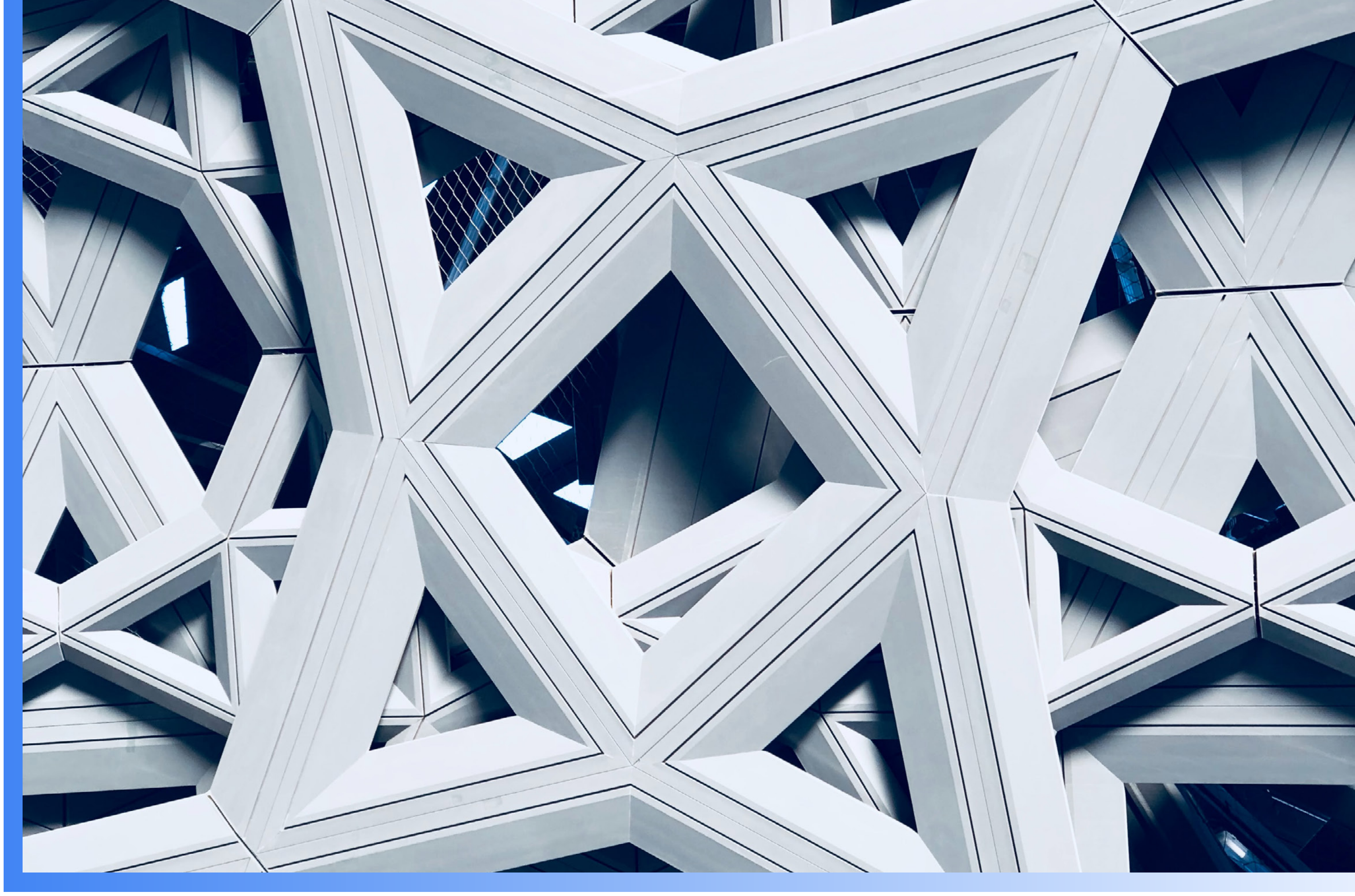


Data governance: Principles for securing and managing logs

Best practices for data residency, retention and access control

Google Cloud

About this whitepaper

This whitepaper is designed to provide cloud architects and IT leaders guidance for how to meet their data governance requirements with respect to logs data in Google Cloud. We will present an overview of the four stages of the lifecycle of logs and how data governance can be woven into each stage, walk through common requirements, and offer best practices we've used with Google Cloud customers to help them meet data governance challenges. In a companion whitepaper, Logs data: A step by step guide for overcoming common compliance challenges, we provide sample solution guides for four of the most common data governance requirements.

Data Governance requirement	Cloud Logging solution	Target audience
Data residency (e.g. GDPR or FedRAMP)	Store logs in a regional bucket	Cloud architect or Cloud admin (There should be no impact on developers or IT operators, provided they have appropriate access to view logs.)
Access to search across all logs in organization (e.g. compliance audit)	Aggregated sink at org level	
Long term retention of logs (e.g PCI DSS)	Configurable retention per log bucket	
Granular access control (e.g. role based access control restricting access to HIPAA data)	Row level access via log views; field level access via field restrictions on log buckets	

Four of the most common scenarios for logs data governance

Limitations around logs data governance

Please note that saved, shared, and recent queries, dynamically generated short links for sharing links from the cloud console, and log based metrics are not yet covered by Cloud Logging data residency guarantees. Avoid using these features to store sensitive data. Error Reporting, a tool which automatically detects anomalies in log entries, does not yet support data residency guarantees so will not work on logs that are stored in a regional rather than a global bucket.

We are invested in your success. The information and best practices in this guide are meant to support you in building solutions that would be used as part of your comprehensive data governance strategy. It is your responsibility to ensure your operations are compliant with any applicable laws and compliance regulations. We plan to keep doing our part by building solutions that help you get there.

Background

As more organizations move to the cloud, the volume of machine generated data has grown exponentially, both in volume and importance to the teams who rely on it. Software engineers and Site Reliability Engineers (SREs) rely on logs to develop new applications and troubleshoot existing apps to meet reliability targets. Security operators depend on logs to find and address threats and meet compliance needs. And business teams leverage logs for invaluable insights that can fuel growth. But first logs must be collected, processed, stored and analyzed.

Many organizations have found self-hosted log management solutions to be expensive and difficult to manage at scale. Yet log management in the cloud can also be challenging as organizations need the flexibility to adapt their logging solution to ensure that sensitive logs comply with their region’s regulations or internal corporate governance.

Cloud Logging, along with Cloud Monitoring and several other tools, are part of the Cloud operations suite of managed service offerings. They are built on the same observability platforms used by all of Google that handle over 16 million metrics queries per second, 2.5 exabytes of logs per month, and over 14 quadrillion metric points on disk in 2020[1]. They also provide capabilities to help customers achieve a variety of regulatory, compliance, and governance requirements with respect to their observability data (metrics, logs and traces).

1. https://cloud.google.com/blog/products/management-tools/cloud-operations-suite-gets-21-new-features

Logs data governance through the 4 stages of the logging lifecycle

Nearly every company in every industry relies on logs for different use cases including security, compliance, troubleshooting and business intelligence. But different industries and geographies have various requirements when it comes to how logs are stored and accessed. Here are a few examples:

Data residency examples

We have seen customers with data residency requirements from all around the world:

- European companies are often required to keep logs data in the European Union (EU) or even in a specific country to comply with General Data Protection Regulation (GDPR).
- An Australian telecommunications company needs to keep their logs in Australia.
- A large Canadian healthcare provider needs to keep their logs in Canada.
- Suppliers to the US government need to comply with FedRAMP High and keep their logs in the United States.

Centralized log management examples

Savvy organizations from financial institutions to retailers to cloud-native startups have internal governance requiring audit logs from across their organization to be searchable to follow audit trails, identify anomalies or threats.

Access examples

Healthcare providers, financial institutions, and retailers often have strict regulations around access to user data.

Retention examples

Retailers are often required to keep logs for at least a year with a requirement that at least 90 days be immediately available. Financial institutions often require logs be retained for years in case of an audit.

We can address all of these logs data governance challenges by looking at the four-stage lifecycle of logs data and the requirements of an ideal logging system at each stage:

Log generation and collection

We need a system that will collect all relevant logs no matter what system or application generates them or whether they are from Google Cloud, another cloud provider or on prem.

Log processing and routing

We need a system that can process logs according to our compliance requirements, discard irrelevant logs, and make copies of logs for different use cases and requirements.

Log storage and retention

We need a system that will allow us to choose where, and for how long, logs will be retained.
This stage is particularly important for data residency requirements such as FedRAMP High or GDPR.

Log analysis and access management

We need a log management system that offers access control of least privilege, limiting sensitive logs to only users who need this information for their roles. The system hides implementation details of your compliance configuration from your end users. Additionally, the system needs to search across multiple regions as needed to gain insights from the logs.

Let's look into each of these stages and explore how you can address key logs data governance challenges with Cloud Logging. We'll also share best practices we've gathered from different industries.

01

Log generation & collection

Many customers have applications that run both inside and outside of Google Cloud, including other public clouds or their own data centers. Cloud Logging supports collecting logs from all of these sources, a critical part of comprehensive compliance strategies.

Here are some of the most common classes of logs supported by Cloud Logging and how they are generated:

Google Cloud platform logs

Service-specific logs that can help you debug and troubleshoot issues, as well as better understand the Google Cloud services you're using.

- Generation: Basic Google Cloud platform logs are collected automatically for most services, and for Google Kubernetes Engine (GKE), agents that provide comprehensive logs are automatically installed with each new node provisioned. If you need more in-depth platform logs for your virtual machines, you can install an agent that will collect more granular data.

Audit logs

Cloud Audit Logs include Admin Activity, Data Access, System Event and Access Transparency logs. Together, these logs provide complete audit trails of administrative changes and data accesses of your Google Cloud resources including any access by Google.

- Generation: Audit logs are generated automatically and made available in Cloud Logging.

Within Google Cloud, most platform logs and audit logs are collected automatically, simplifying the compliance effort. A few high volume logs are off by default and do need to be manually enabled. Turn on data access audit logs across your Google Cloud organization and VPC flow logs for critical subnetworks.

Multi-cloud and on prem logs

Cloud Logging can collect logs from other cloud services providers, including Microsoft Azure and Amazon Web Services (AWS) or on prem.

- Generation: Multi-cloud and on-prem logs can be captured by installing the on-prem and hybrid cloud agent.

Apps logs

Generated by user software, services, or applications: since virtually all applications produce logs, Cloud Logging supports a variety of shapes and sizes for logs including both structured and unstructured log entries.

- Generation: User logs: You can send logs directly to the Cloud Logging API, usually by using client libraries or using one of the logging agents mentioned above.

Install a logging agent on all VMs, whether on GCP, other clouds, or on prem and configure it to collect the logs that are important to you.

The enemy of our ideal log collection system is noisy high volume logs which increase the resources needed to reliably collect and process logs. Since logs are machine generated, it's possible to generate huge volumes of data.

The ideal log collection system will be invisible, allowing developers and IT operators to see logs from their applications reliably, with minimal overhead and in real time.

Guidance for structured versus unstructured logs

We often get asked by customers for Google's best practices around logging standards.

"Different teams at Google have different approaches to logging," says Marc Unangst, Director of Engineering for Cloud Logging. "For debugging-oriented logs, we recommend developers use severity levels consistently to improve troubleshooting, and include relevant troubleshooting information in the log entry. For example, "error: parameter foo=5 out of range: must be between 0 and 3; see <link>" is much more useful to a developer than simply logging "error: bad parameter foo". We also encourage the use of structured logs wherever possible. For logs retained for more than a couple of weeks, structured logs are a requirement to help comply with privacy and legal regulations, such as PCI or GDPR. Logging common information like request ID or user ID in a standard field also makes it easier to stitch together logs from different microservices when troubleshooting across your stack."

Use severity levels consistently within your application so that you can easily identify actionable log messages. This also makes it easier to control costs by downsampling less important logs.

"We also encourage the use of structured logs wherever possible."

"Structured logs are a requirement to help comply with privacy and legal regulations."

The difference between structured and unstructured logs is illustrated with the example below. You can see the fields that are created with structured logs and how those fields can be used for access control. In the sample transaction below, you could create an access control rule to limit who can view the customer's email address.

Unstructured logs data	Structured logs data

`...`
`textPayload``:`
`alice@test.com purchased 4 widgets.`
`...`
	`...`
	`jsonPayload``: {`
	`"customer": "alice@test.com"`
	`"action": "purchased"`
	`"quantity": "4"`
	`"item": "widgets"`
	`}`
	`...`

Encourage developers to use structured logs where possible. This allows you to more easily analyze your logs and promotes better logging hygiene, reducing the chance of logging sensitive information such as passwords. It also makes it possible to restrict access to sensitive data in your logs.

Once you have a strategy around what to log, how to log it and you are sending logs data to the Cloud Logging APIs via agents or client libraries, you're ready to move on to the next stage of the logging lifecycle, processing and routing logs.

02

Log processing & routing

Processing logs and sending them to different destinations based on compliance requirements plays a critical part of most organizations' data governance strategies. Managing a log pipeline such as Kafka can be expensive and time consuming. Fortunately, Google Cloud offers this as a managed service without cost in the Logs Router.

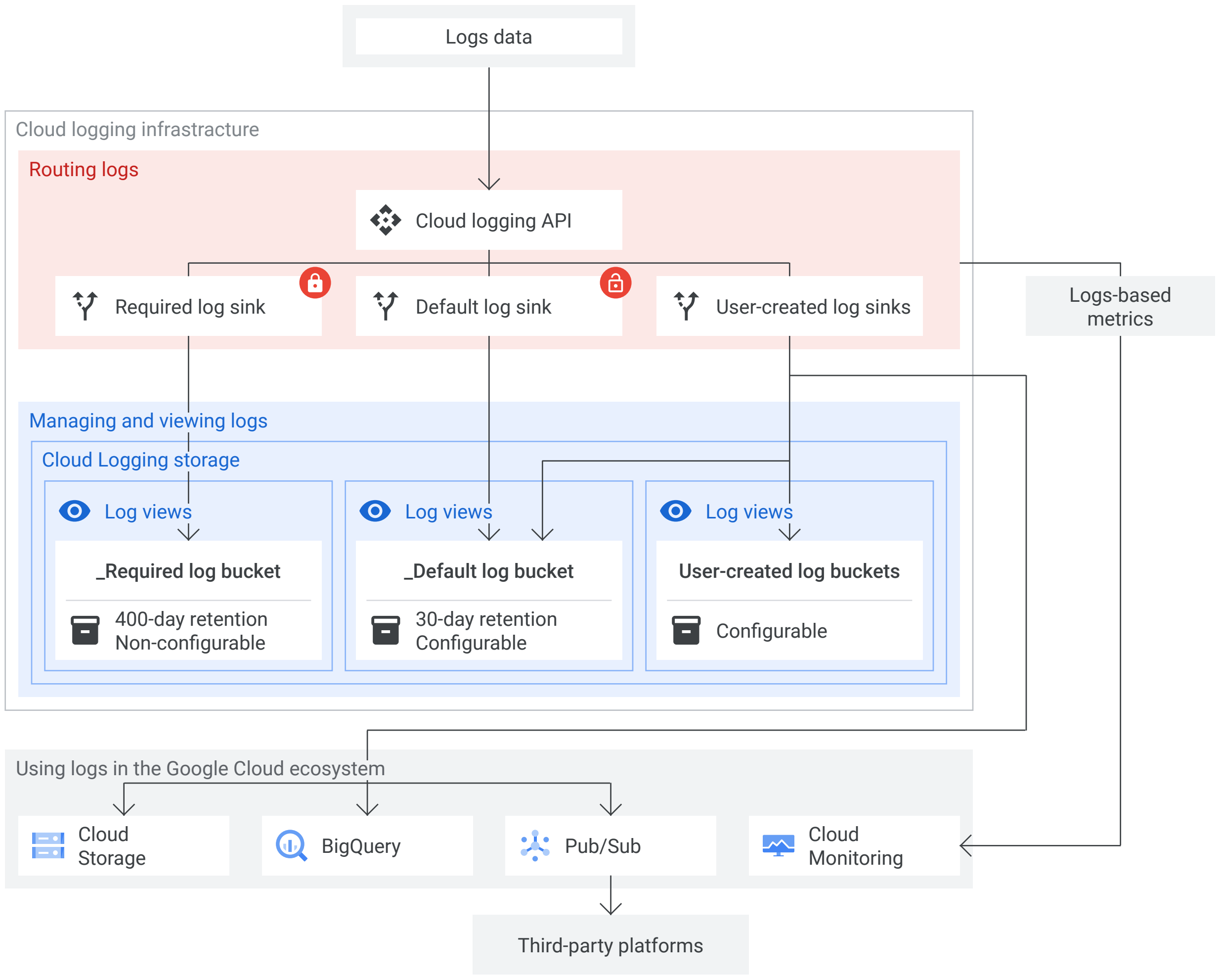
Logs Router is a stream processing powerhouse receiving trillions of logs per day and determines what to do with the logs by evaluating each log entry against rules defined by you (log sinks). These rules determine which log entries to discard, which log entries to store and make available within Cloud Logging, and which log entries to route to supported destinations like BigQuery, Google Cloud Storage, Cloud Pub/Sub. Logs may also be routed to other Google services you use that leverage logs such as Security Command Center Threat Detection or Eventarc. Finally, the Logs Router can do aggregations on logs using rules you define with logs-based metrics.

Use an aggregated log sink at the org or folder level to collect logs from all projects, especially for security use cases. It's particularly important to centralize logs that are relevant for compliance (e.g. audit logs, PCI logs, etc.) in a logging project so that even if the projects generating the logs are deleted, the logs will be available.

Here are some examples of how customers use the power of the Logs Router to achieve their data governance rules:

- The ADEO data platform SRE team uses Cloud Logging as its universal back end for logs across Google Cloud services, on-premises, and SaaS services, automating compliant turnup of new projects using Cloud Logging APIs. In addition to keeping logs in Cloud Logging, ADEO copies important logs to BigQuery for advanced analytics, reporting and to leverage BigQuery ML.
- To meet existing internal security needs, several large financial institutions send security related logs such as audit logs to an on prem SIEM using a log sink to Pub/Sub.

- A tech firm with independent teams of developers running their own apps within a shared GKE cluster split logs into multiple projects owned by individual teams for increased visibility on costs per team and to control access.
- From startups to Fortune 100 corporations, excluding low importance chatty logs helps control costs and improve the user experience with a higher signal to noise ratio in logs.
- A publishing company uses log based metrics to alert when configuration is changed to ensure that their system always stays in compliance.
- Companies around the world are required to keep logs in a specific country. As an example, an energy company was required to keep logs in Europe redirected default log sinks to a regionalized log buckets in `europe-west1` for projects running in Europe.
- Several large financial institutions running in Google Cloud centralize audit logs using an aggregated org level log sink to have a single pane of glass for troubleshooting security issues.



An overview of how Google Cloud's operations suite routes logs (source)

We've architected the Logs Router to process the massive scale of logs in real-time with incredibly high reliability including buffering against service or network disruptions impacting any log sink. In order to do this, the log router temporarily stores a copy of the log entries. The Logs Router also always runs alongside our Cloud Logging API in every region where you can write log entries to our API. This ensures that within the log router, logs are kept and processed in the region in which they are received.

Automate the setup of log sinks and log buckets for new projects to comply with your requirements regarding where logs are stored.

Within Google Cloud, log entries will automatically be received within the region in which the resource is running[2]. For example, if you're running a GKE cluster in `europe-west2`, your logs will automatically be received in `europe-west2` without any additional configuration.

Best Practices: When sending logs from outside of Google Cloud, use a regional endpoint, e.g. `europe-west2-logging.googleapis.com`, to guarantee that a log entry is received in a specific region.

Now that your log sinks are configured to send the logs you want to each destination, let's look at how storing logs within Cloud Logging can help you achieve your data governance goals.

As of May 2021, the Logs Router does not yet run in `asia-east2` so logs generated in this region will be redirected to `asia-east1`.

03

Log storage & retention

Log buckets are the constructs in your Google Cloud projects that store and organize your logs data. From the name, log buckets may sound like Cloud Storage buckets, but they are built on the same logging tech stack we've been using to deliver your logs in real time with advanced indexing and optimizations for timestamps. However, they have many attributes in common with their counterparts in Cloud Storage, including the ability to:

- Set retention — 1 day to 10 years.

Configure retention on a log bucket to match your compliance requirements.

- Lock a bucket to prevent anyone from deleting logs or reducing the retention period of the bucket.

If you have compliance requirements to prove logs have not been modified or deleted, consider locking the log bucket. However, keep in mind that this cannot be undone should requirements change in the future or if unintended data is sent to this log bucket so this is best used with well structured logs that you control.

- Select a region for your bucket — choose the Google Cloud region for your bucket or choose global, which means that Google will pick a region for you.

Log buckets offer zonal separation in regions that supports it today.

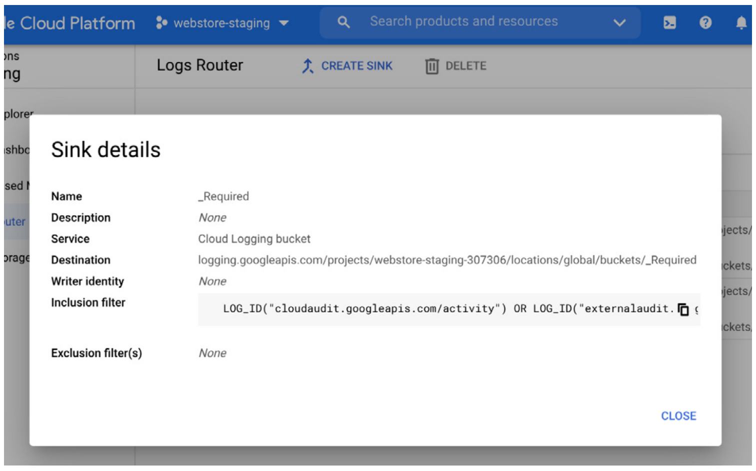
Log buckets serve many purposes including centralizing or decentralizing logs, managing retention according to compliance needs, controlling access to sensitive logs, and managing logs data residency.

Set up organization policies so that all new custom log buckets will be created in approved regions.

A Canadian retailer needed to store logs for 13 months to meet their compliance and auditing requirements. A system like Splunk or Elasticsearch can be very expensive for long term retention since the costs scale with the total amount of data stored. They were able to easily meet their retention goal with Cloud Logging without needing to move logs in and out of cold storage, delivering a better experience for their developers.

Logs data governance & system generated log buckets and log sinks

When you create a new project in Google Cloud, you will notice two global buckets created by the system, `_Required` and `_Default`, in your project along with two matching log sinks. You can also create your own log buckets. Let's look at these more closely to see how compliance sensitive customers use log buckets.



Sink details

Name	_Required
Description	*None*
Service	Cloud Logging bucket
Destination	logging.googleapis.com/projects/webstore-staging-307306/locations/global/buckets/_Required
Writer identity	*None*
Inclusion filter	`LOG_ID("cloudaudit.googleapis.com/activity") OR LOG_ID("externalaudit.`
Exclusion filter(s)	*None*

*A screenshot of the sink details for the **_Required** bucket.*

The `_Required` bucket - system generated

The `_Required` bucket receives logs defined by the `_Required` sink. If you inspect the log sink from the log router page, you'll see it includes a filter to capture certain audit logs.

Specifically, this is the filter for Admin Activity audit logs, System Event audit logs, and Access Transparency logs, **not your content (user logs/application logs data, platform logs, multi-cloud and on-prem logs and Data Access audit logs)**. The `_Required` bucket contains three types of logs.

Admin activity	System event	Access transparency
Record API calls modyfying **configuration** or **metadata**	Record API calls for **Google Cloud** actions modifying **configuration**	Audit **Google access** to your resources with access justification
Example record	**Example record**	**Example record**
Service: `storage.googleapis.com`	**Object:** `/instances/XYZ`	**Object:** `/buckets/XYZ`
Method: `setIamPolicy`	**Action:** `Instance migrated during Compute Engine maintenance`	**Action:** `READ`
Actor: `devops-service-account`	**Actor:** `compute.instances.migrateOnHostMaintenance`	**Reason:** `Ticket #12345`
Always enabled	Always enabled	Needs to be enabled

As you can see from the example, each of these logs contain critical information answering how Google Cloud APIs are used by your Google Cloud users or Google employees. Although the `_Required` bucket is global, since the logs contain only Google Cloud generated system audit trails, the data is not subject to data residency requirements.

This audit trail is important to providing you a secure experience. Because of this, neither the `_Required` sink, nor the `_Required` bucket can be changed, and the logs in the `_Required` bucket are stored free of charge for 400 days.

The `_Default` bucket - system generated

Upon creation, the `_Default` bucket is configured to receive the logs from the `_Default` sink which includes all other ingested logs in a Google Cloud project (such as user logs/application logs data, platform logs, multi-cloud and on-prem logs and Data Access audit logs), that are not held in the `_Required` bucket.

Unlike the `_Required` log bucket, the `_Default` log bucket retention can be changed and the `_Default` sink can be disabled and replaced or redirected entirely away from the `_Default` log bucket. This is an important step to meet data residency requirements, and we illustrate it in our step by step guide available here.

Create a custom log bucket for the region in which you need to store your logs and redirect your log sinks to the new log bucket instead of the global `_Default` bucket. Automate this step for new projects.
The `_Required` bucket only contains logs about system events, Google actions and administrative actions (Create, Update, Delete) on GCP APIs.

User-created log buckets and sinks - not system generated

User generated buckets give you more control over how your logs are retained and accessed as well as where your logs are stored. You can configure location, retention and advanced access control for these buckets. User-created log sinks can be used to route any combination of logs to any destination — from a user generated bucket to a _ Default log bucket in a different project or even to other Google Cloud services such as BigQuery, Cloud Pub/Sub or Cloud Storage. For an example of how user-created log buckets can help meet data residency, access control and centralized analysis requirements, see our step by step guide.

Now we're collecting logs, processing them, and storing them in a way that meets our data governance needs. Let's explore the final stage — getting value from our logs with log analysis.

04

Log analysis and access management

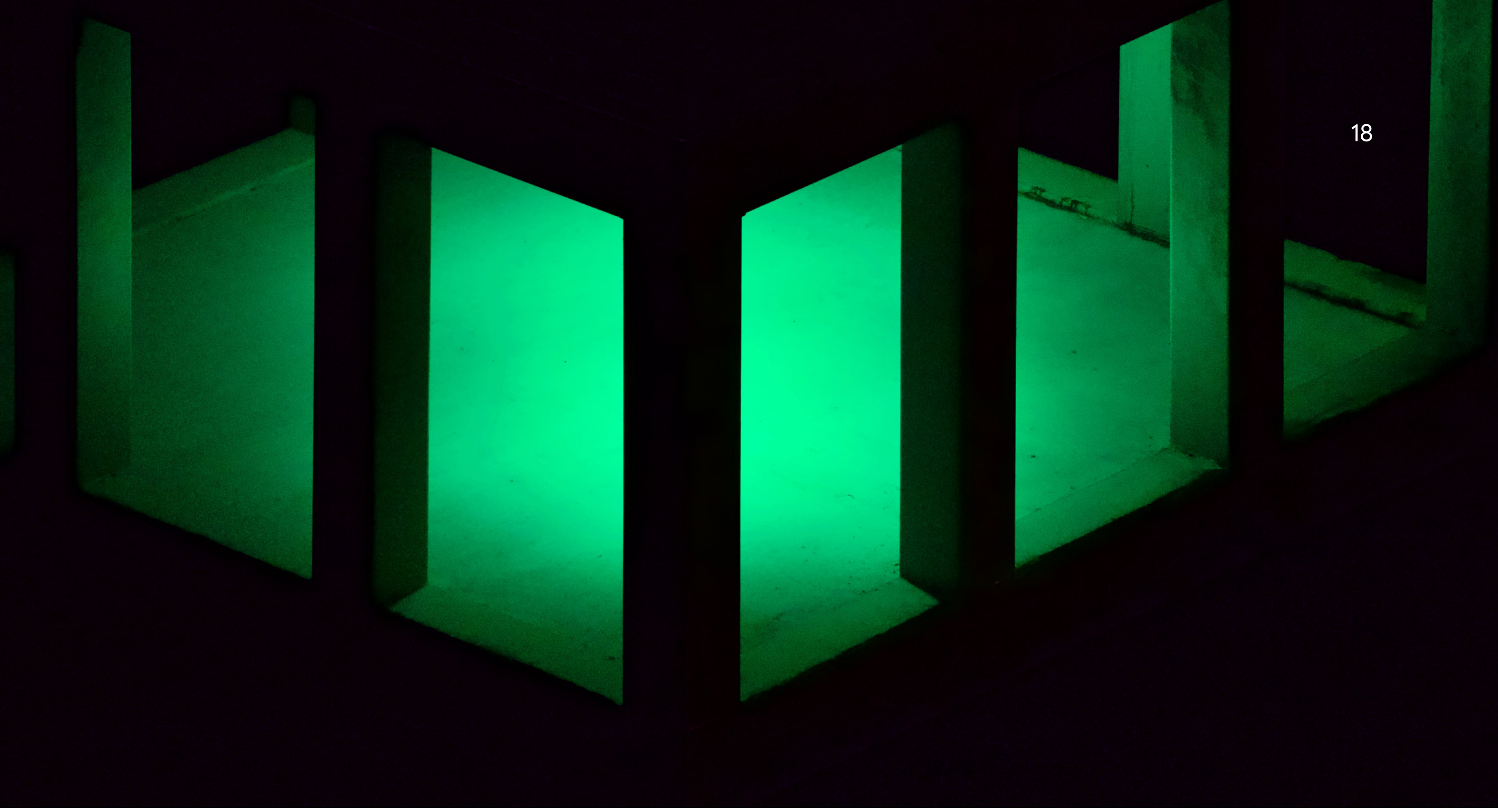
Ideally, your data governance requirements don't interfere with properly credentialed developers, IT Operators and SREs finding logs as quickly as possible, no matter where they are stored. In Cloud Logging, the Logs Explorer and Cloud Logging APIs support querying logs from multiple log buckets, multiple regions or even multiple projects together. This means that developers and IT operators can mostly ignore your setup and we'll keep displaying the logs right in the context of the Google Cloud Console, providing they have the appropriate access.

Grant users only access to logs they need for their job.

With Cloud Logging, queries are processed in the same location as the log buckets being queried and then aggregated in the region the query was received to return the results. This sensitivity to regional requirements helps you remain compliant.

As Sébastien De Nef, Site Reliability Engineer at ADEO Services explains, "Logs are crucial for helping technical teams understand what's happening in a software. Storing them in the same back end allows us to plug multiple data visualization tools according to who is consuming. We don't have to look at disparate systems to find the origin of the problem, so we can fix it faster."

In addition to addressing data residency requirements, you can use Cloud Logging and Google Cloud Identity and Access Management (IAM) to meet compliance requirements around who has access to sensitive data in logs:



Log buckets can control access to logs. Sensitive data can be written to a special bucket with limited access, for example. This is much easier to control and reason about when using structured logs. There are two other options for controlling access within a log bucket:

- Log views control which log entries a user can view within a single log bucket, e.g. Alice can only access logs from App Engine in a log bucket.

Use log views to subdivide access within a log bucket if your users can divide access to logs by source project, resource type, or log name. For other delineations, create different log buckets.

- Field-level access controls prevent users from viewing sensitive fields of a log entry within a bucket, e.g. users cannot view sensitive field protoPayload.resource unless explicitly granted access.

Use field level access control to deny access to sensitive fields by default.

Conclusion

Logs provide valuable data for troubleshooting, business insight, governance and security. The best practices discussed in the context of the four-stage logs data lifecycle can help you create a useful and compliant logs system. While this paper discusses the strategy around your logs collection and analysis, our companion whitepaper, Logs data: A step by step guide for overcoming common compliance challenges, provides step by step guidance for how to implement this strategy in Cloud Logging.vWe encourage you to download that guide when you are ready to scope the updates you may plan for your system.

If you would like to provide feedback or have questions about this guide or other Cloud Logging topics, please join the discussion group.

Appendix

At a glance: best practices for compliance across the logging lifecycle

Although different customers have different compliance needs, we've collected 14 best practices that are a great start for most customers in addressing their compliance requirements.

Lifecycle stage	Best practice
Log generation & collection	Encourage developers to use structured logs where possible. This allows you to more easily analyze your logs and promotes better logging hygiene, reducing the chance of logging sensitive information such as passwords. It also makes it possible to restrict access to sensitive data in your logs.
	Use severity levels consistently within your application so that you can easily identify actionable log messages. This also makes it easier to control costs by downsampling less important logs.
	Within Google Cloud, most platform logs and audit logs are collected automatically, simplifying the compliance effort. A few high volume logs are off by default and do need to be manually enabled. Turn on data access audit logs across your Google Cloud organization and VPC flow logs for critical subnetworks.
	Install a logging agent on all VMs, whether on GCP, other clouds, or on prem and configure it to collect key logs.

Lifecycle stage	Best practice
Log processing & routing	When sending logs from outside of Google Cloud, use a regional endpoint, e.g. `europe-west2-logging.googleapis.com`, to guarantee that a log entry is received in a specific region.
	Use an aggregated log sink at the org or folder level to collect logs from all projects, especially for security use cases. It's especially important to centralize logs that are important for compliance (e.g. audit logs, PCI logs, etc.) in a logging project so that even if the projects generating the logs are deleted, the logs will be available.
	Automate the setup of log sinks and log buckets for new projects to comply with your requirements around where logs are stored.
Log storage and retention	Create a custom log bucket for the region in which you need to store your logs and redirect your log sinks to the new log bucket instead of the global `_Default` bucket. Automate this step for new projects. The `_Required` bucket only contains logs about system events, Google actions and administrative actions (Create, Update, Delete) on GCP APIs.
	Set up organization policies so that all new custom log buckets will be created in approved regions.
	Configure retention on a log bucket to match your compliance requirements.
	If you have compliance requirements to prove logs have not been modified or deleted, consider locking the log bucket. However, keep in mind that this cannot be undone should requirements change in the future or if unintended data is sent to this log bucket so this is best used with well structured logs that you control.
Log analysis and access	Grant users only access to logs they need for their job.
	Use log views to subdivide access within a log bucket if your users can divide access to logs by source project, resource type, or log name. For other delineations, create different log buckets.
	Use field level access control to deny access to sensitive fields by default.