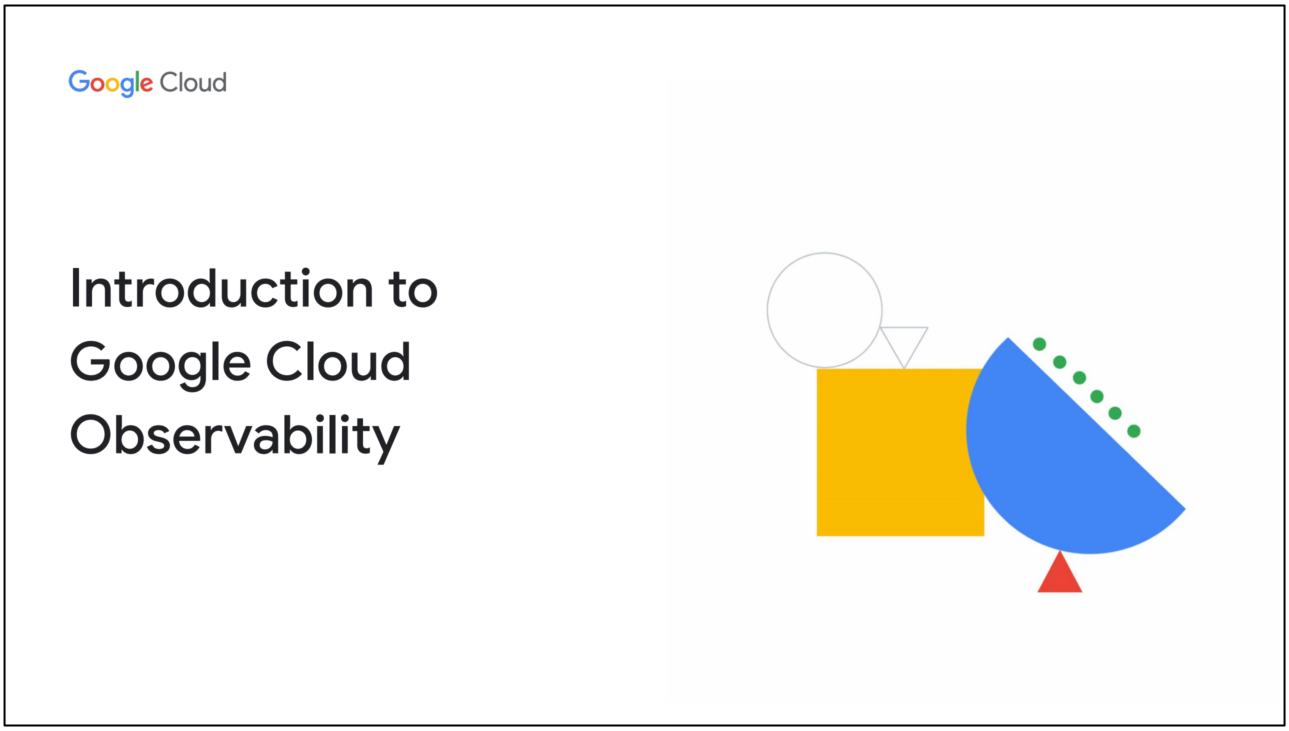


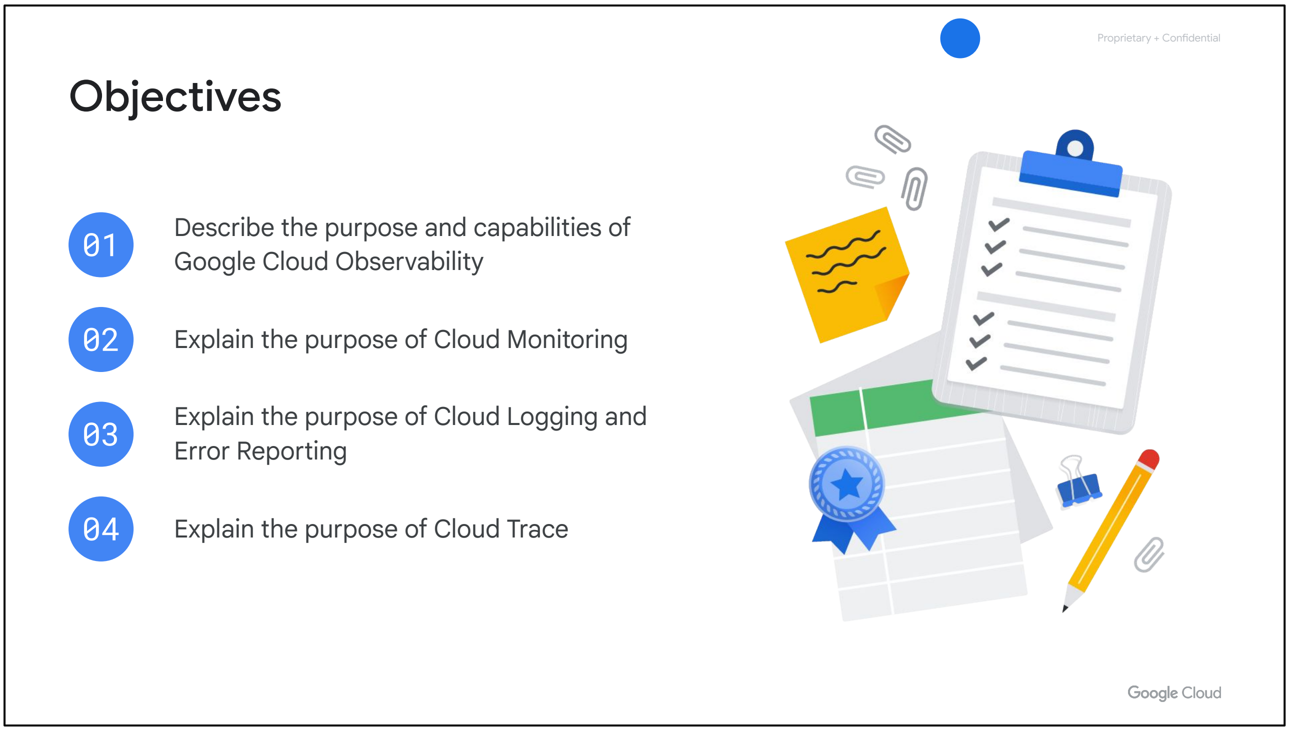
Google Cloud

Introduction to Google Cloud Observability

Let's start this module with a quick introduction to Google Cloud Observability.

Google Cloud Cloud Observability is a suite of products to monitor, diagnose and troubleshoot infrastructure, services, and applications at scale. It incorporates capabilities to let DevOps (development operations), SREs (site reliability engineering), or ITOps (information technology operations) users operate services in a manner similar to how Google SREs operate their own services. It offers integrated capabilities for monitoring, logging, and advanced observability services like Trace and Profiler.

Let us look at the objectives of the module in the next slide.



Objectives

01 Describe the purpose and capabilities of Google Cloud Observability

02 Explain the purpose of Cloud Monitoring

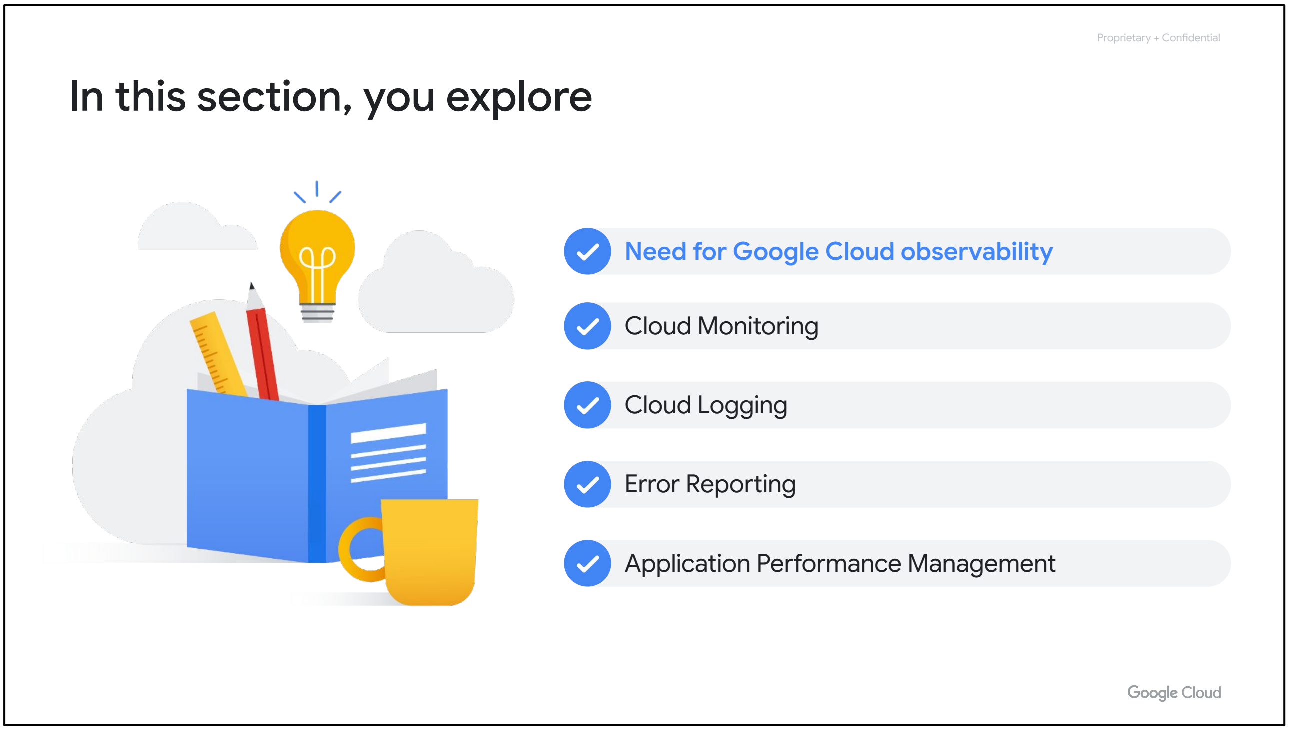
03 Explain the purpose of Cloud Logging and Error Reporting

04 Explain the purpose of Cloud Trace

Google Cloud

Google Cloud Observability consists of three broad categories, Cloud Logging, Cloud Monitoring and Application Performance Management. In this module, we will start with an overview of why we need these tools, and then we get to know both the operations and the Application Performance Management products.

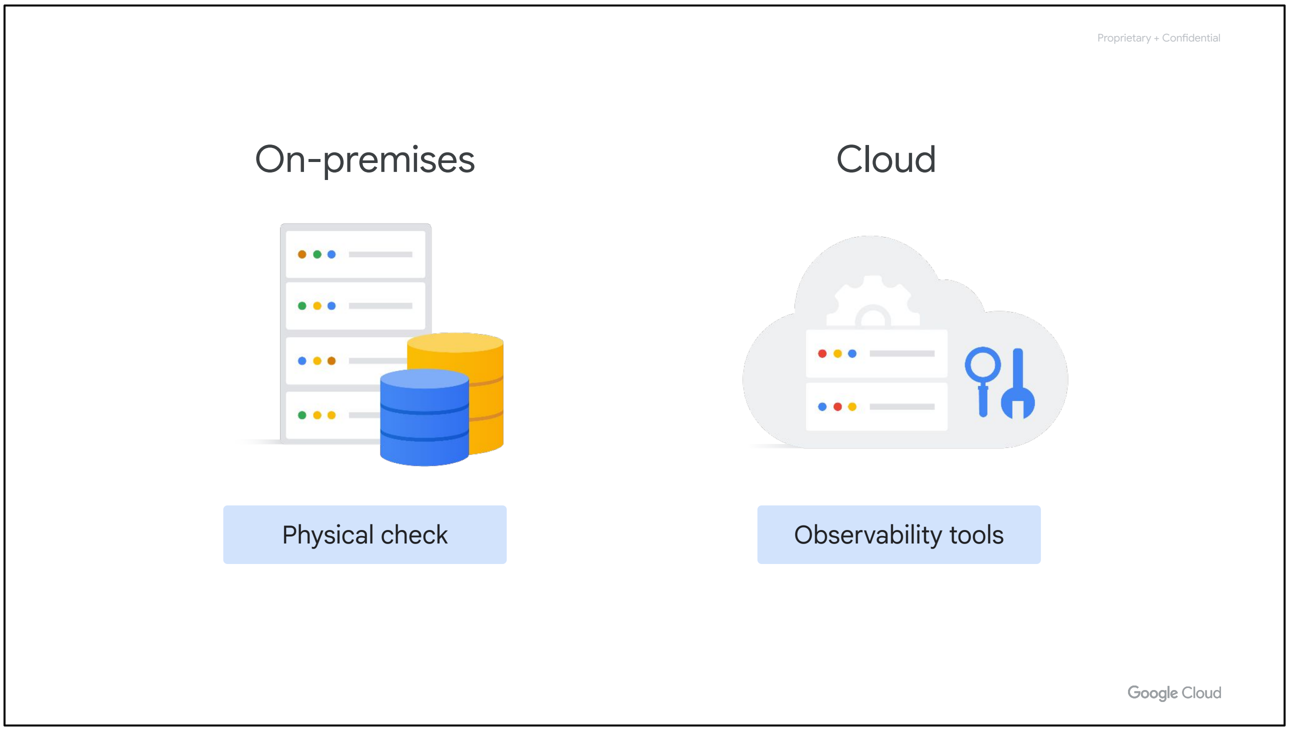
We will explore what the Google Cloud Observability architecture consists of to understand how the three pieces Cloud Logging, Cloud Monitoring and APM are connected.



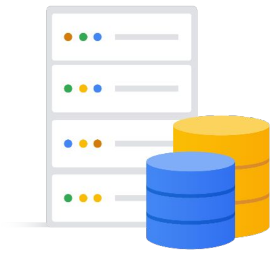
In this section, you explore

- ✓ **Need for Google Cloud observability**
- ✓ Cloud Monitoring
- ✓ Cloud Logging
- ✓ Error Reporting
- ✓ Application Performance Management

We start this section with an overview of why we need these tools, and then we'll spend a little time understanding the role of monitoring in product reliability. We will explore the significance of the four golden signals in measuring the system's performance and reliability. We then move on to explore the products in Google Cloud Observability.

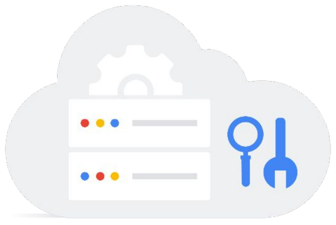


On-premises



Physical check

Cloud

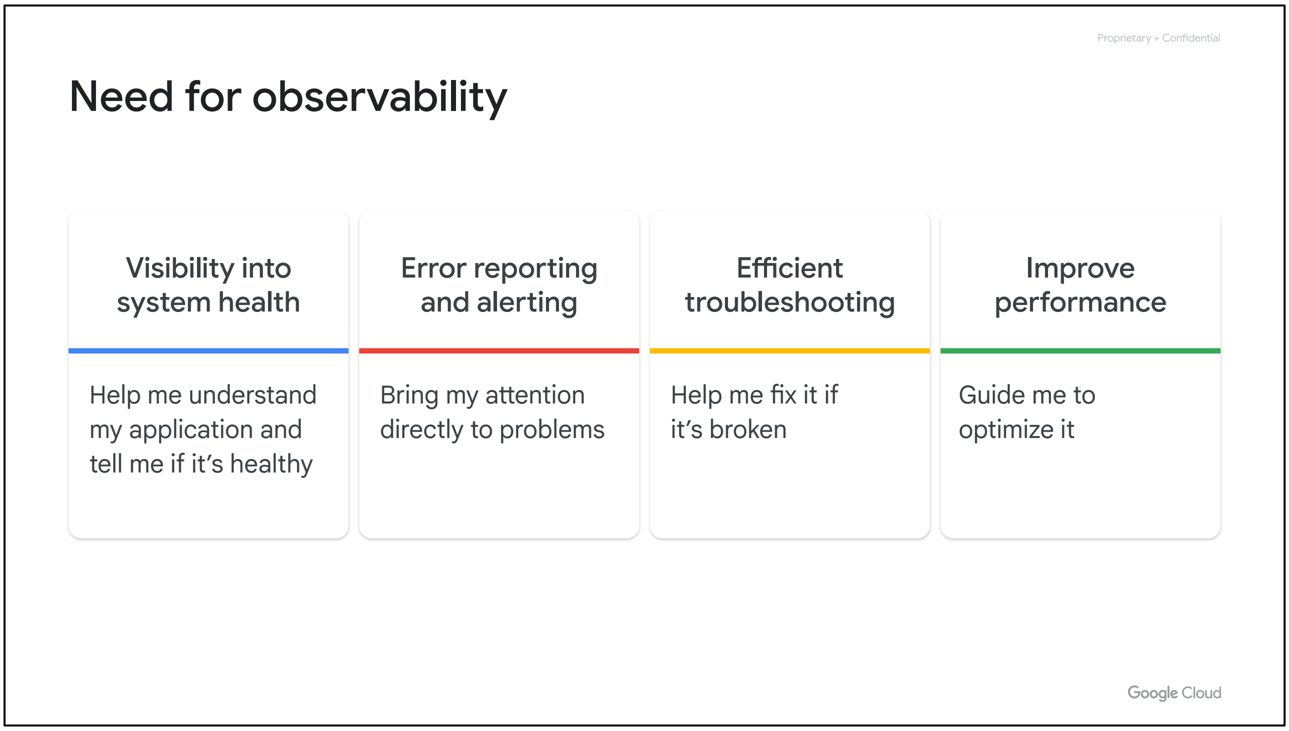


Observability tools

If you've ever worked with on-premises environments, you know that you can physically touch the servers. If an application becomes unresponsive, someone can physically determine why that happened.

In the cloud though, the servers aren't yours—they're Google's—and you can't physically inspect them. So the question becomes, how do you know what's happening with your server, or database, or application?

The answer is by using Google's integrated observability tools.

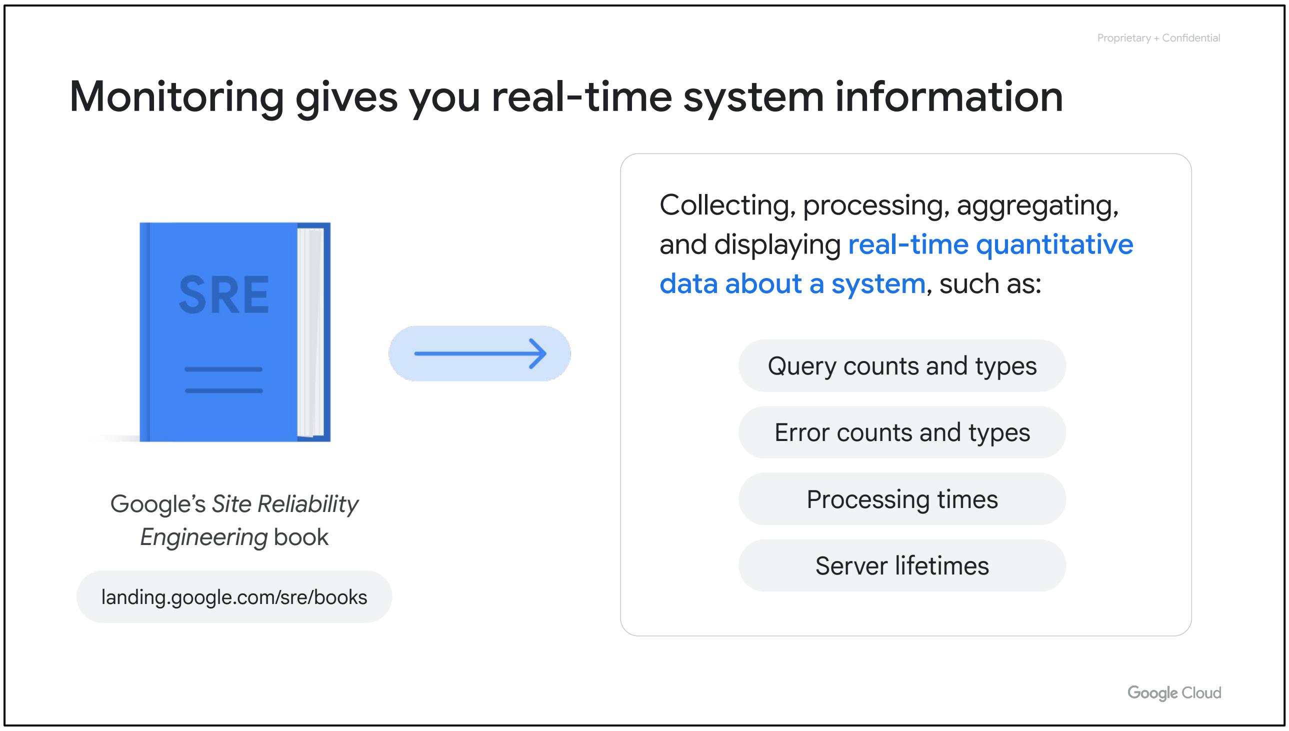


Need for observability

Visibility into system health	Error reporting and alerting	Efficient troubleshooting	Improve performance
Help me understand my application and tell me if it's healthy	Bring my attention directly to problems	Help me fix it if it's broken	Guide me to optimize it

The five distinct recurring user needs for observability are as follows:

1. Visibility into system health: Users want to understand what is happening with their application and system. They rely on a service that provides a clear mental model for how their application is working on Google Cloud. They need a report on the overall health of systems. The services should help answer questions such as "are my systems functioning?" or ""do my systems have sufficient resources available?"
2. Error reporting and alerting: Users want to monitor their service at a glance through healthy/unhealthy status icons or red/green indicators. Customers appreciate any proactive alerting, anomaly detection, or guidance on issues. Ideally, they want to avoid connecting the dots themselves.
3. Efficient troubleshooting: Users don't want multiple tabs open. They need a system that can proactively correlate relevant signals and make it easy to search across different data sources, like logs and metrics. If possible, the service needs to be opinionated about the potential cause of the issue and recommend a meaningful direction for the customer to start their investigation. It should allow users to immediately act on what they discover. For instance, a metric indicating insufficient quota should be accompanied by a button to increase quota.
4. Improve performance: Users need a service that can perform retrospective analysis. Generally, help them plan intelligently by analyzing trends and understand how changes in the system affect its performance.



Monitoring gives you real-time system information

Google's *Site Reliability Engineering* book

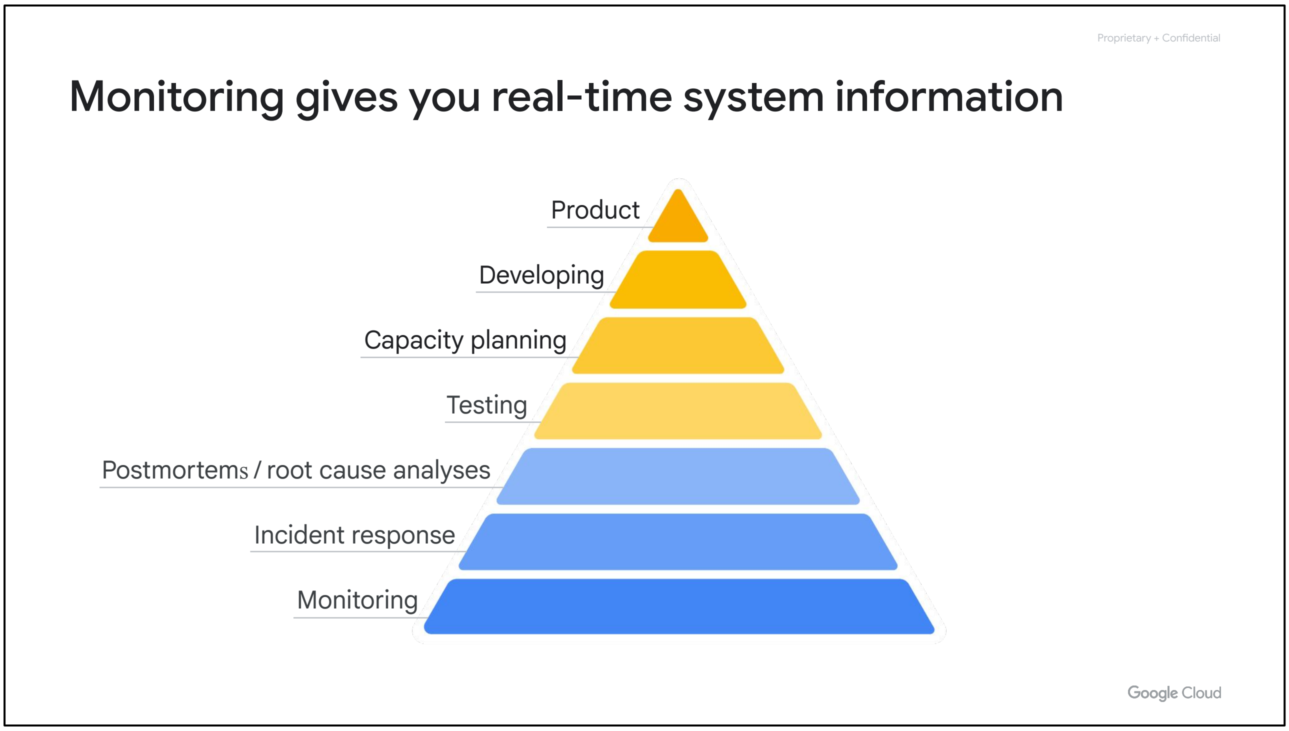
landing.google.com/sre/books

Collecting, processing, aggregating, and displaying **real-time quantitative data about a system**, such as:

- Query counts and types
- Error counts and types
- Processing times
- Server lifetimes

In Google's *Site Reliability Engineering* book, which is available to read at landing.google.com/sre/books, monitoring is defined as:

> *"Collecting, processing, aggregating, and displaying real-time quantitative data about a system, such as query counts and types, error counts and types, processing times, and server lifetimes."*



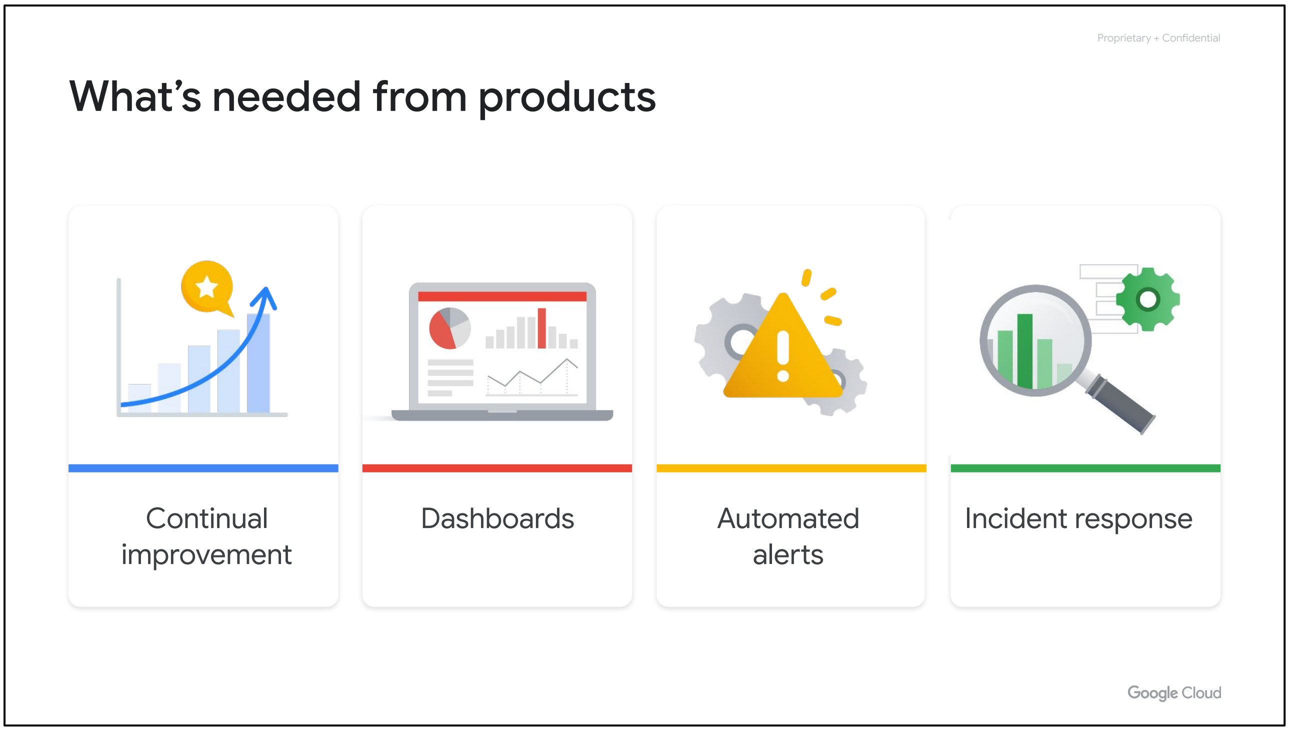
Monitoring gives you real-time system information

An application client normally only sees the public side of a **product**, and as a result, developers and business stakeholders both tend to think that the most crucial way to make the client happy is by spending the most time and effort on developing that part of the product.

However, to be truly reliable, even the very best products still must be deployed into environments with enough **capacity** to handle the anticipated client load.

Great products also need thorough **testing**, preferably automated testing, and a refined continuous integration/continuous development (CI/CD) release pipeline.

Postmortems and **root cause analyses** are the DevOps team's way of letting the client know why an **incident** happened and why it is unlikely to happen again. In this context we are discussing a system or software failure, but the term “incident” can also be used to describe a breach of security. Transparency here is key to building trust.



What’s needed from products

- Continual improvement
- Dashboards
- Automated alerts
- Incident response

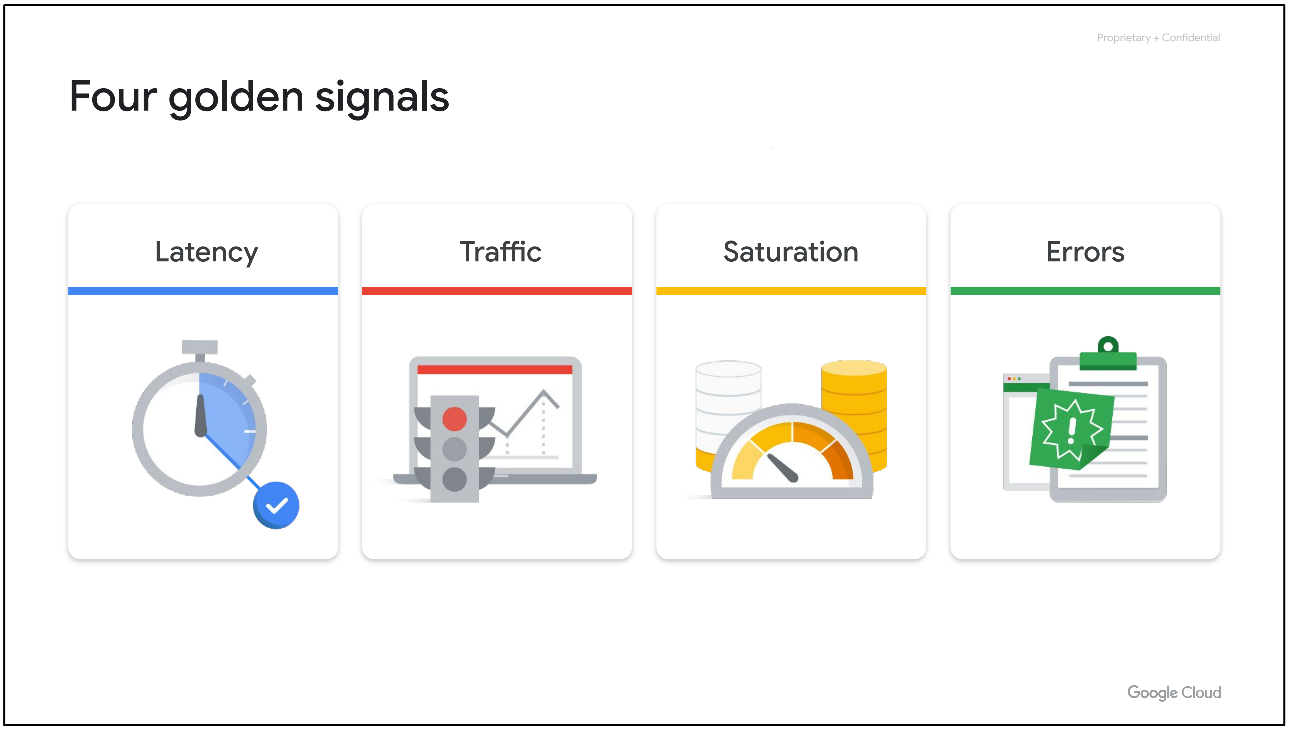
We need our products to improve continually, and we need we need monitoring data to ensure that happens.

We need dashboards to provide business intelligence so our DevOps personnel have the data they need to do their jobs.

We need automated alerts because humans tend to look at things only when there's something important to look at. An even better option is to construct automated systems to handle as many alerts as possible so humans only have to look at the most critical issues.

Typically, there's some triggering event: a system outage, data loss, a monitoring failure, or some form of manual intervention. The trigger leads to a response by both automated systems and DevOps personnel.

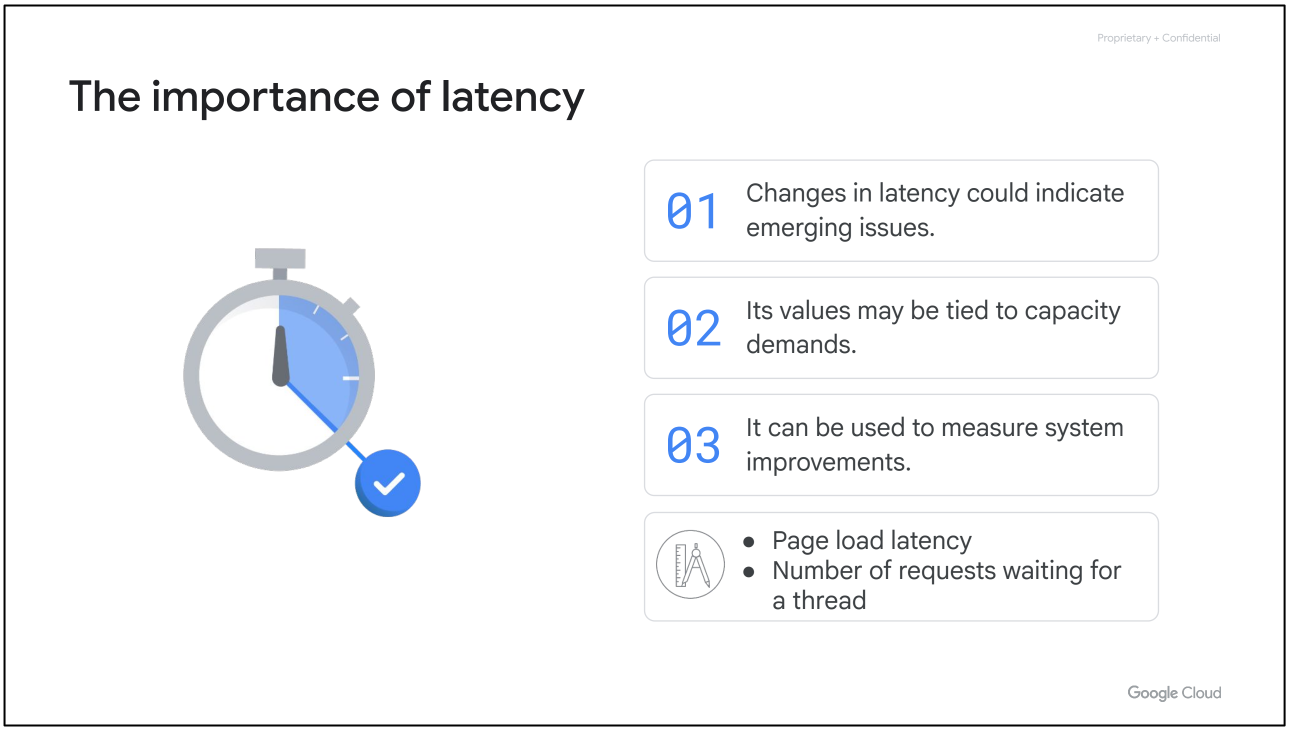
Many times the response starts by examining signal data that comes in through monitoring. The impact of the issue is evaluated and escalated when needed, and an initial response is formulated. Throughout, good SREs will strive to keep the customer informed and respond when appropriate.



Four golden signals

Latency
Traffic
Saturation
Errors

There are “four golden signals” that measure a system’s performance and reliability. They are **latency**, **traffic**, **saturation**, and **errors**.



The importance of latency

01 Changes in latency could indicate emerging issues.

02 Its values may be tied to capacity demands.

03 It can be used to measure system improvements.

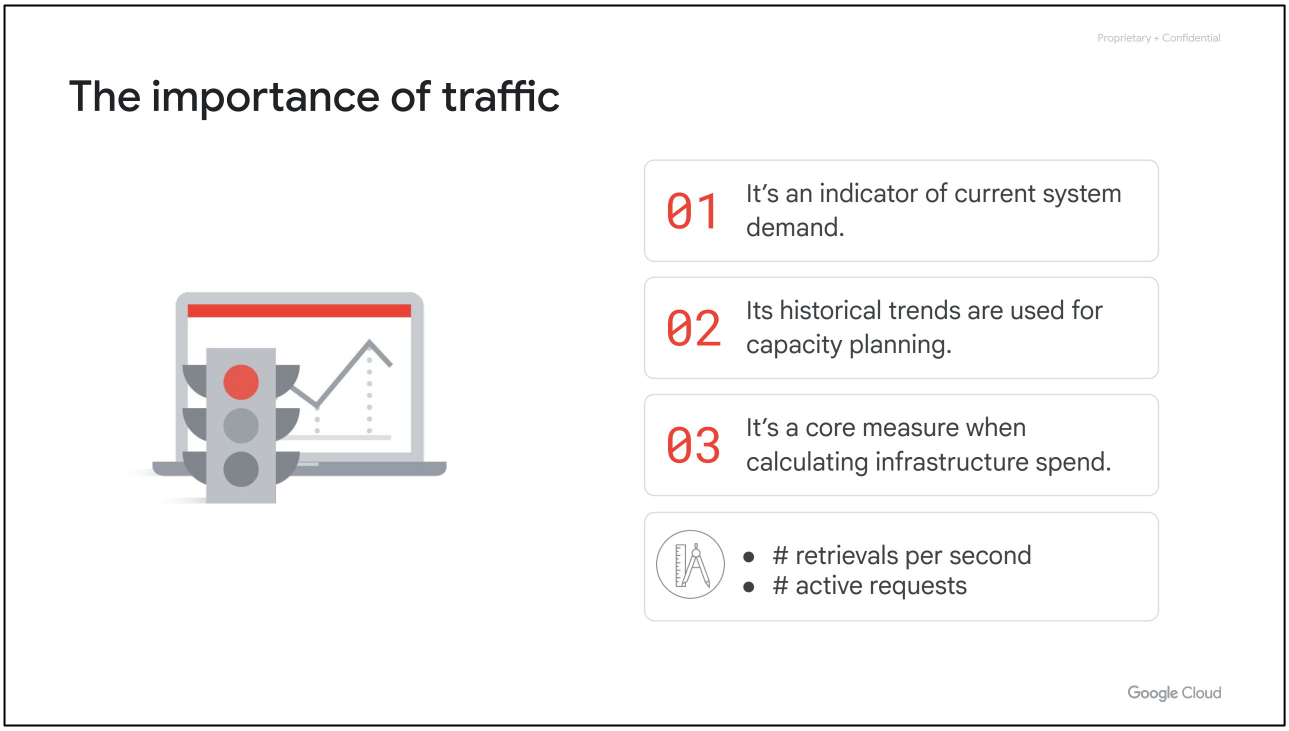
- Page load latency
- Number of requests waiting for a thread

Latency measures how long it takes a particular part of a system to return a result.

Latency is important because:
1. It directly affects the user experience.
2. Changes in latency could indicate emerging issues.
3. Its values may be tied to capacity demands.
4. It can be used to measure system improvements.

But how is it measured? Sample latency metrics include:

- Page load latency
- Number of requests waiting for a thread
- Query duration
- Service response time
- Transaction duration
- Time to first response
- Time to complete data return



Proprietary + Confidential

The importance of traffic

01 It’s an indicator of current system demand.

02 Its historical trends are used for capacity planning.

03 It’s a core measure when calculating infrastructure spend.

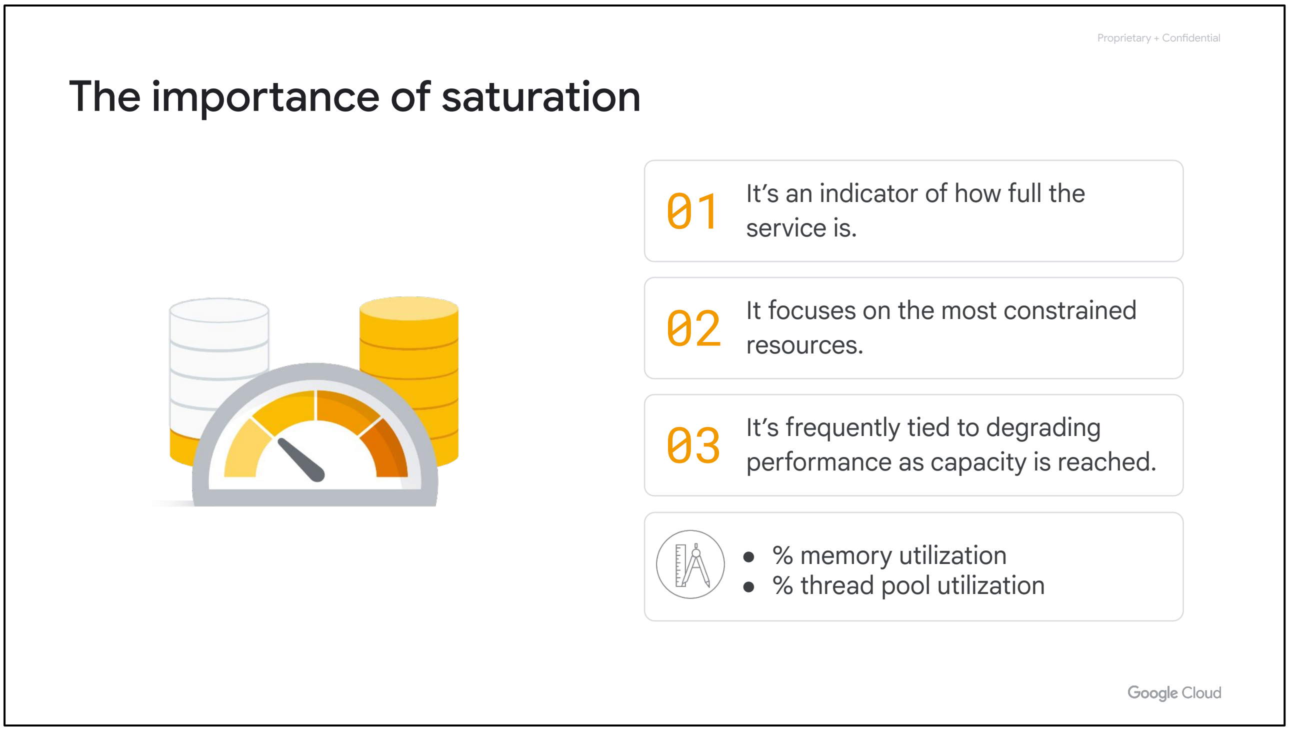
- # retrievals per second
- # active requests

Google Cloud

The next signal is **traffic**, which measures how many requests are reaching your system.

Traffic is important because:
1. It’s an indicator of current system demand.
2. Its historical trends are used for capacity planning.
3. It’s a core measure when calculating infrastructure spend.

Sample traffic metrics include:
- # HTTP requests per second
- # requests for static vs. dynamic content
- Network I/O
- # concurrent sessions
- # transactions per second
- # retrievals per second
- # active requests
- # write ops
- # read ops
- And # active connections



Proprietary + Confidential

The importance of saturation

01 It's an indicator of how full the service is.

02 It focuses on the most constrained resources.

03 It's frequently tied to degrading performance as capacity is reached.

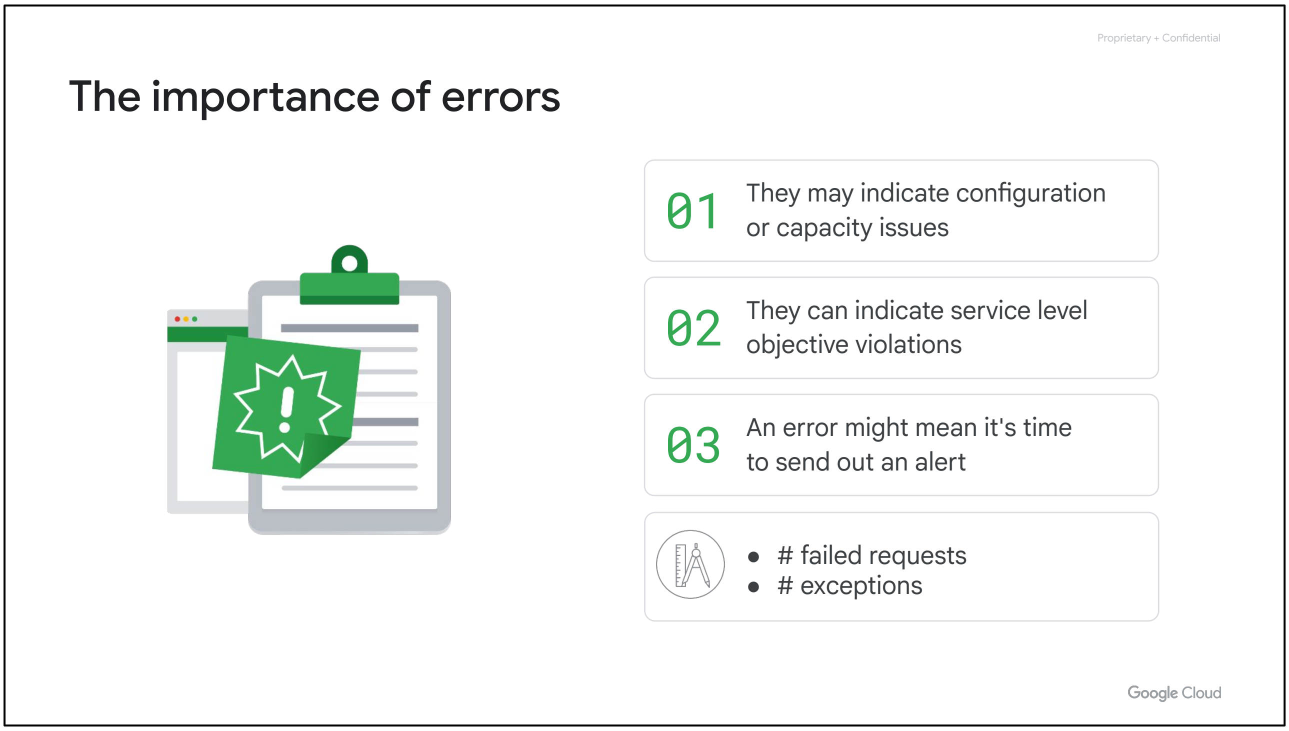
- % memory utilization
- % thread pool utilization

Google Cloud

The third signal is **saturation**, which measures how close to capacity a service is. It's important to note, though, that capacity is often a subjective measure, that depends on the underlying service or application.

Saturation is important because:
1. It's an indicator of how full the service is.
2. It focuses on the most constrained resources.
3. It's frequently tied to degrading performance as capacity is reached.

Sample capacity metrics include:
- % memory utilization
- % thread pool utilization
- % cache utilization
- % disk utilization
- % CPU utilization
- Disk quota
- Memory quota
- # of available connections
- And # of users on the system



The importance of errors

01 They may indicate configuration or capacity issues

02 They can indicate service level objective violations

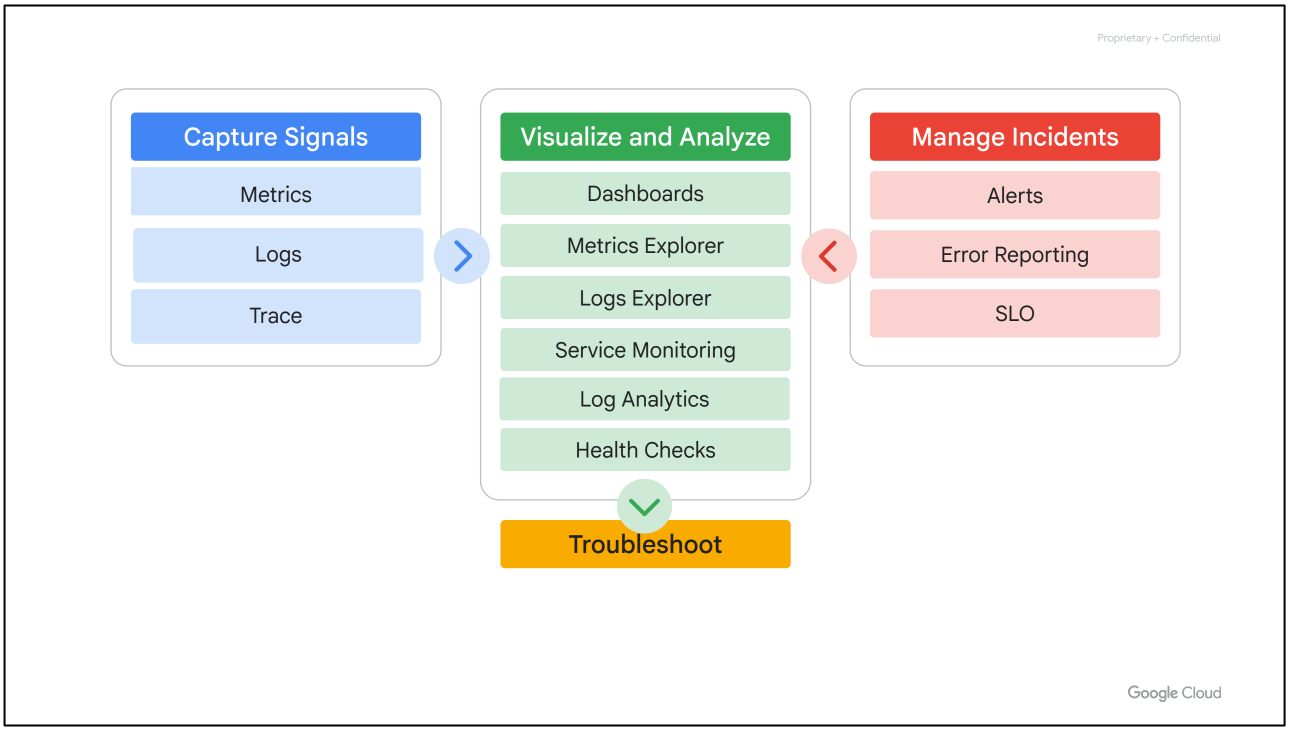
03 An error might mean it's time to send out an alert

- # failed requests
- # exceptions

The fourth signal is **errors**, which are events that measure system failures or other issues. Errors are often raised when a flaw, failure, or fault in a computer program or system causes it to produce incorrect or unexpected results, or behave in unintended ways.

Errors might indicate:
1. Configuration or capacity issues
2. Service level objective violations
3. That it's time to emit an alert

Sample error metrics include:
- Wrong answers or incorrect content
- # 400/500 HTTP codes
- # failed requests
- # exceptions
- # stack traces
- Servers that fail liveness checks
- And # dropped connections



Now, let's return to the observability concept. Observability starts with signals, which are metric, logging, and trace data captured and integrated into Google products from the hardware layer up.

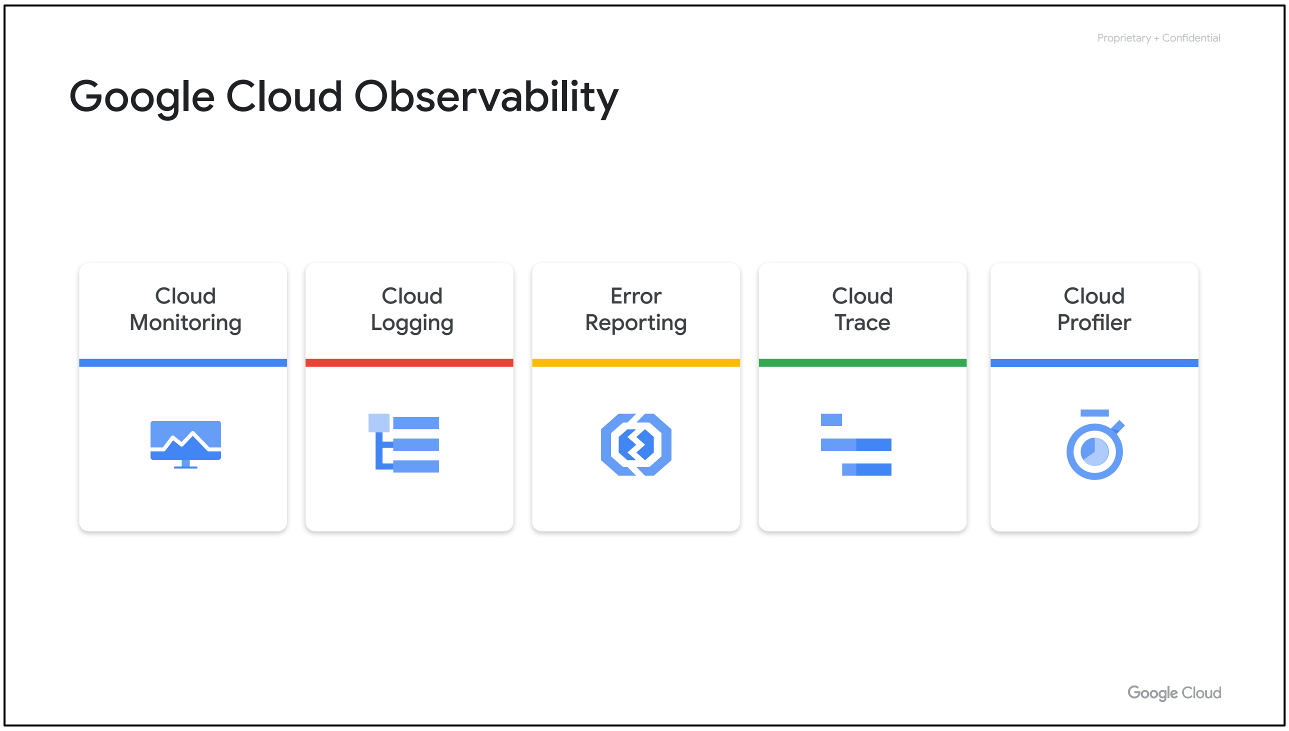
From those products:

- The signal data flows into Google Cloud Observability's tools where it can be visualized in dashboards and through the Metrics Explorer.
- Automated and custom logs can be dissected and analyzed in the Logs Explorer.
- Services can be monitored for compliance with service level objectives (SLOs), and error budgets can be tracked.
- Health checks can be used to check uptime and latency for external-facing sites and services.

When incidents occur:

- Signal data can generate automated alerts to code or, through various information channels, to key personnel.
- Error Reporting can help operations and developer teams spot, count, and analyze crashes in cloud-based services.
- The visualization and analysis tools can then help troubleshoot what's happening in Google Cloud.

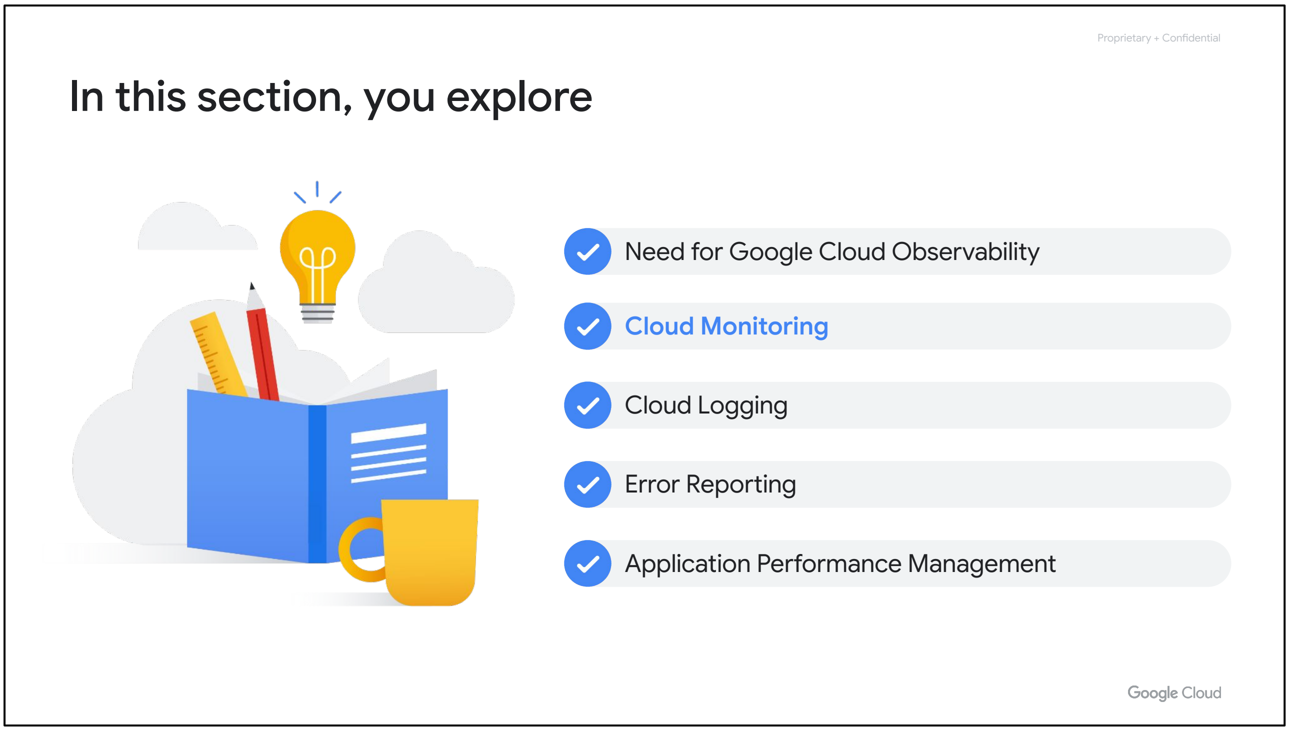
Ultimately, you won't miss that easy server access, because Google provides more precise insights into your Cloud install than you ever had on-premises.



Google Cloud Observability

Cloud Monitoring
Cloud Logging
Error Reporting
Cloud Trace
Cloud Profiler

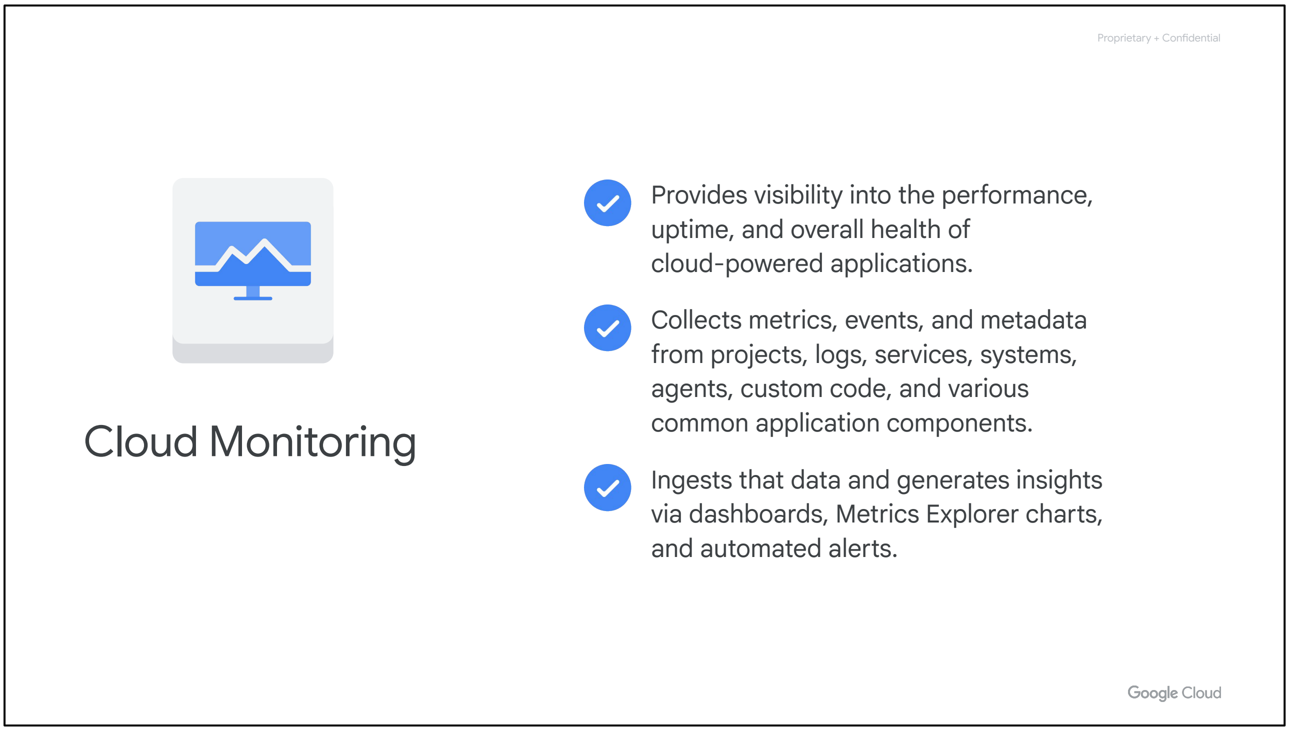
Let’s explore the products most applicable for those in operations roles that work with Cloud **Monitoring**, Cloud **Logging**, **Error Reporting**, **Cloud Trace** and **Cloud Profiler.**



In this section, you explore

- ✓ Need for Google Cloud Observability
- ✓ **Cloud Monitoring**
- ✓ Cloud Logging
- ✓ Error Reporting
- ✓ Application Performance Management

We defined general monitoring and its benefits in the previous section. Let us take a look at Cloud Monitoring features and benefits.

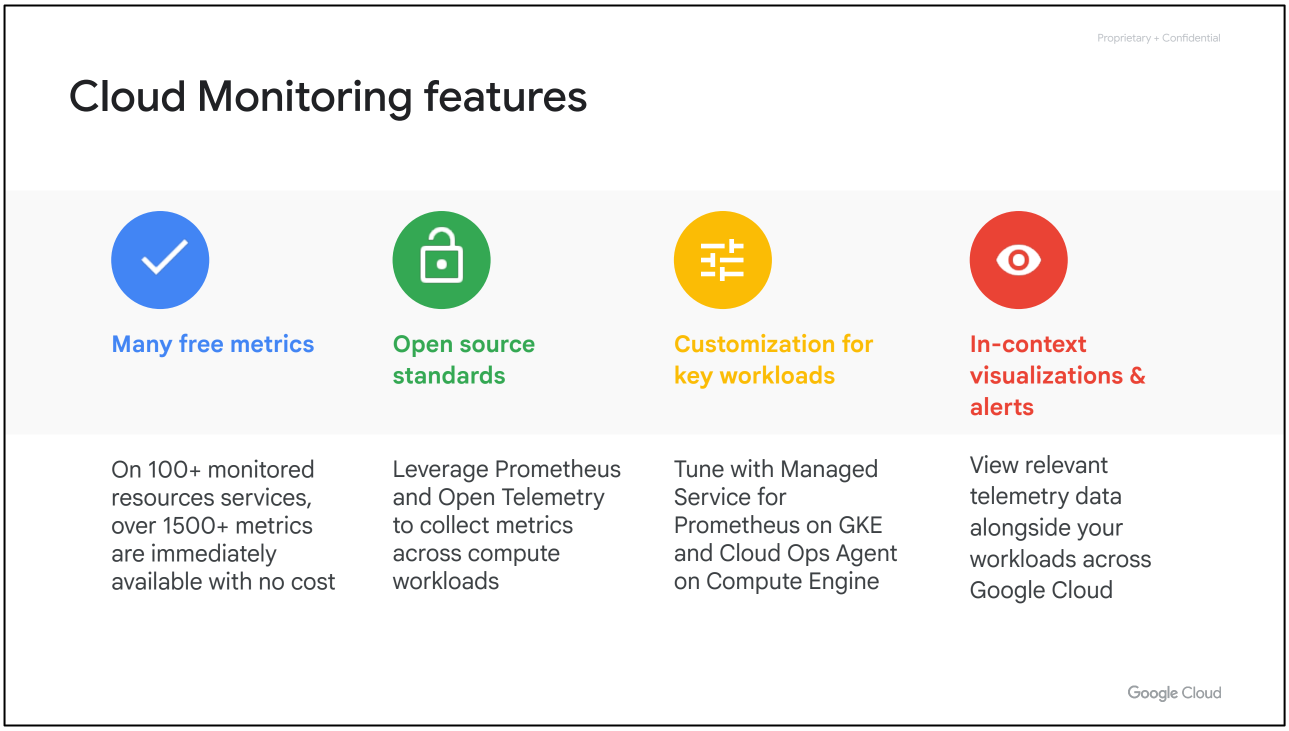


Cloud Monitoring

- ✓ Provides visibility into the performance, uptime, and overall health of cloud-powered applications.
- ✓ Collects metrics, events, and metadata from projects, logs, services, systems, agents, custom code, and various common application components.
- ✓ Ingests that data and generates insights via dashboards, Metrics Explorer charts, and automated alerts.

Cloud Monitoring provides visibility into the performance, uptime, and overall health of cloud-powered applications. It collects metrics, events, and metadata from projects, logs, services, systems, agents, custom code, and various common application components, including Cassandra, Nginx, Apache Web Server, Elasticsearch, and many others.

Monitoring ingests that data and generates insights via dashboards, Metrics Explorer charts, and automated alerts.


Proprietary + Confidential

Cloud Monitoring features

Many free metrics

On 100+ monitored resources services, over 1500+ metrics are immediately available with no cost

Open source standards

Leverage Prometheus and Open Telemetry to collect metrics across compute workloads

Customization for key workloads

Tune with Managed Service for Prometheus on GKE and Cloud Ops Agent on Compute Engine

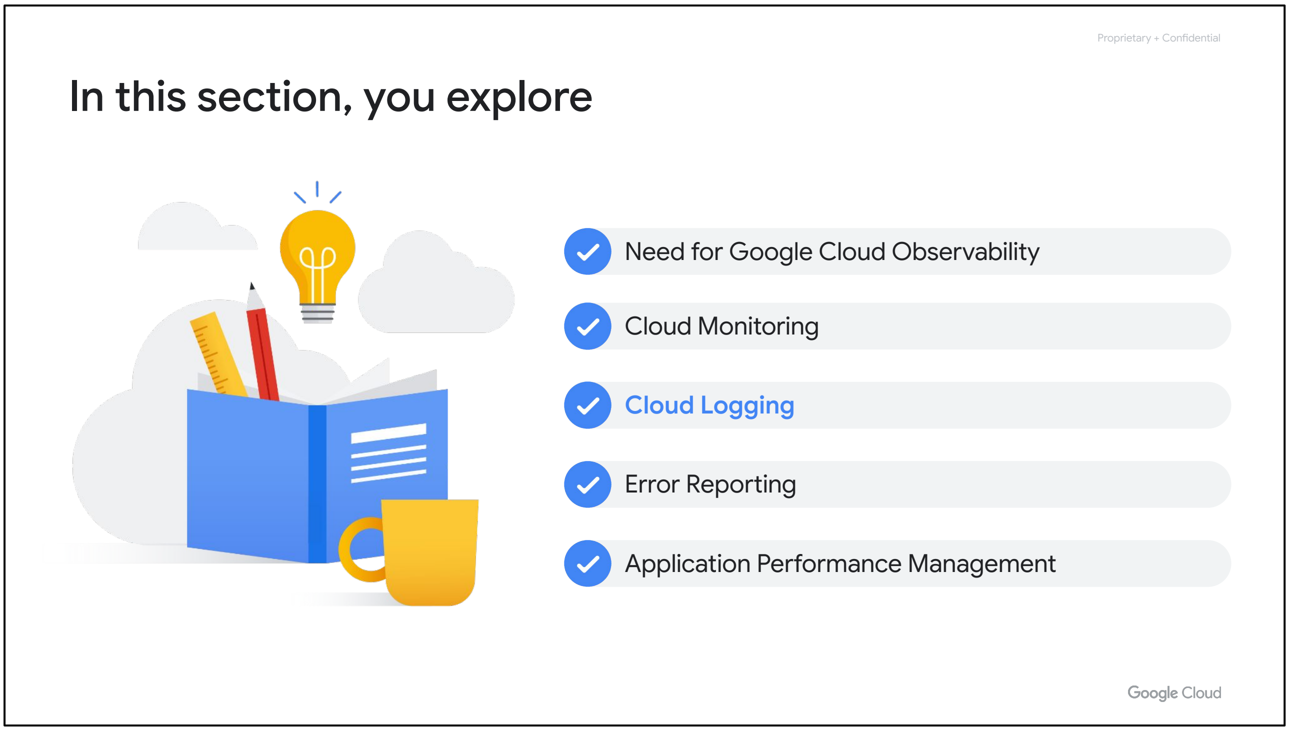
In-context visualizations & alerts

View relevant telemetry data alongside your workloads across Google Cloud

Google Cloud

Cloud Monitoring provides many advanced capabilities that helps address the monitoring challenges and these include:

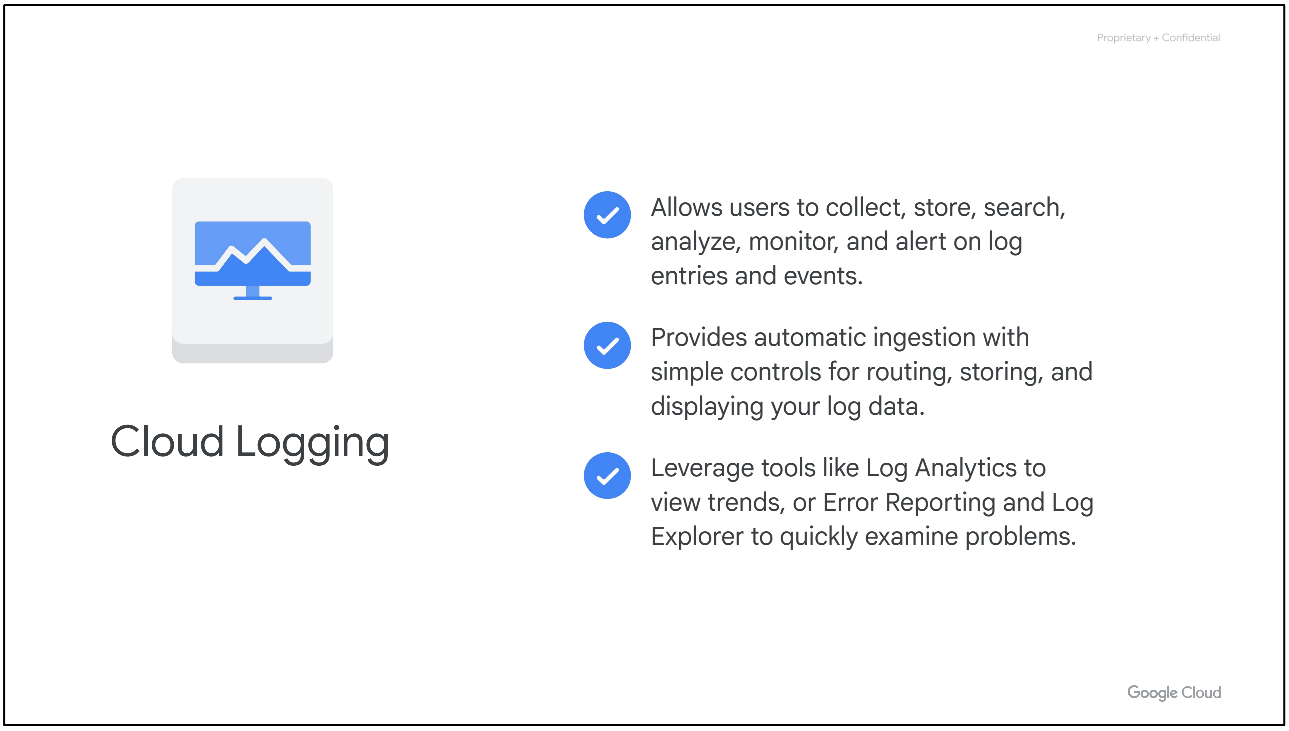
- **Many free metrics:** On 100+ monitored resources, over 1,500 metrics are immediately available with no cost. You can find out more about this at Google Cloud Observability pricing.
- **Open source standards**: Leverage Prometheus and Open Telemetry to collect metrics across compute workloads.
- **Customization for key workloads**: Cloud Monitoring offers custom visualization capabilities for GKE through Google Cloud Managed Service for Prometheus and for Compute Engine through Ops Agent.
- **In-context visualizations & alerts**: View relevant telemetry data alongside your workloads across Google Cloud.



In this section, you explore

- ✓ Need for Google Cloud Observability
- ✓ Cloud Monitoring
- ✓ **Cloud Logging**
- ✓ Error Reporting
- ✓ Application Performance Management

The next service we will explore is Cloud Logging.



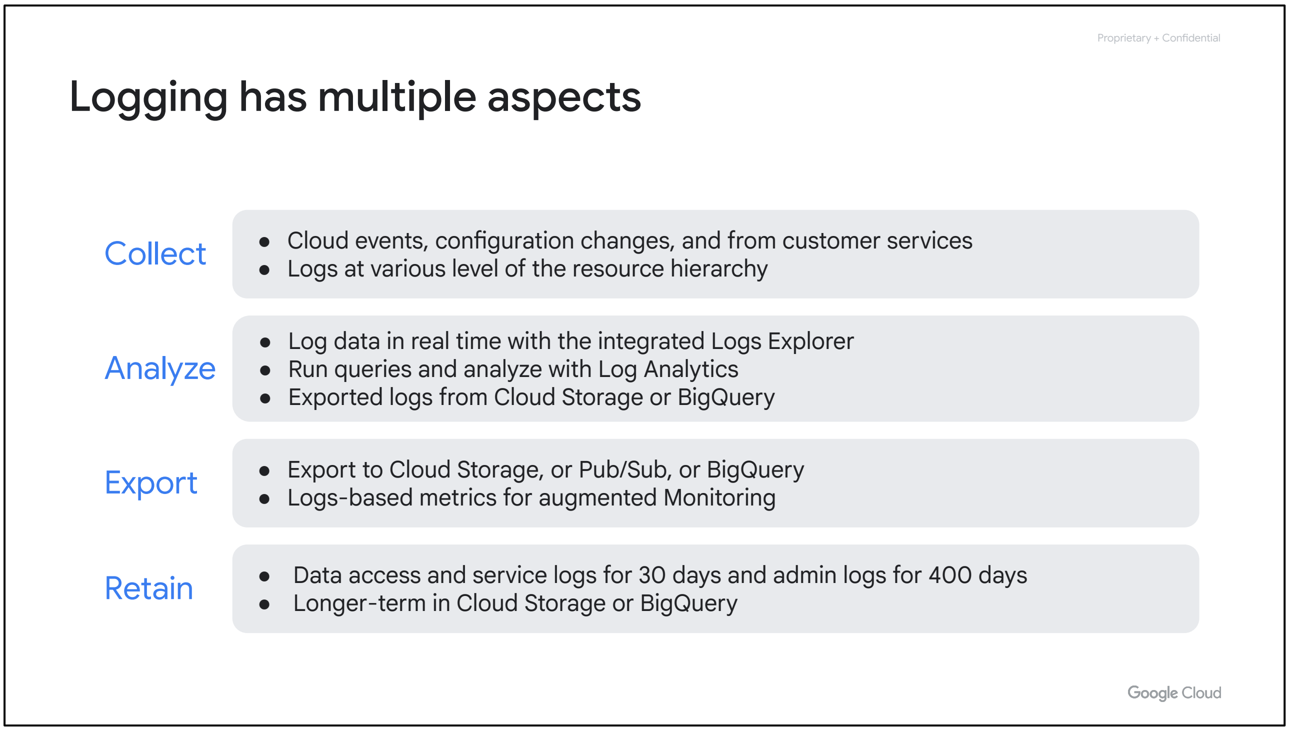
Cloud Logging

- ✓ Allows users to collect, store, search, analyze, monitor, and alert on log entries and events.
- ✓ Provides automatic ingestion with simple controls for routing, storing, and displaying your log data.
- ✓ Leverage tools like Log Analytics to view trends, or Error Reporting and Log Explorer to quickly examine problems.

Google's **Cloud Logging** allows users to collect, store, search, analyze, monitor, and alert on log entries and events. Automated logging is integrated into Google Cloud products like App Engine, Cloud Run, Compute Engine VMs running the logging agent, and GKE.

Cloud Logging also provides massive features that makes managing and exploring tons of logs easier. These include:

- **Automatic, easy log ingestion:** Immediate ingestion from Google Cloud services across your stack
- **Gain insight quickly:** Tools like Error Reporting, Log Explorer, and Log Analytics let you quickly focus from large sets of data
- **Customize routing & storage:** Route your logs to the region or service of your choice for additional compliance or business benefits
- **Compliance Insights:** Leverage audit and app logs for compliance patterns and issues



Logging has multiple aspects

Collect	• Cloud events, configuration changes, and from customer services • Logs at various level of the resource hierarchy
Analyze	• Log data in real time with the integrated Logs Explorer • Run queries and analyze with Log Analytics • Exported logs from Cloud Storage or BigQuery
Export	• Export to Cloud Storage, or Pub/Sub, or BigQuery • Logs-based metrics for augmented Monitoring
Retain	• Data access and service logs for 30 days and admin logs for 400 days • Longer-term in Cloud Storage or BigQuery

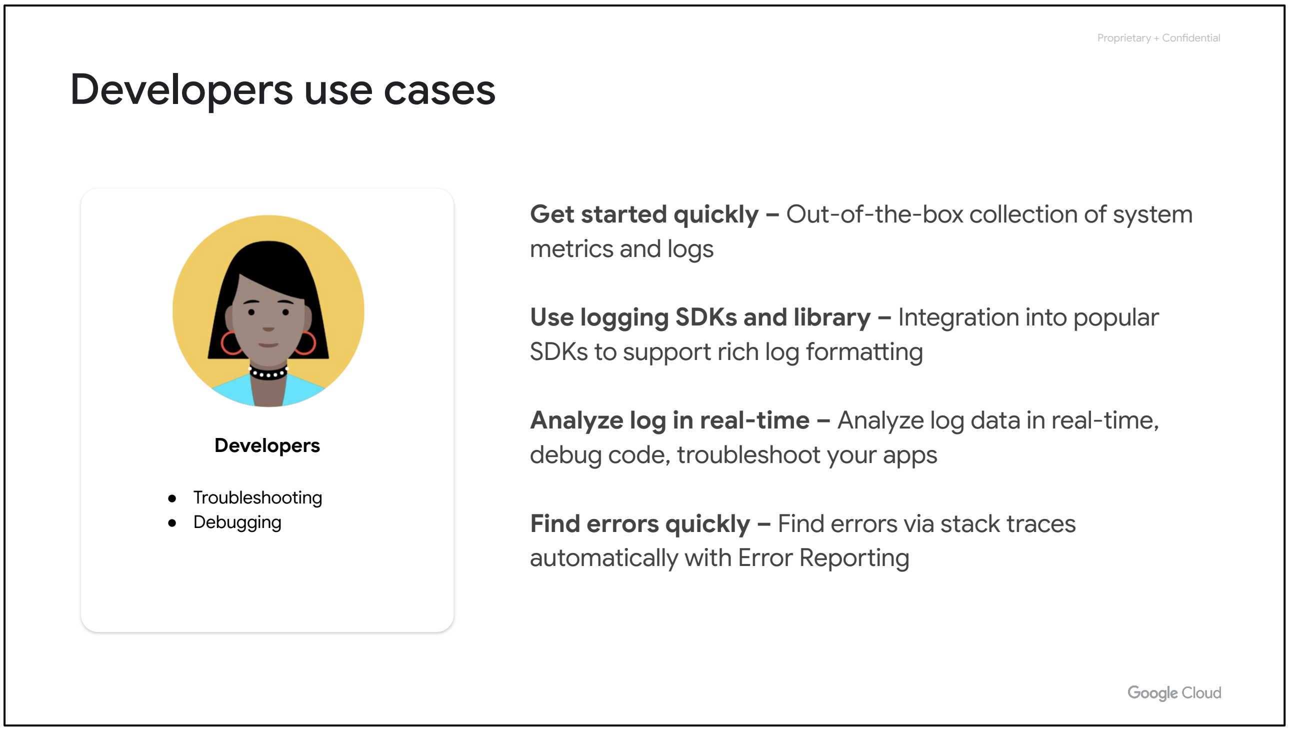
Google Cloud

Google's Cloud Logging allows users to collect, store, search, analyze, monitor, and alert on log entries and events. Automated logging is integrated into Google Cloud products like App Engine, Cloud Run, Compute Engine VMs that run the logging agent, and GKE. You can aggregate and centralize logs at a organizational level, project level and folder level based on your needs.

Most log analysis start with Google Cloud's integrated Logs Explorer. You can run queries and analyze log data with Log Analytics. Logging entries can also be exported to several destinations for alternative or further analysis.

Export log data as files to Google Cloud Storage, or as messages through Pub/Sub, or into BigQuery tables. Pub/Sub messages can be analyzed in near-real time using custom code or stream processing technologies like Dataflow. BigQuery allows analysts to examine logging data through SQL queries. And archived log files in Cloud Storage can be analyzed with several tools and techniques. Logs-based metrics may be created and integrated into Cloud Monitoring dashboards, alerts, and service SLOs.

Default log retention in Cloud Logging depends on the log type. Data access logs are retained by default for 30 days, but this is configurable up to a max of 3650 days. Admin logs are stored by default for 400 days. Export logs to Google Cloud Storage or BigQuery to extend retention.



Proprietary + Confidential

Developers use cases

Developers

- Troubleshooting
- Debugging

Get started quickly – Out-of-the-box collection of system metrics and logs

Use logging SDKs and library – Integration into popular SDKs to support rich log formatting

Analyze log in real-time – Analyze log data in real-time, debug code, troubleshoot your apps

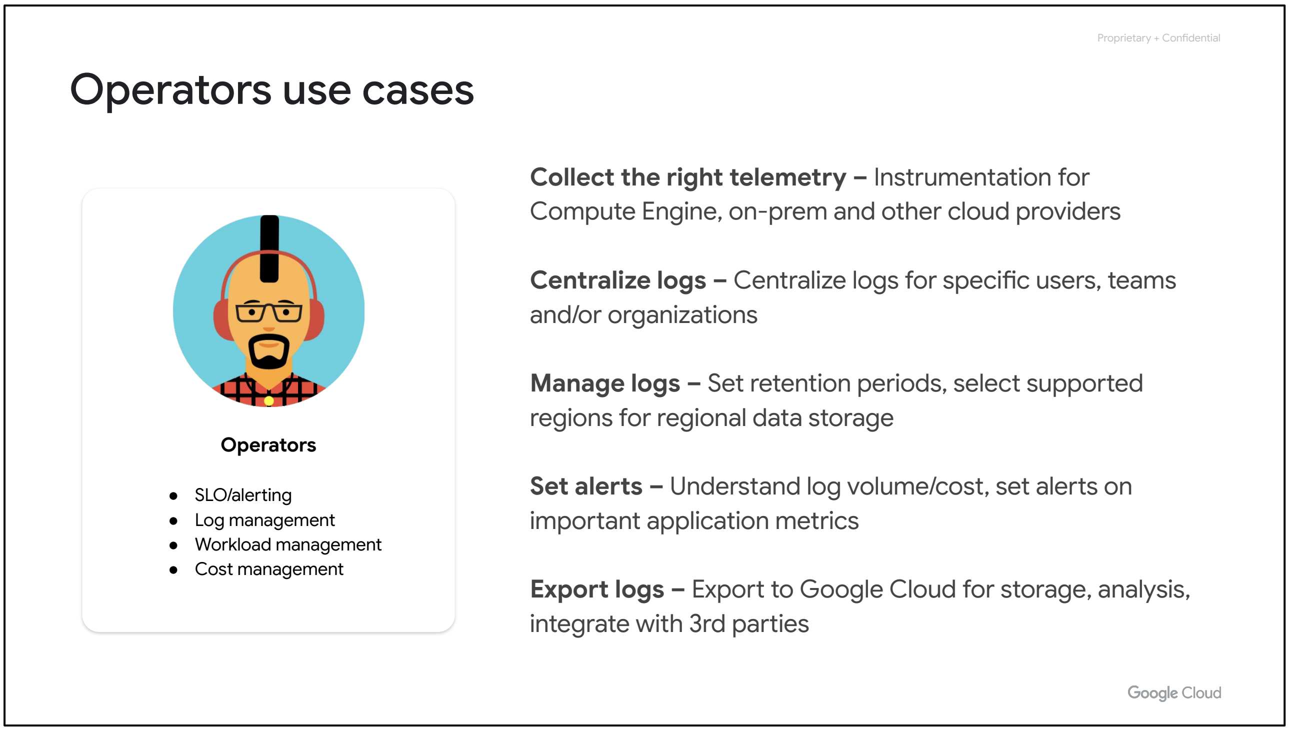
Find errors quickly – Find errors via stack traces automatically with Error Reporting

Google Cloud

A developer would love to get started quickly, thus we have out of the box collection of system metrics and logs. You also get support for most of the SDKs and languages available on the market.

It also allows you to do real time analysis, debugging and troubleshooting of your code.

Stack traces are automatically mapped to error types for convenient access and visibility



Operators use cases

Operators

- SLO/alerting
- Log management
- Workload management
- Cost management

Collect the right telemetry – Instrumentation for Compute Engine, on-prem and other cloud providers

Centralize logs – Centralize logs for specific users, teams and/or organizations

Manage logs – Set retention periods, select supported regions for regional data storage

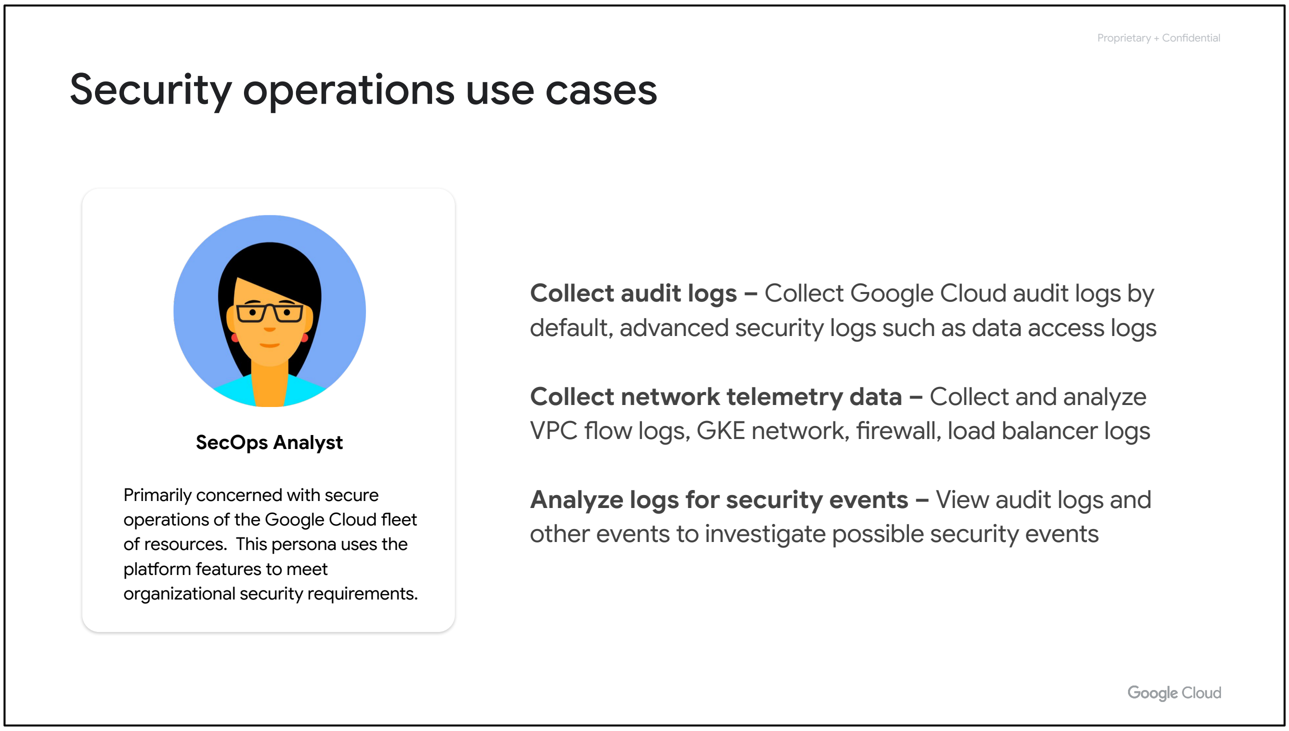
Set alerts – Understand log volume/cost, set alerts on important application metrics

Export logs – Export to Google Cloud for storage, analysis, integrate with 3rd parties

Google Cloud

Operators also take massive advantage of our offering. These include collecting telemetry that is not limited to Google Cloud, centralization of all the logs for users, teams and organizations. You are in control of retention periods and location of the logs. You can also understand log volume, cost and set alerts on important application metrics.

You can export logs for storage, analysis and also integrate with third-party services.



Security operations use cases

SecOps Analyst

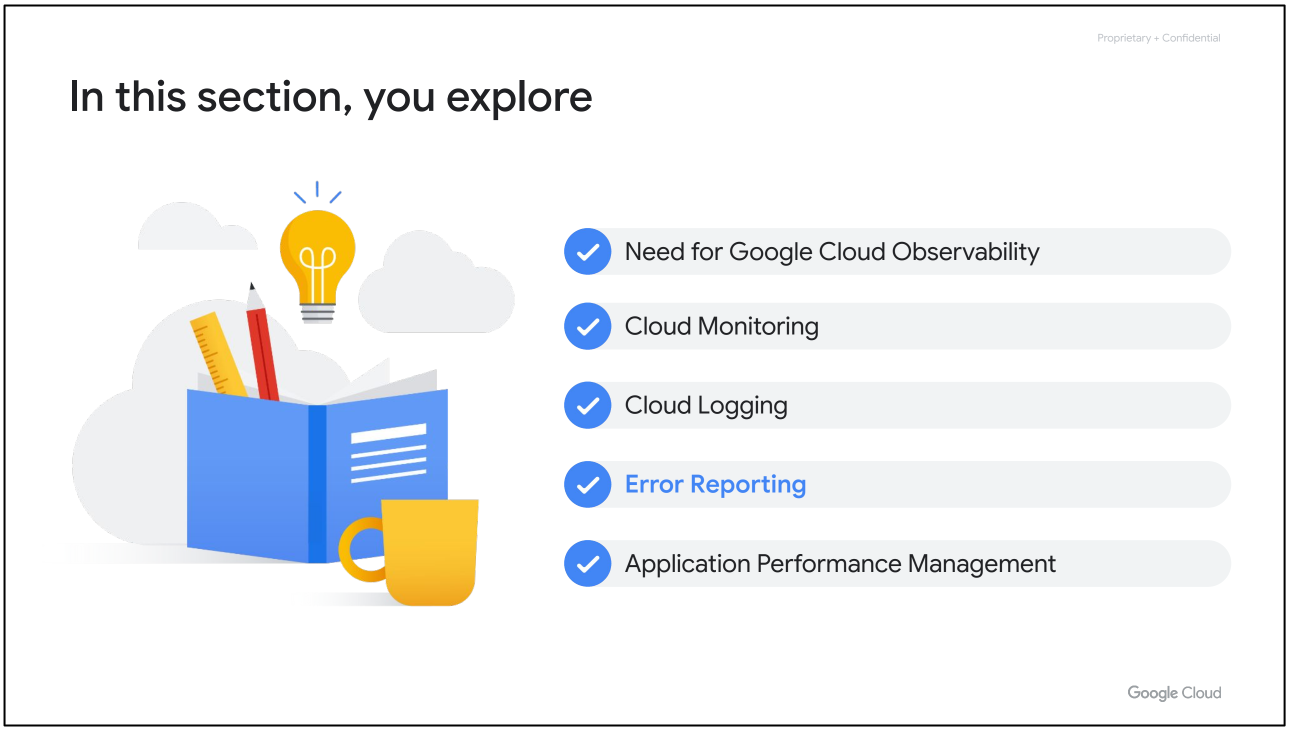
Primarily concerned with secure operations of the Google Cloud fleet of resources. This persona uses the platform features to meet organizational security requirements.

Collect audit logs – Collect Google Cloud audit logs by default, advanced security logs such as data access logs

Collect network telemetry data – Collect and analyze VPC flow logs, GKE network, firewall, load balancer logs

Analyze logs for security events – View audit logs and other events to investigate possible security events

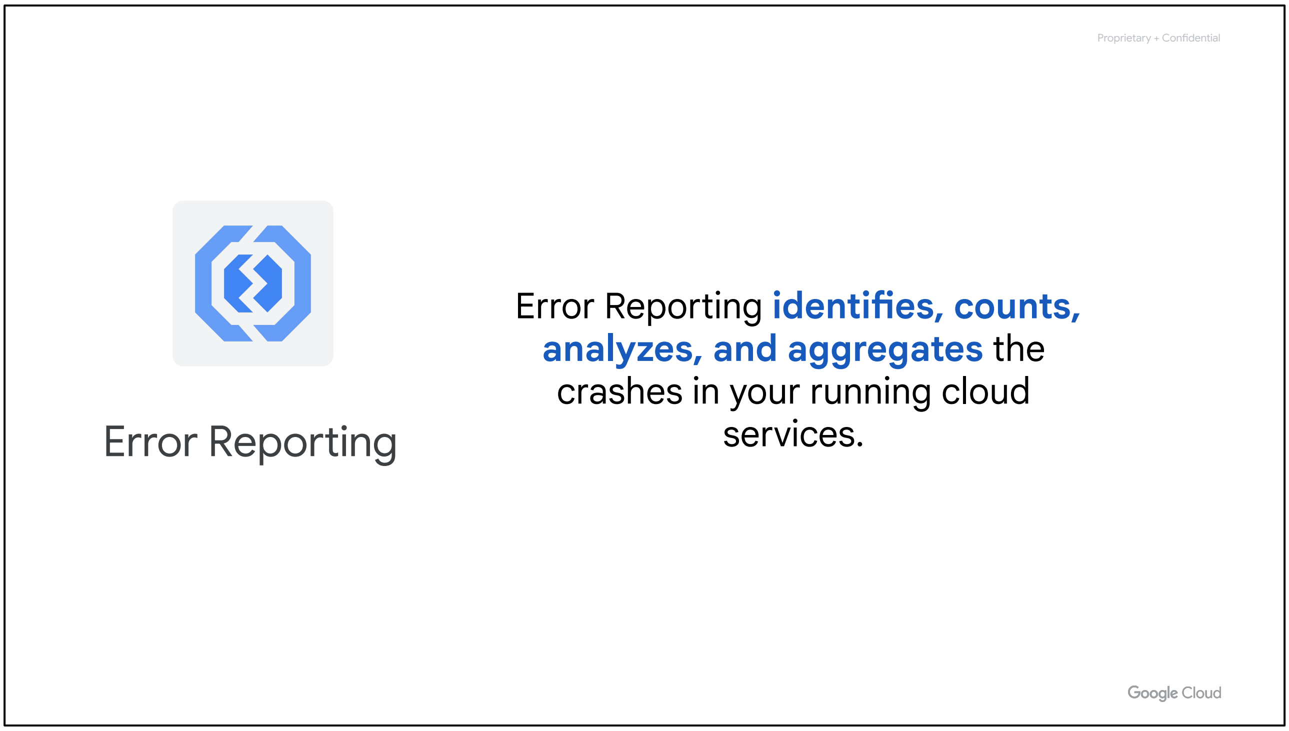
Lastly, security operations, or SecOps are in charge of ensuring that all access is authorized and that bad actors are not navigating your network. With audit logs, network telemetry and log analysis it can be achieved in a streamlined way.



In this section, you explore

- ✓ Need for Google Cloud Observability
- ✓ Cloud Monitoring
- ✓ Cloud Logging
- ✓ **Error Reporting**
- ✓ Application Performance Management

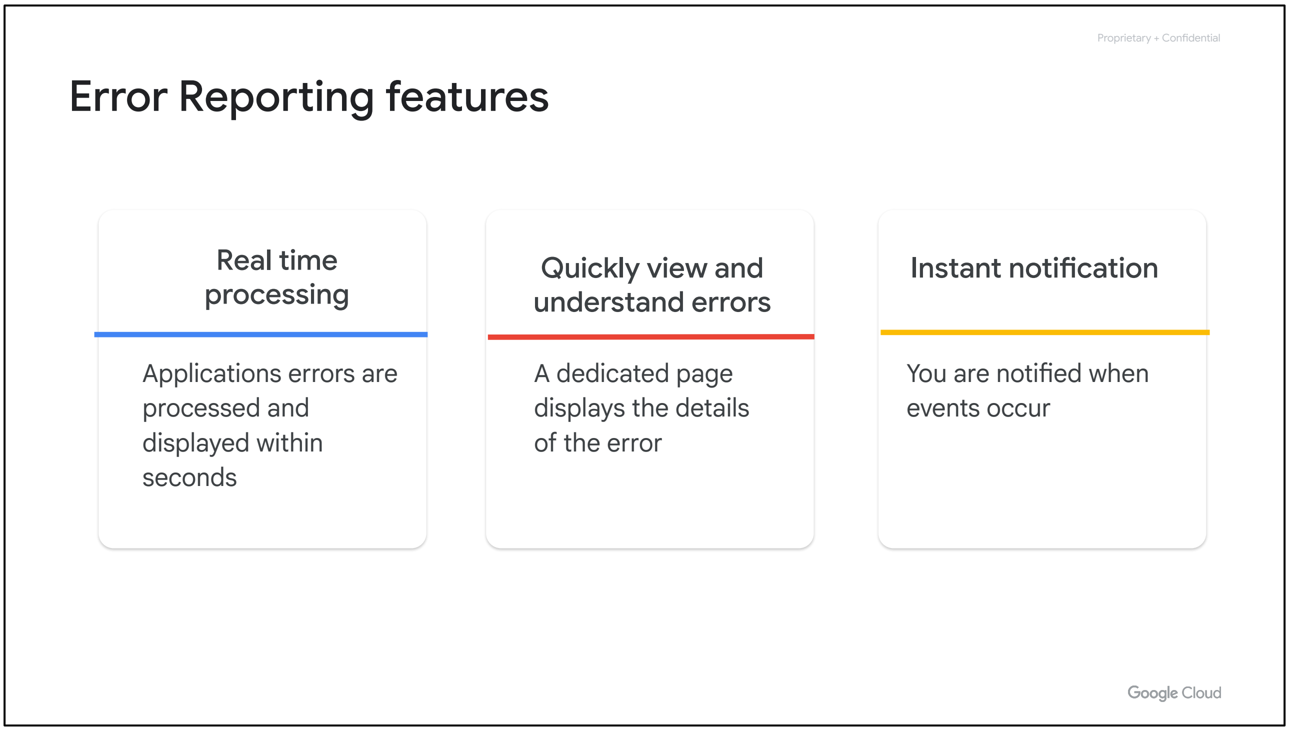
The next service we will explore is Error Reporting.



Error Reporting

Error Reporting **identifies, counts, analyzes, and aggregates** the crashes in your running cloud services.

Error Reporting counts, analyzes, and aggregates the crashes in your running cloud services.



Error Reporting features

Real time processing

Applications errors are processed and displayed within seconds

Quickly view and understand errors

A dedicated page displays the details of the error

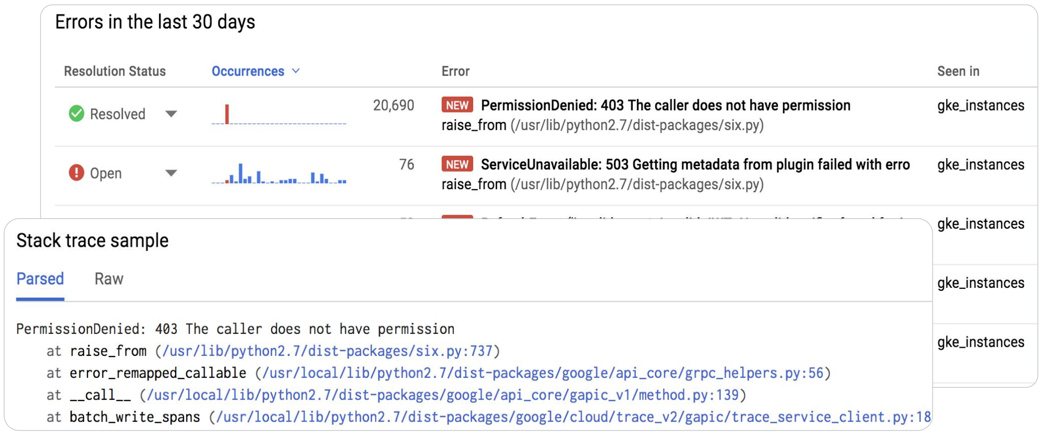
Instant notification

You are notified when events occur

Error reporting enables you to perform a lot of advanced functionalities that ensures your application runs smoothly. These include:

- Real time processing: Application errors are processed and displayed in the interface within seconds.
- Quickly view and understand errors: A dedicated page displays the details of the error: bar chart over time, list of affected versions, request URL and link to the request log.
- Instant notification: Do not wait for your users to report problems. Error Reporting is always watching your service and instantly alerts you when a new application error cannot be grouped with existing ones. Directly jump from a notification to the details of the new error.

Error Reporting interface



Errors in the last 30 days

Resolution Status	Occurrences		Error	Seen in
Resolved		20,690	NEW PermissionDenied: 403 The caller does not have permission raise_from (/usr/lib/python2.7/dist-packages/six.py)	gke_instances
Open		76	NEW ServiceUnavailable: 503 Getting metadata from plugin failed with erro raise_from (/usr/lib/python2.7/dist-packages/six.py)	gke_instances
				gke_instances
				gke_instances
				gke_instances

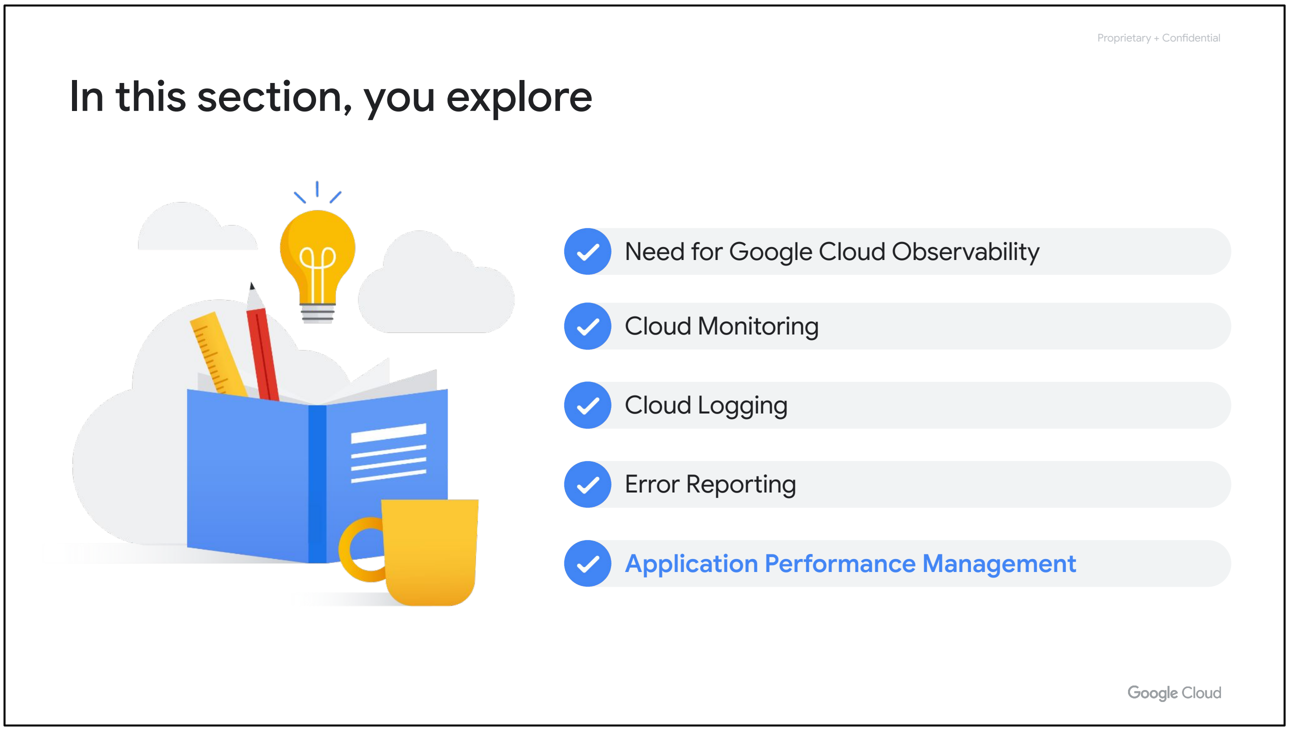
Stack trace sample

Parsed Raw

```
PermissionDenied: 403 The caller does not have permission
    at raise_from (/usr/lib/python2.7/dist-packages/six.py:737)
    at error_remapped_callable (/usr/local/lib/python2.7/dist-packages/google/api_core/grpc_helpers.py:56)
    at __call__ (/usr/local/lib/python2.7/dist-packages/google/api_core/gapic_v1/method.py:139)
    at batch_write_spans (/usr/local/lib/python2.7/dist-packages/google/cloud/trace_v2/gapic/trace_service_client.py:18
```

Google Cloud

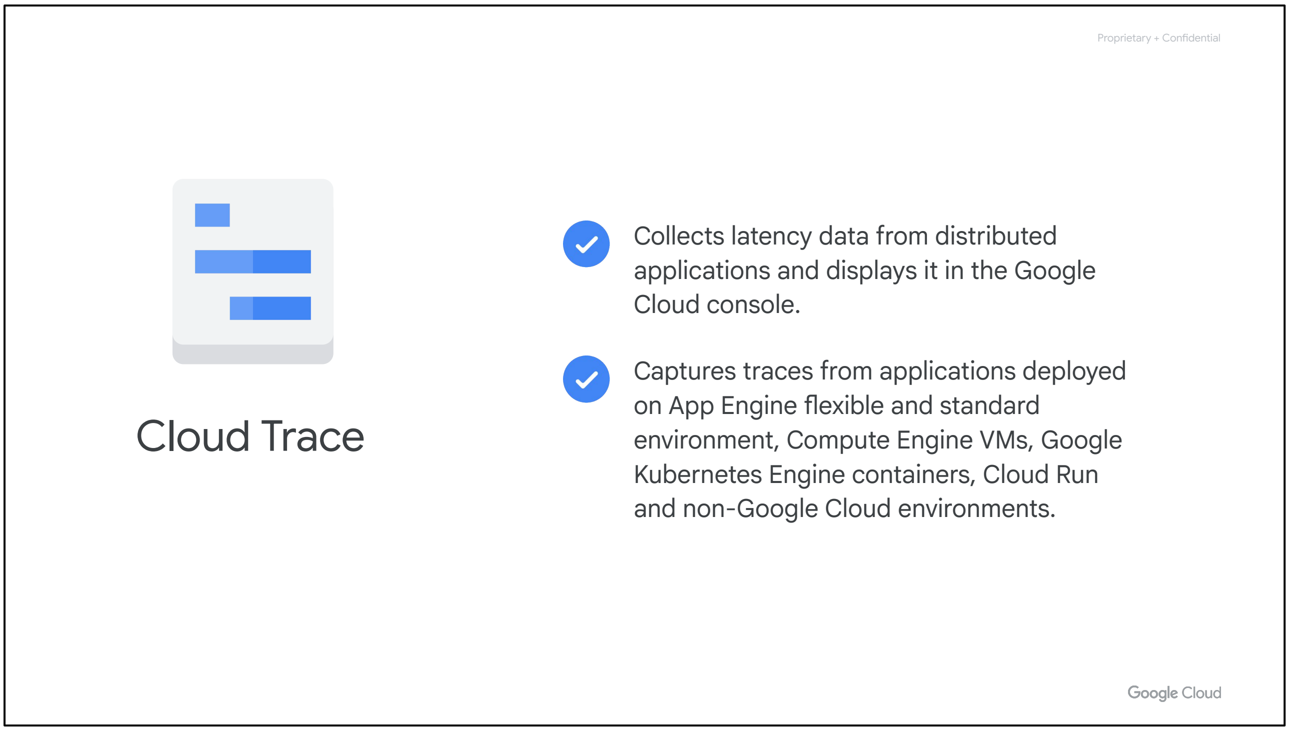
Crashes in most modern languages are exceptions which are not caught and are handled by the code itself. Its management interface displays the results with sorting and filtering capabilities. A dedicated view shows the error details: time chart, occurrences, affected user count, first- and last-seen dates, and a cleaned exception stack trace. You can also create alerts to receive notifications on new errors.



In this section, you explore

- ✓ Need for Google Cloud Observability
- ✓ Cloud Monitoring
- ✓ Cloud Logging
- ✓ Error Reporting
- ✓ Application Performance Management

The last section is about Application Performance Management this include Cloud Profiler and Cloud Trace.



Cloud Trace

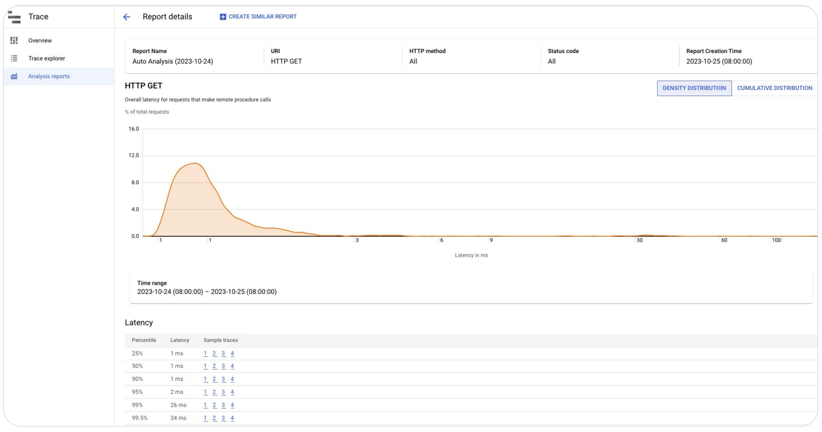
- ✓ Collects latency data from distributed applications and displays it in the Google Cloud console.
- ✓ Captures traces from applications deployed on App Engine flexible and standard environment, Compute Engine VMs, Google Kubernetes Engine containers, Cloud Run and non-Google Cloud environments.

Cloud Trace, based on the tools Google uses on its production services, is a tracing system that collects latency data from your distributed applications and displays it in the Google Cloud console.

Trace can capture traces from applications deployed on App Engine, Compute Engine VMs, and Google Kubernetes Engine containers.

Latency reports

- Provide performance insights in near-real time.
- Generate in-depth latency reports to surface performance degradations.
- Identify recent changes to application performance.



Report Name	URI	HTTP method	Status code	Report Creation Time
Auto Analysis (2023-10-24)	HTTP GET	All	All	2023-10-25 (08:00:00)

HTTP GET

Overall latency for requests that make remote procedure calls

Time range
2023-10-24 (08:00:00) – 2023-10-25 (08:00:00)

Latency

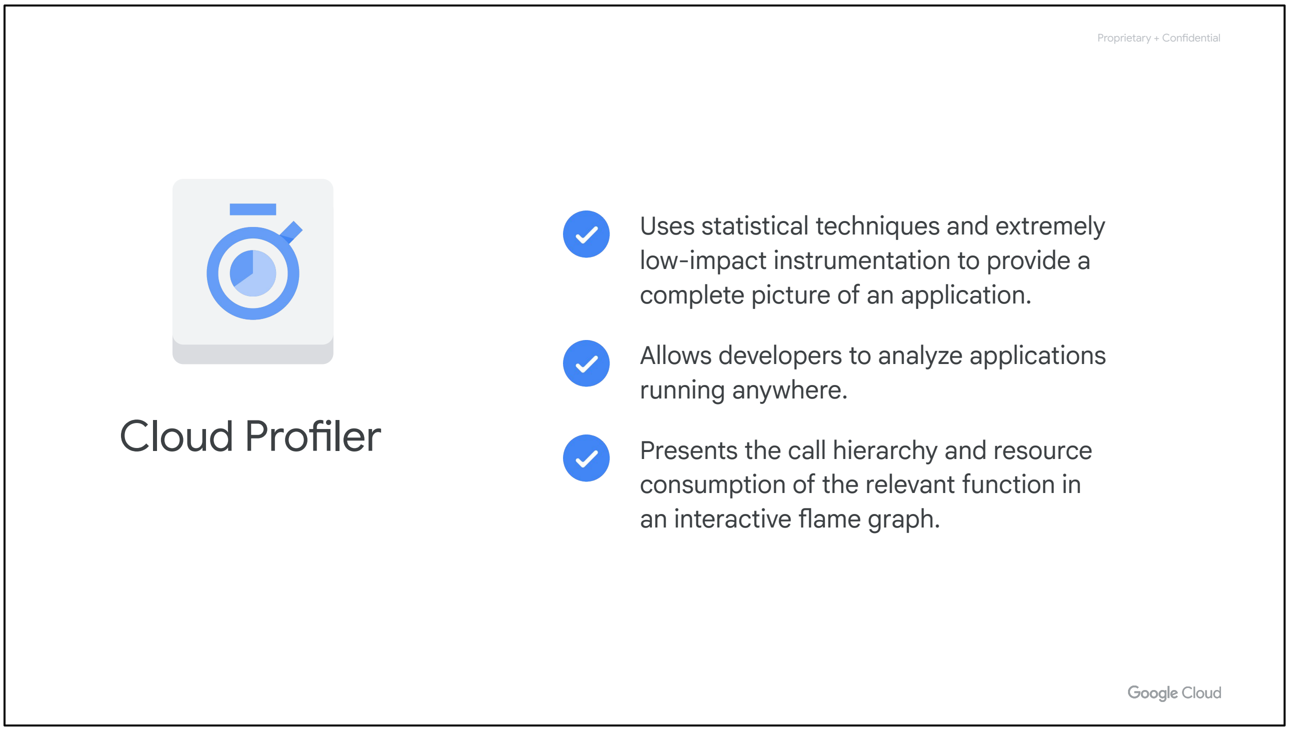
Percentile	Latency	Sample traces
25%	1 ms	1 2 3 4
50%	1 ms	1 2 3 4
90%	1 ms	1 2 3 4
95%	2 ms	1 2 3 4
99%	26 ms	1 2 3 4
99.5%	34 ms	1 2 3 4

Google Cloud

Now you can analyze changes to applications' latency profiles through the Google Cloud console and on Android devices.

Using the latency reports feature, you can:

- View performance insights in near-real time.
- Automatically analyze all of your application's traces to generate in-depth latency reports to surface performance degradations.
- Continuously gather and analyze trace data to automatically identify recent changes to application performance.



Cloud Profiler

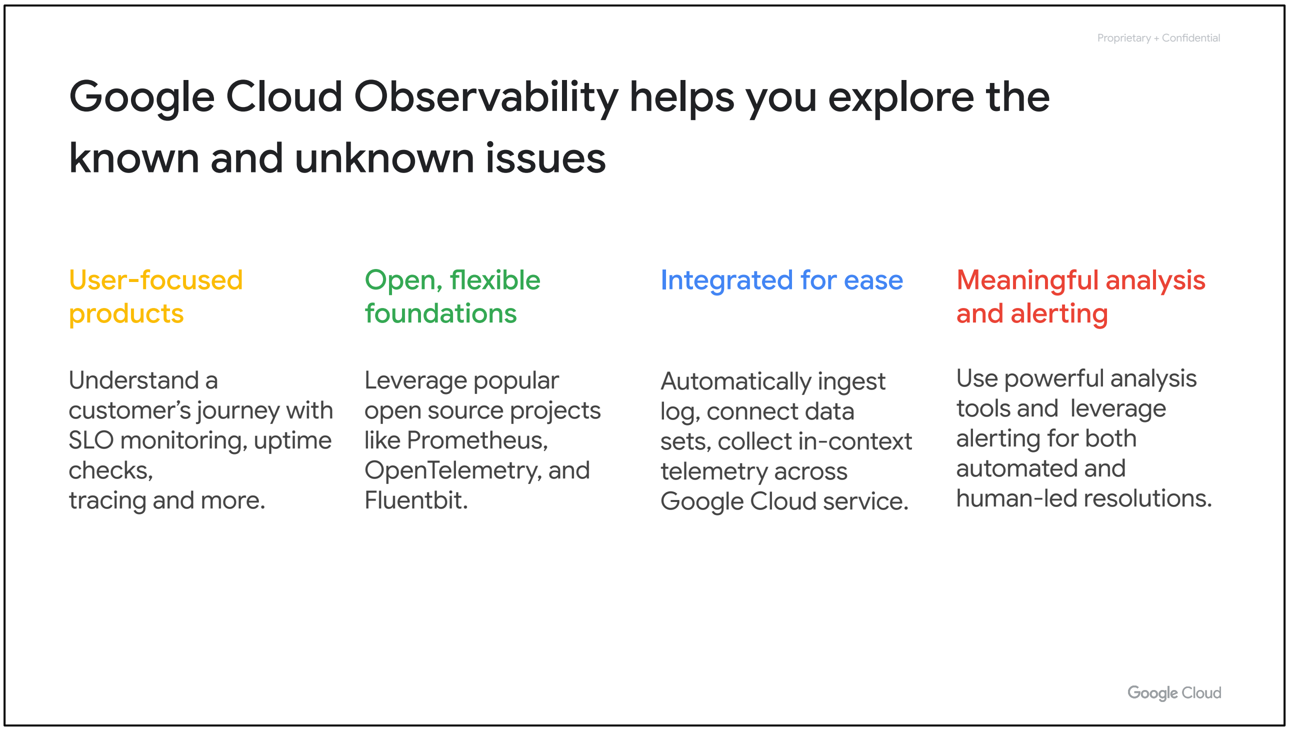
- ✓ Uses statistical techniques and extremely low-impact instrumentation to provide a complete picture of an application.
- ✓ Allows developers to analyze applications running anywhere.
- ✓ Presents the call hierarchy and resource consumption of the relevant function in an interactive flame graph.

Poorly performing code increases the latency and cost of applications and web services every day, without anyone knowing or doing anything about it.

Cloud Profiler changes this by using statistical techniques and extremely low-impact instrumentation that runs across all production application instances to provide a complete CPU and heap picture of an application without slowing it down.

With broad platform support that includes Compute Engine VMs, App Engine, and Kubernetes, it allows developers to analyze applications running anywhere, including Google Cloud, other cloud platforms, or on-premises, with support for Java, Go, Python, and Node.js.

Cloud Profiler presents the call hierarchy and resource consumption of the relevant function in an interactive flame graph that helps developers understand which paths consume the most resources and the different ways in which their code is actually called.



Google Cloud Observability helps you explore the known and unknown issues

User-focused products

Understand a customer's journey with SLO monitoring, uptime checks,
tracing and more.

Open, flexible foundations

Leverage popular open source projects like Prometheus, OpenTelemetry, and Fluentbit.

Integrated for ease

Automatically ingest log, connect data sets, collect in-context telemetry across Google Cloud service.

Meaningful analysis and alerting

Use powerful analysis tools and leverage alerting for both automated and human-led resolutions.

Overall, Google Cloud Observability helps you explore both the known and unknown issues underlying your workloads.

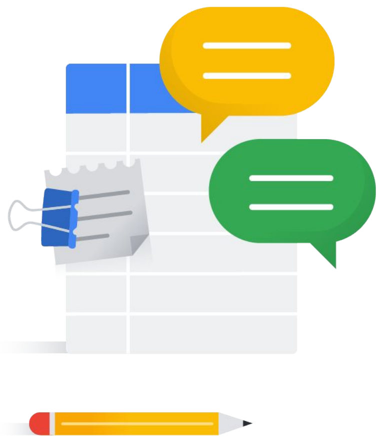
The products are user focussed designed to understand a customer's journey with SLO monitoring, uptime checks, tracing and more

They are open, flexible and leveral popular open source projects like Prometheus, OpenTelemetry, and Fluentbit.

Integrated for ease through automatic ingestion, connect data sets, in-context telemetry across Google Cloud service views.

They also provide meaningful analysis and alerting through powerful analysis tools, leverage alerting for both automated and human-led resolutions.

Knowledge Check



Quiz | Question 1

Question

You want a simple way to see the latency of requests for a web application you deployed to Cloud Run. What Google Cloud tool should you use?

A. Trace

B. Profiler

C. Metrics Explorer

D. Logs Explorer

Quiz | Question 1

Answer

You want a simple way to see the latency of requests for a web application you deployed to Cloud Run. What Google Cloud tool should you use?

A. Trace ✓

B. Profiler

C. Metrics Explorer

D. Logs Explorer

Quiz | Question 2

Question

You want to examine messages generated by running code. Which tool might be best for doing this?

A. Trace

B. Profiler

C. Metrics Explorer

D. Logs Explorer

Quiz | Question 2

Answer

You want to examine messages generated by running code. Which tool might be best for doing this?

A. Trace

B. Profiler

C. Metrics Explorer

[✓ D. Logs Explorer]

Quiz | Question 3

Question

Users have reported that an application occasionally returns garbage data instead of the intended results, but you have been unable to reproduce this problem in your test environment. Which tool might be of best help?

A. Trace

B. Profiler

C. Error Reporting

D. Logs Explorer

Quiz | Question 3

Answer

Users have reported that an application occasionally returns garbage data instead of the intended results, but you have been unable to reproduce this problem in your test environment. Which tool might be of best help?

A. Trace

B. Profiler

C. [✓ Error Reporting]

D. Logs Explorer

Quiz | Question 4

Question

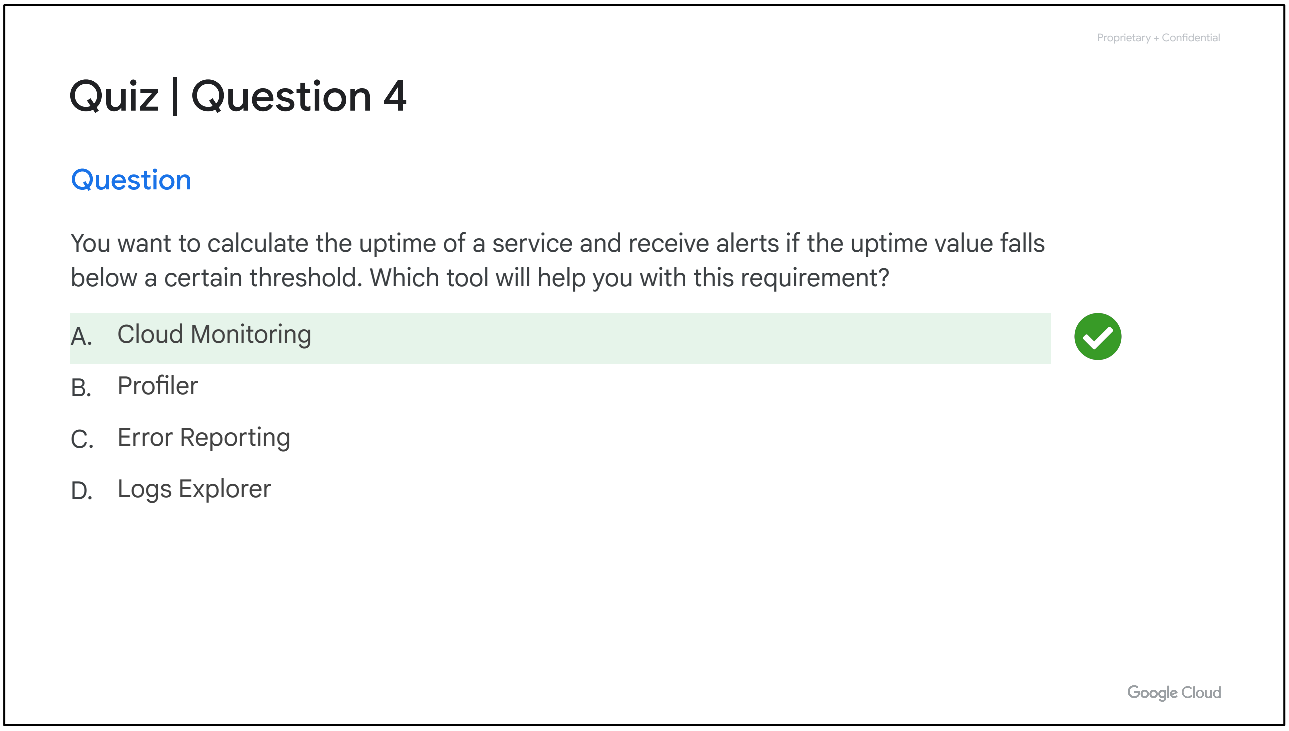
You want to calculate the uptime of a service and receive alerts if the uptime value falls below a certain threshold. Which tool will help you with this requirement?

A. Cloud Monitoring

B. Profiler

C. Error Reporting

D. Logs Explorer



Quiz | Question 4

Question

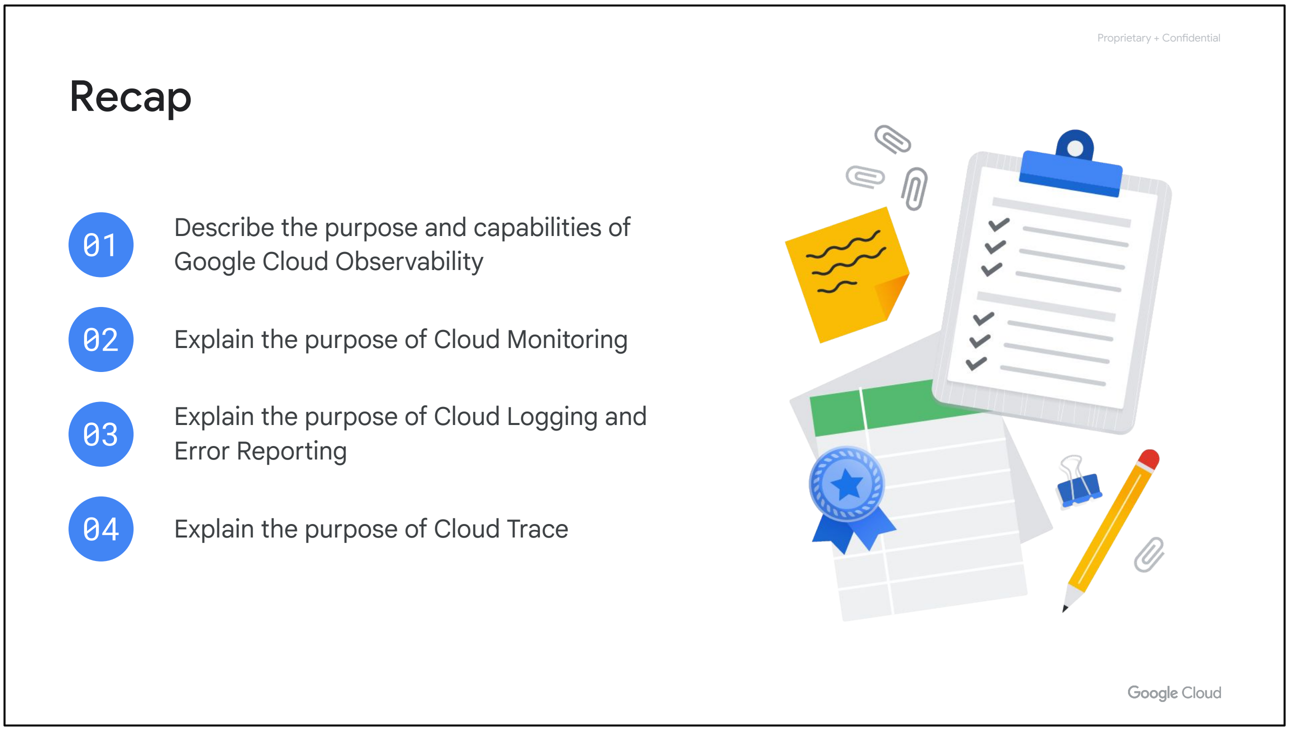
You want to calculate the uptime of a service and receive alerts if the uptime value falls below a certain threshold. Which tool will help you with this requirement?

A. Cloud Monitoring ✓

B. Profiler

C. Error Reporting

D. Logs Explorer



Proprietary + Confidential

Recap

01 Describe the purpose and capabilities of Google Cloud Observability

02 Explain the purpose of Cloud Monitoring

03 Explain the purpose of Cloud Logging and Error Reporting

04 Explain the purpose of Cloud Trace

Google Cloud

In this module, we've explored the core observability tools in Google Cloud, including Logging, Cloud Monitoring, and Error Reporting, and the application performance management tools, including Trace, and Profiler.

Now that we have a foundation, let's move on to cover the various tools in greater detail.