ELAPSED TIME 00:01:36 RESULTS 11,753,794 SESSION 3B70074EAEE7AA5B LOAD ●●●

Displayed Fields

- ☑ Time
- ☑ ['application'] 1
- ☑ duration 9388
- ☑ endtimestamp 48925
- ☑ lowerCaseTags 49039
- ☑ service 1
- ☑ starttimestamp 48330
- ☑ tags 50083
- ☑ Message

Hidden Fields

- ☐ Timeslice End 1
- ☐ Collector 1
- ☐ Size
- ☐ Source 1
- ☐ Source Category 1
- ☐ Source Host 4
- ☐ Source Name 1
- ☐ Timeslice 1
- ☐ billedbytes 312
- ☐ events 22126
- ☐ kind 2

```
_sourcename: "http input",
_sourceid: "0000000073bd642c",
sampling.probability: "1",
location.href: "http://the-coffee-bar-frontend:3000/",
```

#	Time	['application']	duration	endtimestamp	lowerCaseTags	service	starttimestam
24	12/12/2022 10:05:08.467 AM -0800	the-coffee-bar-app	20,000	1,670,868,308,487,300,000	View as Raw	the-coffee-bar-frontend	1,670,868,308,467,30
25	12/12/2022 10:05:07.857 AM -0800	the-coffee-bar-app	209,999.872	1,670,868,308,067,900,000	View as Raw	the-coffee-bar-frontend	1,670,868,307,857,90

```
{
   application: "the-coffee-bar-app",
   _sourcecategory: "http input",
   http.user_agent: "mozilla/5.0 (windows nt 10.0; win64; x64)
                    applewebkit/537.36 (khtml, like gecko)
                    chrome/93.0.4577.82 safari/537.36",
   http.browser_time_to_dns_resolution_end: "13.100100",
   _source: "coffee_bar_rum_sedemoprod",
   http.browser_time_to_tcp_established: "14.300000",
   http.action_type: "document_loads",
   peer.service: "the-coffee-bar",
   http.browser_time_to_request_end: "16.600000",
   _sourcehost: "18.217.189.171",
   http.action_name: "loading of http://the-coffee-bar-
```

```
{
   application: "the-coffee-bar-app",
   _sourcecategory: "http input",
   _source: "coffee_bar_rum_sedemoprod",
   http.action_type: "xhr_requests",
   peer.service: "the-coffee-bar",
   longtask.entry_type: "longtask",
   _sourcehost: "18.217.189.171",
   http.action_name: "click on 'pay' on http://the-coffee-bar-
                      frontend:3000",
   longtask.attribution.container_type: "window",
   longtask.attribution.name: "unknown",
   telemetry.sdk.version: "1.0.1",
```

■

Deep interrogation through robust log queries of all datasets accelerates threat detection and troubleshooting performance issues.

Centralized logging best practices

Centralized logging takes place through a tool that collects data from diverse sources across the IT infrastructure. Data moves from various sources into a public cloud or other third-party storage solution. DevOps and security teams should aggregate logs across sources, including:

→ Switches, routers and access points to identify misconfiguration and other issues.

→ Web server data to reveal the finer details of customer acquisition and behavior.

→ Security data such as logs from intrusion systems and firewalls to pinpoint and address security risks before they become concerns.

→ Cloud infrastructure and services logs to assess if current resources are adequate and help address latency issues or other needs.

→ Application-level logs to help determine where errors may require resolution, i.e., payment gateways, in-app databases and other features.

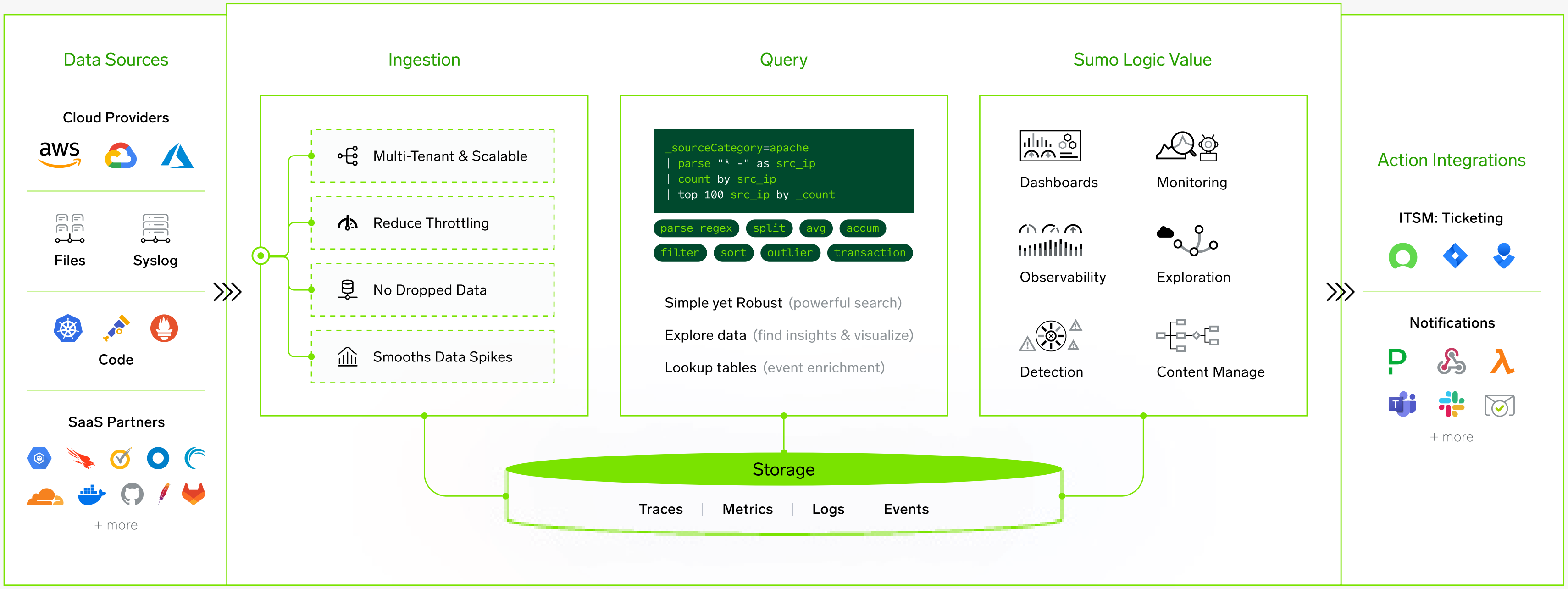
→ Server logs from databases like PostgreSQL, MongoDB, MySQL and more reveal user experience issues and other performance challenges.

→ Container logs from Kubernetes or Docker provide close monitoring of a dynamic environment to help organizations balance resource demands with costs and address issues quickly.

mongoDB. MySQL.

kubernetes docker.

A powerful platform



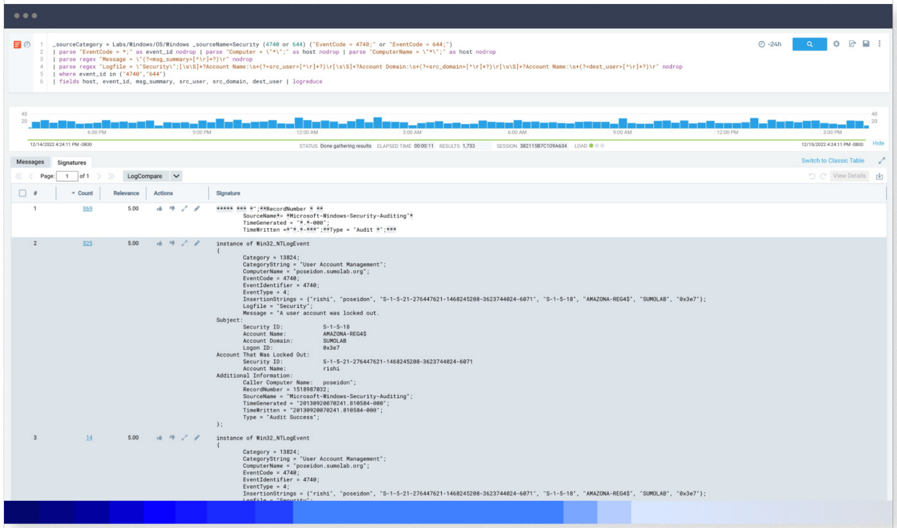
```
_sourceCategory=apache
| parse "* -" as src_ip
| count by src_ip
| top 100 src_ip by _count
```

■

Modern log management and analytics platforms must aggregate data from multiple cloud-based sources, leverage machine learning to extract critical insights quickly and integrate with multiple applications for broad awareness and analysis.

What is the benefit of centralized logging?

Relocating logs into a centralized tool means information silos dissolve. Instead, log correlation, normalization and analysis within a single tool generate insights in context. Such oversight allows teams to proactively identify and address the application's critical needs, including SLA, performance issues and availability problems.



```
_sourceCategory = Labs/Windows/OS/Windows _sourceName=Security (4740 or 644) ("EventCode = 4740;" or "EventCode = 644;")
| parse "EventCode = *;" as event_id nodrop | parse "Computer = \"*\";" as host nodrop | parse "ComputerName = \"*\";" as host nodrop
| parse regex "Message = \"(?<msg_summary>[^\r]+?)\r" nodrop
| parse regex "Logfile = \"Security\";[\s\S]+?Account Name:\s+(?<src_user>[^\r]+?)\r[\s\S]+?Account Domain:\s+(?<src_domain>[^\r]+?)\r[\s\S]+?Account Name:\s+(?<dest_user>[^\r]+?)\r" nodrop
| where event_id in ("4740","644")
| fields host, event_id, msg_summary, src_user, src_domain, dest_user | logreduce
```

```
12/14/2022 4:24:11 PM -0800    STATUS Done gathering results   ELAPSED TIME 00:00:11   RESULTS 1,733   SESSION 382115B7C109A634   LOAD    12/15/2022 4:24:11 PM -0800
```

```
#   Count   Relevance   Signature
1   869     5.00        ***** *** *";**RecordNumber * **
                                SourceName*= *Microsoft-Windows-Security-Auditing"*
                                TimeGenerated = "*.*-000";
                                TimeWritten =*"*.*-***";**Type = "Audit *";***
2   825     5.00        instance of Win32_NTLogEvent
                        {
                                Category = 13824;
                                CategoryString = "User Account Management";
                                ComputerName = "poseidon.sumolab.org";
                                EventCode = 4740;
                                EventIdentifier = 4740;
                                EventType = 4;
                                InsertionStrings = {"rishi", "poseidon", "S-1-5-21-276447621-1468245208-3623744024-6071", "S-1-5-18", "AMAZONA-REG4$", "SUMOLAB", "0x3e7"};
                                Logfile = "Security";
                                Message = "A user account was locked out.
                        Subject:
                                Security ID:            S-1-5-18
                                Account Name:           AMAZONA-REG4$
                                Account Domain:         SUMOLAB
                                Logon ID:               0x3e7
                        Account That Was Locked Out:
                                Security ID:            S-1-5-21-276447621-1468245208-3623744024-6071
                                Account Name:           rishi
                        Additional Information:
                                Caller Computer Name:   poseidon";
                                RecordNumber = 1518987032;
                                SourceName = "Microsoft-Windows-Security-Auditing";
                                TimeGenerated = "20130920070241.810584-000";
                                TimeWritten = "20130920070241.810584-000";
                                Type = "Audit Success";
                        };
3   14      5.00        instance of Win32_NTLogEvent
                        {
                                Category = 13824;
                                CategoryString = "User Account Management";
                                ComputerName = "poseidon.sumolab.org";
                                EventCode = 4740;
                                EventIdentifier = 4740;
                                EventType = 4;
                                InsertionStrings = {"rishi", "poseidon", "S-1-5-21-276447621-1468245208-3623744024-6071", "S-1-5-18", "AMAZONA-REG4$", "SUMOLAB", "0x3e7"};
```

■

Sumo Logic offers detection tools like LogReduce® and LogCompare to extract critical insights from mountains of data.

The right log management platform makes log analysis significantly simpler.

Suppose you're wondering which tool to use for logging in microservices architecture with multiple clouds or pulling logs from third-party systems.

Sumo Logic centralizes logs and offers detection tools to extract critical insights from mountains of data. Powerful outlier detection lets you eliminate the white noise and find the true unknowns affecting your system. Automated service mapping reveals the exact dependencies and traces connected to a specific issue, generating a high-level picture of the health of your microservices infrastructure. Root Cause Explorer uses correlated telemetry to help experts go from being aware of an issue to understanding its potential roots within a few moments.

Structured logging best practices

Structured logging means converting data into a format capable of machine learning analysis. You can't query log data as a plain text file or filter to deliver insights. Structured logs in a more parsable format like XML or JavaScript Object Notation (JSON) make the data usable for monitoring, troubleshooting and investigations.

Some logs may come to you with a predetermined structure from the vendor, while others will be custom logs created internally. Having all these logs in a consistent format is essential, and this conversion is what log management means in practice for many developers and organizations. Even if the custom logs from your platform export in a structured format, other logs require updating and reformatting to allow mastery of the data.

Structured logging makes data more usable by making logs more specific. Key values identified and named within the log equip an algorithm to detect patterns or filter results. Log structure benefits from the same five elements as every other good piece of reporting: who, what, where, when and why.

Five elements of good log reporting

Who
Username(s) or other ID associated with the log.

What
Error code, impact level, source IP, etc.

When
Specific gateway or hostname within your infrastructure.

Where
Timestamp

Why
This is what your log management platform will help you figure out!

Remember, logs are not data written down in plain text. Instead, document a log in a specific format that humans and automation understand.

Here are examples of unstructured and structured logs. Structure helps clarify the meaning of your logs. Learn more about structured logging.

```
<4>Nov 21 2:53:17 192.168.0.1 fluentd[11111]:
[error] Syslog test
```

Unstructured log record generated through the Google Cloud Platform.

This log contains a wealth of information, but it may not be immediately obvious in the above example what each part of the message is referring to and some important details might have been left out. We can introduce structured logging to help clarify the meaning of this log message and make it more readable for machines.

```
JSON payload: { “pri”: “6”, “host”: “192.168.0.1”,
“ident”: “fluentd”, “pid”: “11111”, “message”:
“[error] Syslog test” }
```

Example of using the JSON format to change the structure of the payload.

As you can see, the modified payload contains essentially the same information as the initial message. The key difference is that attributes have been identified, named, and presented as a set of ordered pairs along with the corresponding values. Now, a data analysis program can use these attributes to filter search results or to detect patterns in the data.

Application logging best practices

Generally speaking, another high-level logging best practice is that you should never write your logs to files manually. Automated output paired with parsing tools helps you convert the logs from an unstructured to a structured format.

```
2017-04-10 09:50:32 -0700 dan12345 10.0.24.123 GET
/checkout/flights/ credit.payments.io Success 2
241.9
```

↓

```
{ timestamp: 2017-04-10 09:50:32 -0700,username:
dan12345,source_ip: 10.0.24.123,method:
GET,resource: /checkout/flights/,gateway: credit.
payments.io,audit: Success,flights_purchased:
2,value: 241.98,}
```

Example of Sumo Logic converting an unstructured log entry into a structered format like JSON.

Here are some more short mentions of basic best practices specific to certain programming languages, network structures or applications:

Logging best practices in C

When logging in C or C#, make sure logs go to the cloud, not the console. Use structured logging to preserve future users' contextual and diagnostic data.

REST API logging best practices

The REST API assembly of resources helps systems exchange information online. The JSON log structure is the best approach to REST API logging.

Logging best practices for Node.js

Node.js functions for server-side programming, where efficient logging can sometimes be a puzzle. You can automate the inclusion of timestamps, tags and other information in these logs.

Learn more about Node.js logging best practices.

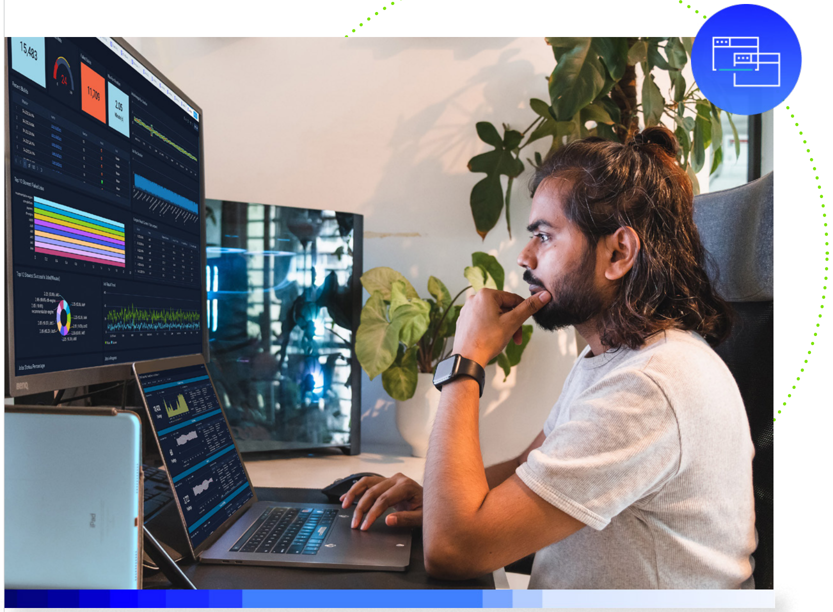
AWS centralized logging best practices

Amazon Web Services (AWS) logging best practices start with centralizing logs, analytics and security to keep software running efficiently and safely.

Learn more about AWS monitoring and logging.

Use log analytics to identify risk

Log analytics in cybersecurity refers to keeping track of unexpected events and occurrences to identify and manage risks. Log analysis turns this record keeping into valuable insights about your system's security and threats. Analyze security logs at recurring intervals, sometimes with development sprints, to provide insights and reveal errors to address. Security information and event management (SIEM) is a specialized function within security log analytics, and includes behavior analytics and case management.



The following pages includes some of the most important security practices pertaining to log analytics.

Firewall logs

Today's firewalls deliver rich data on threats, application types, command and control (C2) and more. Make sure to monitor all firewall logs, both internal and external.

Proxy and web filtering logs

If your firewall does not include these logs, it's important to monitor IP, URL and domain information for threat hunters to understand any connections to bad locations. User-agent logs are also invaluable in untangling major breaches and issues..

Network security products

If you have standalone systems for other network data like intrusion protection (IPS), intrusion detection (IDS) or network data loss prevention (DLP), centralize these logs for analysis with the rest of the information.

Deep packet inspection (DPI)

DPI is a type of data processing that inspects data sent over a computer network in detail and may take actions such as alerting, blocking, re-routing, or logging. These logs help users track events they otherwise might not have.

User access

Tracking a user's Windows authentication, single sign-on and Active Directory are great sources to tie one user to the event in the system, even if they change IP addresses in the middle of their activity.

Endpoint security solutions

Security logs from each device connected to your network can save experts time addressing alerts. For instance, if an alert from one or more endpoints corresponds with other anomalies, this provides more information for the investigation.

Threat intelligence

Threat intelligence includes logs and other data connected to recent threats at other organizations. You can access some data for free, while paid lists are sometimes better maintained. Introducing this data to your log analytics platform trains the algorithm to recognize similar patterns or behaviors faster in the future.

Careful maintenance and monitoring of these logs is a best practice and for US federal government agencies and contractors, it may be part of NIST 800-53 logging requirements

The National Institute of Standards and Technology (NIST) is an agency of the US Department of Commerce. NIST’s mission is to enable innovation and competitiveness. NIST 800-53 logging and monitoring requirements are some of the controls that create secure and resilient systems where this growth can occur. However, they are not the only ones! Using a log management system like Sumo Logic supports compliance for standards like FedRAMP Moderate, SOC 2 TYPE 2, HIPAA, PCI DSS and ISO 27001.

Compliance and certifications

PCI DSS

FEDRAMP

ISO 27001

AICPA SOC 2

GDPR

HIPAA

Sumo Logic currently holds the following certifications for its platform to put the safety of your data first.

Enable and improve log analysis

Once you have logs managed, organized and parsed, applying analytics best practices generates insights and actionable steps that translate the effort into better business outcomes. Data centralization, automated tagging, pattern recognition, correlation analysis and artificial ignorance emerge as some of the best practices for log analysis.

Sumo Logic delivers a turnkey SaaS analytics platform with these best practices implemented and ready to leverage.

Here's what each boils down to on a basic level

1 **Data centralization**
Bring all logs together in one centralized platform.

2 **Automated tagging and classification**
Logs should be classified automatically and ordered for future analysis.

3 **Pattern recognition and actionable reporting**
Use machine learning to identify log patterns and get insight into the next best steps.

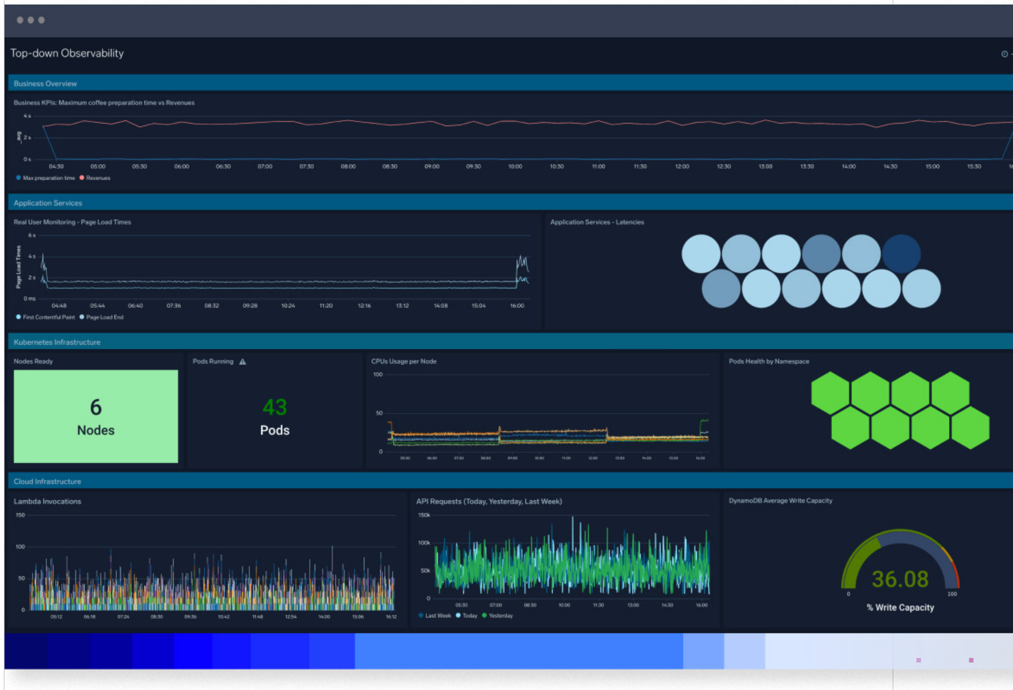
4 **Machine-learning-driven correlation analysis**
Collating logs from different sources gives a complete picture of each event and allows the identification of connections between disconnected data.

5 **Artificial ignorance**
Use tools trained to ignore routine logs and send alerts when something new occurs or something planned does not occur as scheduled.

with Sumo Logic

Sumo Logic is on a mission to help organizations achieve their applications and operations' reliability and security goals. Trusted by thousands of global companies to secure and protect against modern threats, we provide real-time insights to improve application performance and monitor and troubleshoot quickly. This visibility helps teams move from reactive to proactive and turn overwhelming log data into untapped opportunities.

With hundreds of out-of-the-box native integrations, Sumo Logic makes managing and analyzing log files easy. Centralizing logs in one place empowers teams to accelerate innovation without compromising reliability or security. We offer a 30-day free trial with free training and certification; no credit card required. Contact us or sign up today to explore the platform with your data and logs.



Sumo Logic.
The infinite power of log analytics.

About Sumo Logic

Sumo Logic, Inc. (NASDAQ: SUMO) empowers the people who power modern, digital business. Through its SaaS analytics platform, Sumo Logic enables customers to deliver reliable and secure cloud-native applications. The Sumo Logic Continuous Intelligence Platform™ helps practitioners and developers ensure application reliability, secure and protect against modern security threats, and gain insights into their cloud infrastructures. Customers around the world rely on Sumo Logic to get powerful real-time analytics and insights across observability and security solutions for their cloud-native applications. For more information, visit: SUMOLOGIC.COM

sumo logic