

# Datacentre Design & Interconnection Network

# What is a Data Center?

- A data center - also known as a *data center* or *data center* - is a facility made up of networked computers, storage systems, and computing infrastructure that businesses and other organizations use to organize, process, store large amounts of data.
- A business typically relies heavily on applications, services, and data within a data center, making it a focal point and critical asset for everyday operations.
- A small datacentre can have 1000 servers
- A warehouse level datacentre can have 4,00,000 to 1 million servers

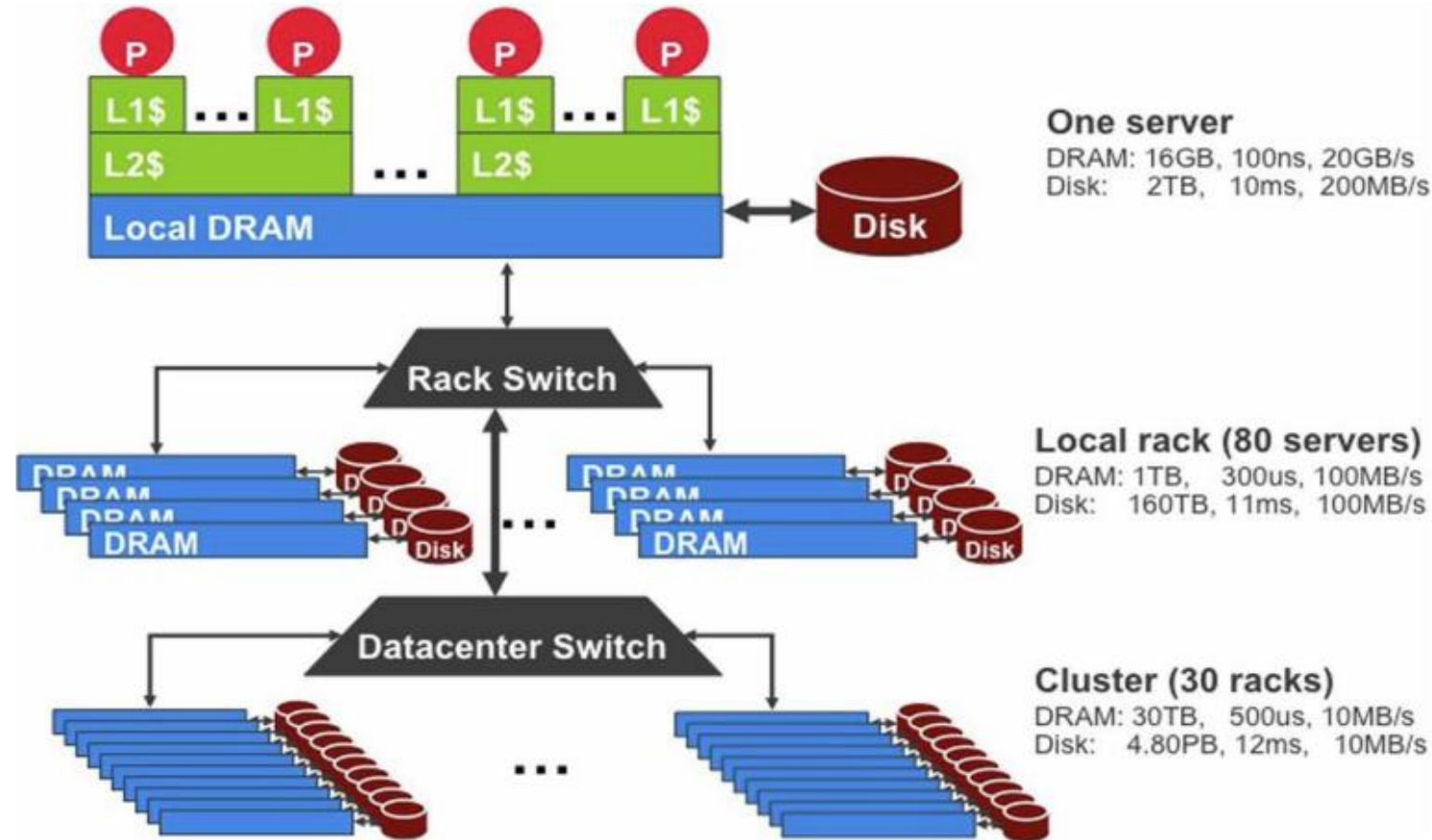
# How do Data Centers work?

- A data center facility enables an organization to assemble its resources and infrastructure for data processing, storage, and communication, including:
  - systems for storing, sharing, accessing, and processing data across the organization;
  - physical infrastructure to support data processing and data communication; And
  - Utilities such as cooling, electricity, network access, and uninterruptible power supplies (UPS).

# What are the main components of Data Centers?

- **Datacenter Resources typically include:**
  - Server consists of Sockets, CPU, & Internal Cache
  - storage subsystems : Local & Coherent DRAM & Drives
  - networking switches, routers, and firewalls;
  - cabling; And
  - Physical racks for organizing and interconnecting IT equipment.
  - power distribution and supplementary power subsystems;
  - electrical switching;
  - UPS;
  - backup generator;
  - ventilation and data center cooling systems, such as in-row cooling configurations and computer room air conditioners; And
  - Adequate provision for network carrier (telecom) connectivity.

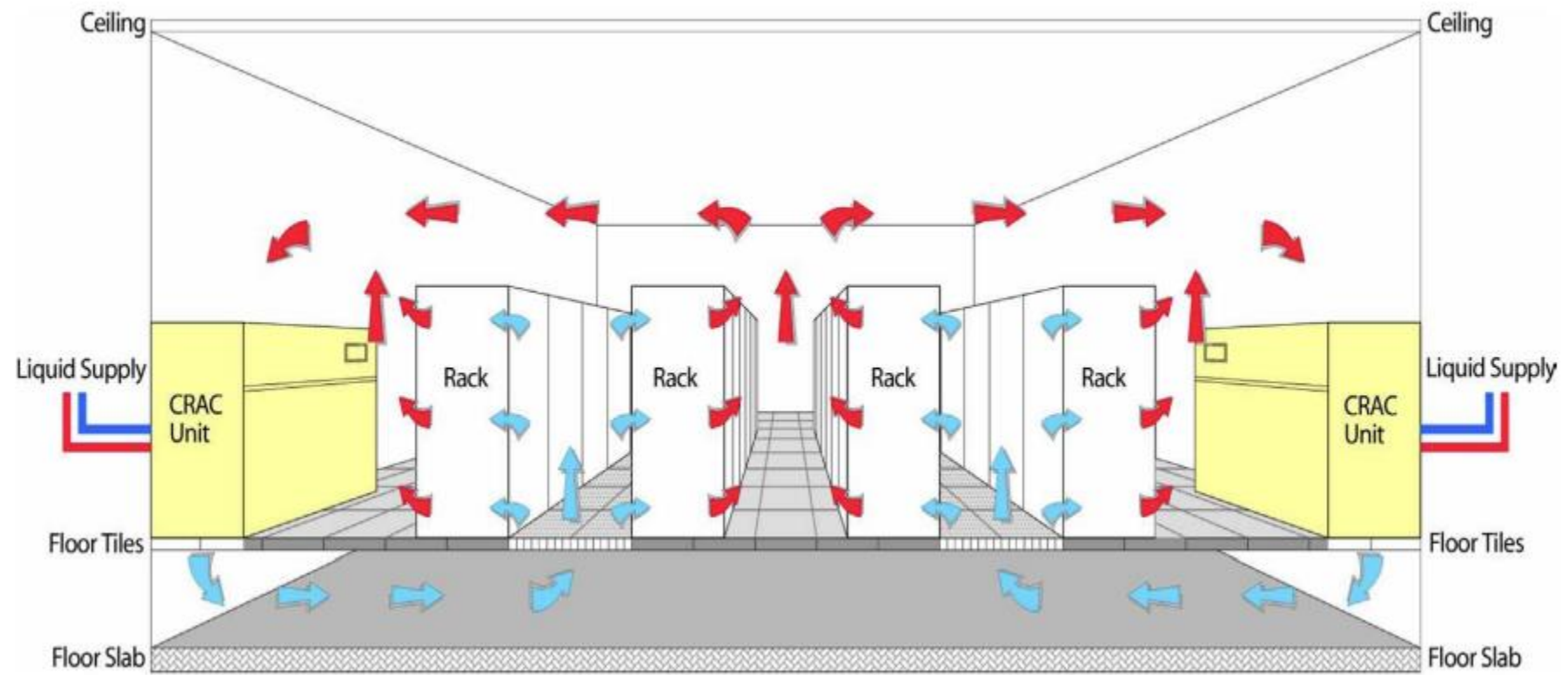
# Ware House Scale Datacentre Design



# Ware House Scale Datacentre Design

- A server consists of a number of processor sockets, each with a multicore CPU and its internal cache hierarchy, local shared and coherent DRAM, and a number of directly attached disk drives. The DRAM and disk resources within the rack are accessible through the first-level rack switches (assuming some sort of remote procedure call API to them), and all resources in all racks are accessible via the cluster-level switch.
- Switch-centric vs. Datacenter Design : Currently, there are two approaches to building datacenter-scale networks : One is switch-centric and the other is server-centric. In a switch-centric network, the switches are used to connect the server nodes. The switch centric design does not affect the server side. No modifications to the servers are needed. The server-centric design does modify the operating system running on servers. Special drivers are designed for relaying the traffic. Switches still have to be organized for achieving the connections.

# Cooling System



# Cooling System

- The data-center room has raised floors for hiding cables, power lines, and cooling supplies. The cooling system is somewhat simpler than the power system. The raised floor has a steel grid resting on stanchions about 2–4 ft above the concrete floor.
- The under-floor area is often used to route power cables to racks, but its primary use is to distribute cool air to the server rack. The CRAC (computer room air conditioning) unit pressurizes the raised floor plenum by blowing cold air into the plenum.



# Datacentre Levels

- Levels can be differentiated by available resources, data center capabilities, or uptime guarantees. **The Uptime Institute defines data center levels as:**
- **Tier I.** These are the most basic types of data centers, including UPS. Tier I data centers do not provide redundant systems but must guarantee at least 99.671% uptime.
- **Tier II.** These data centers include system, power and cooling redundancy and guarantee at least 99.741% uptime.
- **Tier III.** These data centers offer partial fault tolerance, 72-hour outage protection, full redundancy, and a 99.982% uptime guarantee.
- **Tier IV.** These data centers guarantee 99.995% uptime - or no more than 26.3 minutes of downtime per year - as well as full fault tolerance, system redundancy, and 96 hours of outage protection.

# Datacentre Interconnection Network

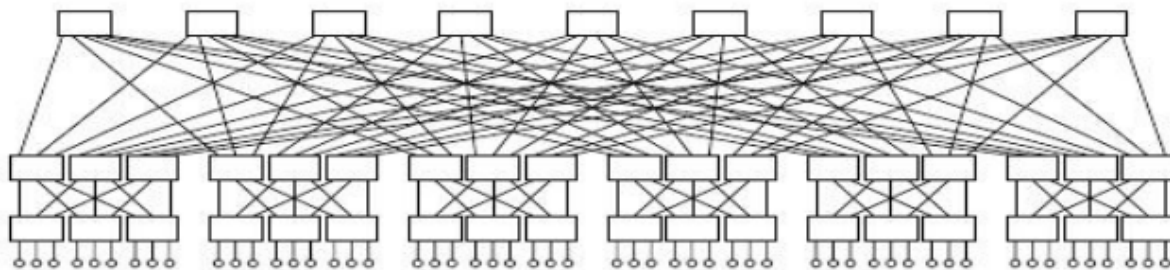
- A critical core design of Datacentre is the interconnection network among all servers in data centre cluster.
- This network must meet 5 Requirements:
  - Low Latency
  - High Bandwidth
  - Fault Tolerance
  - LOW Cost
  - Message Passing Interface

# Datacentre Interconnection Network: FAT Tree

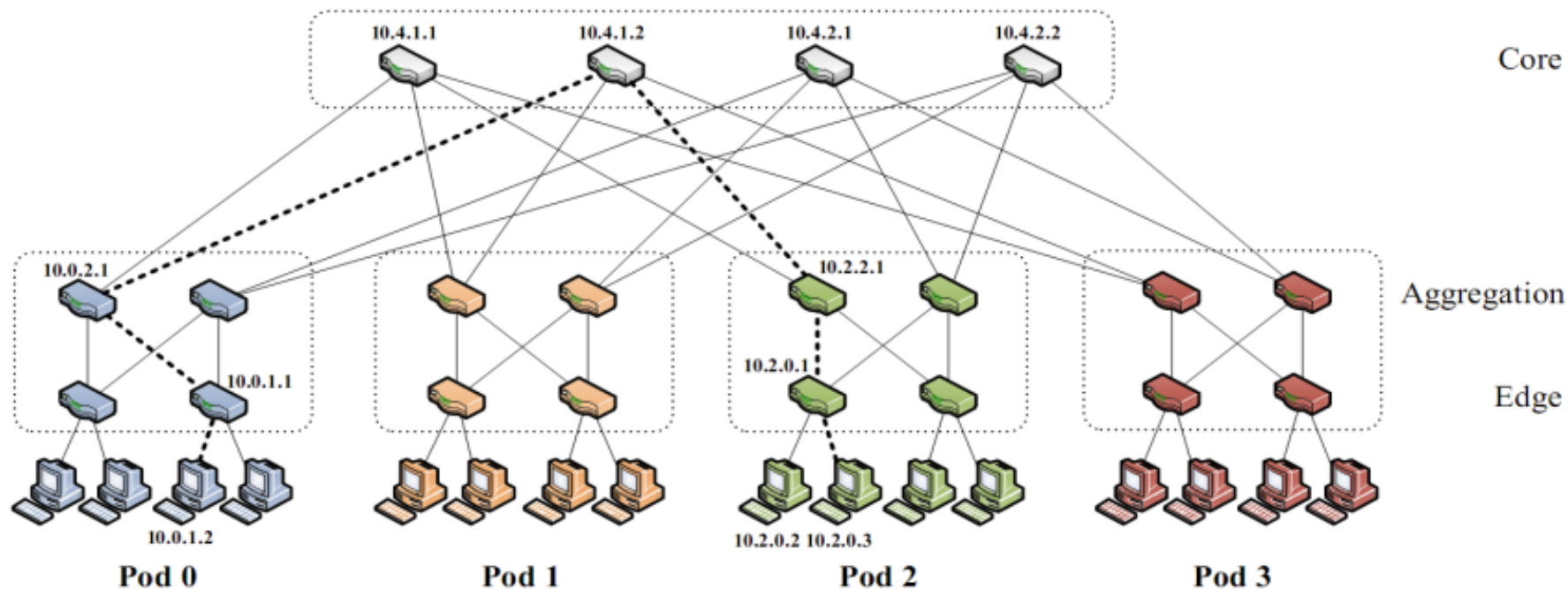
- The fat-tree topology is applied to interconnect the server nodes. The topology is organized in two layers. Server nodes are in the bottom layer. Edge switches are used to connect the nodes in the bottom layer. The upper layer aggregate the lower-layer edge switches.
- A group of aggregation switches, edge switches and their leaf nodes form a pod. On top of the pods are the core switches. Core switches provide paths among different The fat-tree structure provides multiple paths between any two server nodes in the datacenter. This provides fault-tolerant capability with alternat path in case of some isolated link failures.
- As a matter of the fact, the failure of an aggregation switch and core switch will not affect the connectivity of the whole network. The failure of any edge switch can only affect a small number of end server nodes. The extra switches in a pod provide higher bandwidth to support cloud applications in massive data movement.
- The building blocks used in the fat-tree are the low-cost Ethernet switches. This reduces the cost quite a bit. However, traditional IP/Ethernet router switch only provides one single route from the source to destination. The design must overcome this difficulty by adding redundant switches in each pod.
- They have modified the routing table inside the switches to provide extra routing paths in case of switch or link failure. The modifications of routing table and routing algorithms are built inside the switches. The end server nodes in the datacenter are not affected during one switch failure, as long as one of the alternate routing path does not fail at the same time

# FatTree-based DC Architecture

- **Why Fat-Tree?**
  - Fat tree has identical bandwidth at any bisections
  - Each layer has the same aggregated bandwidth
- Can be built using cheap devices with uniform capacity
  - Each port supports same speed as end host
  - All devices can transmit at line speed if packets are distributed uniform along available paths
- Great scalability:  $k$ -port switch supports  $k^3/4$  servers



- *Inter-connect racks (of servers) using a fat-tree topology*  
 K-ary fat tree: three-layer topology (edge, aggregation and core)
  - each pod consists of  $(k/2)^2$  servers & 2 layers of  $k/2$  k-port switches
  - each edge switch connects to  $k/2$  servers &  $k/2$  aggr. switches
  - each aggr. switch connects to  $k/2$  edge &  $k/2$  core switches
  - $(k/2)^2$  core switches: each connects to  $k$  pods



# FatTree Topology is great, But...

Does using fat-tree topology to inter-connect racks of servers in itself sufficient?

- What routing protocols should we run on these switches?
- Layer 2 switch algorithm: data plane flooding!
- Layer 3 IP routing:
  - shortest path IP routing will typically use only one path despite the path diversity in the topology
  - if using equal-cost multi-path routing at each switch independently and blindly, packet re-ordering may occur; further load may not necessarily be well-balanced
  - Aside: control plane flooding!



# Cloud Logs

Logs Explorer

OPTIONS

REFINE SCOPE

Project

SHARE LINK

Query

Recent (1)

Saved (0)

Suggested (3)

resource.type="k8s\_container"

Log fields

Search fields and values

RESOURCE TYPE

Kubernetes Container

Clear X

SEVERITY

Info

2,228

Error

24

LOG NAME

stdout

2,147

stderr

105

PROJECT ID

ygrinshteyn-sandbox

2,252

LOCATION

us-central1

2,252

CLUSTER NAME

prod-cluster-autopilot

2,252

NAMESPACE NAME

default

2,168

kube-system

84

POD NAME

Query results

SEVERITY

TIMESTAMP

PDT

SUMMARY

Showing logs for last 1 hour from 7/1/21, 12:41 PM to 7/1/21, 1:41 PM.

Extend time by: 1 hour

Edit time

> i

2021-07-01 13:41:08.028 PDT

lbm-slis-server

Status log - 200

> i

2021-07-01 13:41:08.027 PDT

lbm-slis-server

request made

> i

2021-07-01 13:41:03.338 PDT

lbm-slis-server

Status log - 200

> i

2021-07-01 13:41:03.337 PDT

lbm-slis-server

request made

> i

2021-07-01 13:40:53.726 PDT

lbm-slis-server

Status log - 200

> i

2021-07-01 13:40:53.725 PDT

lbm-slis-server

request made

> i

2021-07-01 13:40:53.699 PDT

lbm-slis-server

Status log - 200

> i

2021-07-01 13:40:53.699 PDT

lbm-slis-server

request made

> i

2021-07-01 13:40:52.736 PDT

lbm-slis-server

Status log - 200

> i

2021-07-01 13:40:52.736 PDT

lbm-slis-server

request made

> i

2021-07-01 13:40:51.602 PDT

lbm-slis-server

Status log - 200

> i

2021-07-01 13:40:51.602 PDT

lbm-slis-server

request made

> E

2021-07-01 13:40:47.038 PDT

lbm-slis-server

Status log - 500

> i

2021-07-01 13:40:47.038 PDT

lbm-slis-server

request made

> i

2021-07-01 13:40:45.687 PDT

lbm-slis-server

Status log - 200

> i

2021-07-01 13:40:45.687 PDT

lbm-slis-server

request made

> i

2021-07-01 13:40:45.203 PDT

lbm-slis-server

Status log - 200

> i

2021-07-01 13:40:45.202 PDT

lbm-slis-server

request made

> i

2021-07-01 13:40:41.674 PDT

lbm-slis-server

Status log - 200

> i

2021-07-01 13:40:41.673 PDT

lbm-slis-server

request made

# Google Cloud Logging : Case Study

- Cloud Logging is a service for storing, viewing and interacting with logs.
- Answers the questions “Who did what, where and when” within the GCP projects
- Maintains non-tamperable audit logs for each project and organizations
- Logs buckets are a regional resource, which means the infrastructure that stores, indexes, and searches the logs are located in a specific geographical location. Google manages that infrastructure so that the applications are available redundantly across the zones within that region.



# Cloud Logs : Parameter

The screenshot displays the Google Cloud Logs Explorer interface. At the top, there's a 'Logs Explorer' header with an 'OPTIONS' dropdown, a 'REFINE SCOPE' button, and a 'Project' dropdown. Below this, a 'Query' tab is active, showing a recent query: `resource.type="k8s_container"`. On the left, a 'Log fields' sidebar allows filtering by 'RESOURCE TYPE' (Kubernetes Container is selected), 'SEVERITY' (Info, Error), 'LOG NAME' (stdout, stderr), 'PROJECT ID', 'LOCATION' (us-central1), 'CLUSTER NAME' (prod-cluster-autopilot), 'NAMESPACE NAME' (default), and 'POD NAME'. The main area shows 'Query results' for the last 1 hour. A specific log entry is expanded, showing a JSON object with a 'resource' field containing labels for the cluster, container, location, namespace, pod, and project. The 'lbn-slis-server' container is highlighted with a red box. Below the log entry, a table shows other recent logs from the same pod.

**Log fields**

- RESOURCE TYPE
  - Kubernetes Container
- SEVERITY
  - Info
  - Error
- LOG NAME
  - stdout
  - stderr
- PROJECT ID
  - gcp-project-id
- LOCATION
  - us-central1
- CLUSTER NAME
  - prod-cluster-autopilot
- NAMESPACE NAME
  - default
- POD NAME
  - kube-system

**Query results**

Showing logs for last 1 hour from 7/1/21, 12:41 PM to 7/1/21, 1:41 PM. [Extend time by: 1 hour](#) [Edit time](#)

2021-07-01 13:41:08.028 PDT **lbn-slis-server** Status log - 200

```
{
  insertId: "2a0whjtawblxltj2"
  labels: {
    compute.googleapis.com/resource_name: "gk3-prod-cluster-autopilot-default-pool-cbb6e9d5-91g3"
    k8s-pod/app: "lbn-slis"
    k8s-pod/pod-template-hash: "b94f879cd"
  }
  logName: "projects/gcp-project-id/logs/lbn-slis-server"
  receiveTimestamp: "2021-07-01T20:41:12.951387455Z"
  resource: {
    labels: {
      cluster_name: "prod-cluster-autopilot"
      container_name: "lbn-slis-server"
      location: "us-central1"
      namespace_name: "default"
      pod_name: "lbn-slis-deployment-b94f879cd-rzg4v"
      project_id: "gcp-project-id"
    }
    type: "k8s_container"
  }
  severity: "INFO"
  textPayload: "Status log - 200"
  timestamp: "2021-07-01T20:41:08.028351884Z"
}
```

Timestamp	Pod Name	Log Message
2021-07-01 13:41:08.027 PDT	lbn-slis-server	request made
2021-07-01 13:41:03.338 PDT	lbn-slis-server	Status log - 200

# Log Sources

## Platform logs



- Services
- Audit logs

## Custom logs



- Agent
- API and client libraries
- User apps
- Third-party applications
- System software

## Network logs



- VPC flow
- Firewall rules
- NAT gateways
- Load Balancer

# Log Sources

## ❑ Cloud Platform Logs

- Cloud platform logs are service-specific logs that can help troubleshoot and debug issues, as well as better understand the Google Cloud services.
- Cloud Platform logs are logs generated by GCP services and vary depending on which Google Cloud resources are used in your Google Cloud project or organization.

- **Access Transparency Logs**

provides logs of actions taken by Google staff when accessing the Google Cloud content.  
can help track compliance with the organization's legal and regulatory requirements.  
have 400-day retention

# Log Sources

## Security Logs

- Audit Logs

- Cloud Audit Logs includes three types of audit logs:

- Admin Activity,
    - Data Access, and
    - System Event.

- Cloud Audit Logs provide audit trails of administrative changes and data accesses of the Google Cloud resources.

- Admin Activity

- captures user-initiated resource configuration changes
      - enabled by default
      - no additional charge
      - admin activity – administrative actions and API calls
      - have 400-day retention

- System Events

- captures system initiated resource configuration changes
      - enabled by default
      - no additional charge
      - system events – GCE system events like live migration
      - have 400-day retention

- Data Access logs

- Log API calls that create, modify or read user-provided data for e.g. object created in a GCS bucket.
      - 30-day retention
      - disabled by default
      - size can be huge
      - charged beyond free limits
      - Available for GCP-visible services only. Not available for public resources.

# Log Sources

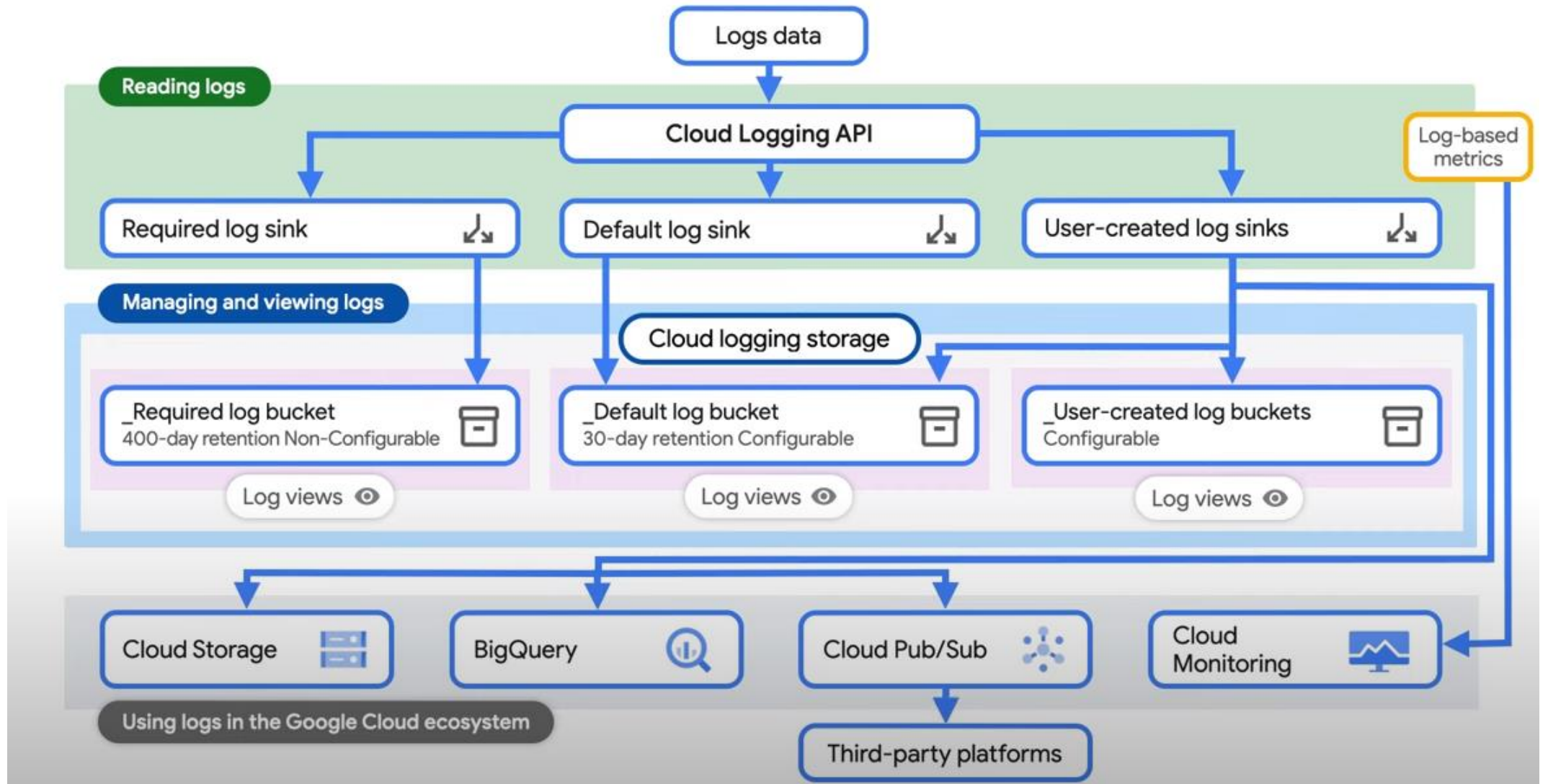
- **User Logs**

- User logs are generated by user software, services, or applications and written to Cloud Logging using a logging agent, the Cloud Logging API, or the Cloud Logging client libraries

- **Agent logs**

- produced by logging agent installed that collects logs from user applications and VMs
- covers log data from third-party applications
- charged beyond free limits
- 30-day retention

# Cloud Logging Service : Architecture



# Cloud Logging Service : Architecture

- For each Google Cloud project, Logging automatically creates two logs buckets: `_Required` and `_Default`.
  - `_Required` bucket
    - holds Admin Activity audit logs, System Event audit logs, and Access Transparency logs
    - retains them for 400 days.
    - the retention period of the logs stored here cannot be modified.
    - aren't charged for the logs stored in `_Required`, and
    - cannot delete this bucket.
  - `_Default` bucket
    - holds all other ingested logs in a Google Cloud project except for the logs held in the `_Required` bucket.
    - are charged
    - are retained for 30 days, by default, and can be customized from 1 to 3650 days
  - these buckets cannot be deleted
- All logs generated in the project are stored in the `_Required` and `_Default` logs buckets, which live in the project that the logs are generated in
- Logs buckets only have regional availability, including those created in the global region.

# Cloud Logging Service : Architecture

- Log entries are stored in logs buckets for a specified length of time i.e. retention period and are then deleted and cannot be recovered
- Logs can be exported by configuring log **sinks**, which then continue to export log entries as they arrive in Logging.
- A sink includes a destination and a filter that selects the log entries to export.
- Exporting involves writing a **filter** that selects the log entries to be exported, and choosing a **destination** from the following options:
  - Cloud Storage: JSON files stored in buckets for long term retention
  - BigQuery: Tables created in BigQuery datasets. for analytics
  - Pub/Sub: JSON messages delivered to Pub/Sub topics to stream to other resources. Supports third-party integrations, such as Splunk
  - Another Google Cloud Cloud project: Log entries held in Cloud Logging logs buckets.
- Every time a log entry arrives in a project, folder, billing account, or organization resource, Logging compares the log entry to the sinks in that resource. Each sink whose filter matches the log entry writes a copy of the log entry to the sink's export destination.
- Exporting happens for new log entries only, it is not retrospective




# Cloud Log : Access Privileges

## Refine scope ✕

---


☐



### Scope by project

Search based on the logs from this project only. Scoping by project allows you to query only the logs that are generated by this project regardless of where the logs are stored.

☒



### Scope by storage ⓘ

Search based on one or more logs storage views. Scoping by views allows you to query logs buckets within this project or logs routed from this project to buckets in other projects.

<input checked="" type="checkbox"/>	MY-DOC-PROJECT-205801 / BUCKET_FOR_TEST_LOGS
<input type="checkbox"/>	_AllLogs <small>projects/my-doc-project-205801/locations/europe-west1/buckets/bucket_f...</small>
<input checked="" type="checkbox"/>	MY-DOC-PROJECT-205801 / _DEFAULT
<input type="checkbox"/>	_AllLogs <small>projects/my-doc-project-205801/locations/global/buckets/_Default/views/_...</small>
<input type="checkbox"/>	_Default <small>projects/my-doc-project-205801/locations/global/buckets/_Default/views/_...</small>
<input checked="" type="checkbox"/>	MY-DOC-PROJECT-205801 / _REQUIRED
<input type="checkbox"/>	_AllLogs <small>projects/my-doc-project-205801/locations/global/buckets/_Required/views...</small>
<input checked="" type="checkbox"/>	MY-DOC-PROJECT-205801 / SAMPLE-BUCKET
<input type="checkbox"/>	_AllLogs <small>projects/my-doc-project-205801/locations/global/buckets/sample-bucket/v...</small>
<input checked="" type="checkbox"/>	my_view <small>projects/my-doc-project-205801/locations/global/buckets/sample-bucket/v...</small>

# Cloud Log : Access Privileges

- Cloud Logging IAM Roles
- Logs Viewer – View logs except Data Access/Access Transparency logs
- Private Logs Viewer – View all logs
- Logging Admin – Full access to all logging actions
- Project Viewer – View logs except Data Access/Access Transparency logs
- Project Editor – Write, view, and delete logs. Create log based metrics. However, it cannot create export sinks or view Data Access/Access Transparency logs.
- Project Owner – Full access to all logging actions

# Logs Explorer

Google Cloud Platform

ygrinshteyn-sandbox

Search products and resources

Logs Explorer

OPTIONS

REFINE SCOPE

Project

SHARE LINK

9:29:00 AM - 9:44:00 AM

PAGE LAYOUT

LEARN

Query

Recent (7)

Saved (0)

Suggested (3)

Resource

Log name

Severity

1

Log fields

Search fields and values

RESOURCE TYPE

Kubernetes Cluster55,312

Audited Resource1,042

GKE Cluster Operations1,006

VM Instance985

Kubernetes Container564

GCE Instance Group Manager465

GAE Application413

GCS Bucket409

Kubernetes Node183

Cloud Scheduler Job60

SHOW MORE

SEVERITY

Default56,789

Info3,849

Error46

Notice9

Warning8

Histogram

2K

1K

0

Jul 2, 9:29:00 AM

9:35 AM

9:40 AM

Jul 2, 9:44:10 AM

Query results

Jump to now

Actions

Configure

SEVERITY	TIMESTAMP	PDT	SUMMARY
> *	2021-07-02 09:43:59.992 PDT		prod-cluster-autopilot k8s.io io.k8s.get readyz system:gke-master-healthc... audit_log, method: "io.k8s.get", principal_email: "system:gke-master-healthcheck"
> *	2021-07-02 09:43:59.986 PDT		prod-cluster-autopilot k8s.io io.k8s.core.v1.configmaps.get _es/kube-system/configmaps/clustermetrics system:clustermetrics audit_log, method: "io.k8s.core.v1.conf..."
> *	2021-07-02 09:43:59.982 PDT		prod-cluster-autopilot k8s.io _ion.rbac.v1.clusterrolebindings.get _lebindings/system:gke-hpa-service-reader 115174137731346286123 audit_log, method: "io.k8s.autho..."
> *	2021-07-02 09:43:59.945 PDT		prod-cluster-autopilot k8s.io _c.v1beta1.clusterrolebindings.patch _ebindings/master-monitoring-role-binding system:addon-manager audit_log, method: "io.k8s.author..."
> *	2021-07-02 09:43:59.936 PDT		prod-cluster-autopilot k8s.io _bac.v1beta1.clusterrolebindings.get _ebindings/master-monitoring-role-binding system:addon-manager audit_log, method: "io.k8s.author..."
> *	2021-07-02 09:43:59.925 PDT		prod-cluster-autopilot k8s.io _horization.rbac.v1.clusterroles.get _usterroles/system:master-monitoring-role system:addon-manager audit_log, method: "io.k8s.author..."
> *	2021-07-02 09:43:59.924 PDT		prod-cluster-autopilot k8s.io io.k8s.core.v1.configmaps.update _kube-system/configmaps/cluster-kubestore system:kubestore-collecto... audit_log, method: "io.k8s.core..."
> *	2021-07-02 09:43:59.921 PDT		prod-cluster-autopilot k8s.io _o.k8s.coordination.v1.leases.update _spaces/kube-system/leases/kube-scheduler system:kube-scheduler audit_log, method: "io.k8s.coord..."
> *	2021-07-02 09:43:59.913 PDT		prod-cluster-autopilot k8s.io io.k8s.coordination.v1.leases.get _a/external-resizer-pd-csi-storage-gke-io system:pdcsi-controller audit_log, method: "io.k8s.coordi..."
> *	2021-07-02 09:43:59.916 PDT		prod-cluster-autopilot k8s.io io.k8s.coordination.v1.leases.get _spaces/kube-system/leases/kube-scheduler system:kube-scheduler audit_log, method: "io.k8s.coordina..."
> *	2021-07-02 09:43:59.908 PDT		prod-cluster-autopilot k8s.io io.k8s.core.v1.configmaps.get _kube-system/configmaps/cluster-kubestore system:kubestore-collecto... audit_log, method: "io.k8s.core.v1..."
> *	2021-07-02 09:43:59.907 PDT		prod-cluster-autopilot k8s.io _horization.rbac.v1.clusterroles.get _on.k8s.io/v1/clusterroles/cloud-provider system:addon-manager audit_log, method: "io.k8s.author..."
> *	2021-07-02 09:43:59.896 PDT		prod-cluster-autopilot k8s.io _k8s.authorization.rbac.v1.roles.get _spaces/kube-system/roles/cloud-provider system:addon-manager audit_log, method: "io.k8s.author..."
> *	2021-07-02 09:43:59.897 PDT		prod-cluster-autopilot k8s.io io.k8s.coordination.v1.leases.get _spaces/kube-system/leases/kube-scheduler system:kube-scheduler audit_log, method: "io.k8s.coordina..."
> *	2021-07-02 09:43:59.891 PDT		prod-cluster-autopilot k8s.io _horization.rbac.v1.clusterroles.get _s.io/v1/clusterroles/gce:cloud-provider system:addon-manager audit_log, method: "io.k8s.author..."
> *	2021-07-02 09:43:59.888 PDT		prod-cluster-autopilot k8s.io _k8s.authorization.rbac.v1.roles.get _ces/kube-system/roles/gce:cloud-provider system:addon-manager audit_log, method: "io.k8s.author..."
> *	2021-07-02 09:43:59.878 PDT		prod-cluster-autopilot k8s.io io.k8s.core.v1.configmaps.get _kube-system/configmaps/cluster-kubestore system:kubestore-collecto... audit_log, method: "io.k8s.core.v1..."
> *	2021-07-02 09:43:59.866 PDT		prod-cluster-autopilot k8s.io _ion.rbac.v1.clusterrolebindings.get _1/clusterrolebindings/gce:cloud-provider system:addon-manager audit_log, method: "io.k8s.author..."
> *	2021-07-02 09:43:59.845 PDT		monitoring.googleapis.com google.monitoring.v3.MetricService.CreateTimeSeries projects/ygrinshteyn-sandbox 183592734342-compute@developer.gservicesaccount.com audit_l...
> *	2021-07-02 09:43:59.838 PDT		prod-cluster-autopilot k8s.io _horization.rbac.v1.rolebindings.get _e-system/rolebindings/gce:cloud-provider system:addon-manager audit_log, method: "io.k8s.author..."

Show debug panel

# Cloud Logging Agent

- Cloud Logging agent streams logs from VM instances and from selected third-party software packages to Cloud Logging.
- Cloud Logging Agent helps capture logs from GCE and AWS EC2 instances
- VM images for GCE and Amazon EC2 don't include the Logging agent and must be installed explicitly.
- Cloud Logging Agent uses fluentd for capturing logs
- Logging features include:
  - Standard system logs (/var/log/syslog and /var/log/messages for Linux, Windows Event Log) collected with no setup.
  - High throughput capability, taking full advantage of multi-core architecture.
  - Efficient resource (e.g. memory, CPU) management.
  - Custom log files.
  - JSON logs.
  - Plain text logs.
  - Regex-based parsing.
  - JSON-based parsing.
- **Logging agent** is pre-configured to send logs from VM instances to Cloud Logging which include syslogs, and other third-party applications like Redis,
- Cloud Logging Agent provides additional plugins and configurations like filter\_record\_transformer that can help modify, delete log entries before the logs
- are pushed to Cloud Logging *for e.g. masking of sensitive PII information*
- Ops Agent doesn't directly support automatic log parsing for third-party applications, but it can be configured to parse these files.

# Collection

- After completing discovery, it is time to collect the logs
- Not as simple as “go get them”

## Difficulty Level

Easy

Meh

Insane



Requires a mix of older log methods and new ones

- Syslog daemons and log agents work on virtual machines
- Major vendors have central logging frameworks
- Container logging
- APIs

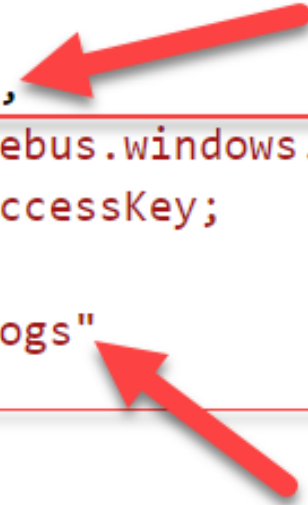
# Major Vendor Support

- SIEM and logging solutions may have native support for Amazon, Google, and Microsoft logs
- **Azure Monitor** logs to **Event Hubs**
  - Event Hubs useful for many other Microsoft logs
  - Collection may combine with Storage blobs for tracking
- Amazon uses **CloudTrail** and **S3** buckets
  - Also, supports **CloudWatch** for central logging of other services

# Azure Event Hubs

- Microsoft cloud logging supports Event Hubs and Monitor
  - Real-time logging
  - Easy to integrate with SIEM
- Requires secure key use
  - And proper access control lists (ACLs)

```
input {
  azure_event_hubs {
    event_hub_connections => [
      "Endpoint=sb://azuremonitorlog.servicebus.windows.net/;
      SharedAccessKeyName=RootManageSharedAccessKey;
      SharedAccessKey=keygoeshere;
      EntityPath=insights-operational-logs",
      "Endpoint=sb://azuremonitorlog.servicebus.windows.net/;
      SharedAccessKeyName=RootManageSharedAccessKey;
      SharedAccessKey=keygoeshere;
      EntityPath=insights-logs-operationallogs"
    ]
  }
}
```





# S3 Buckets

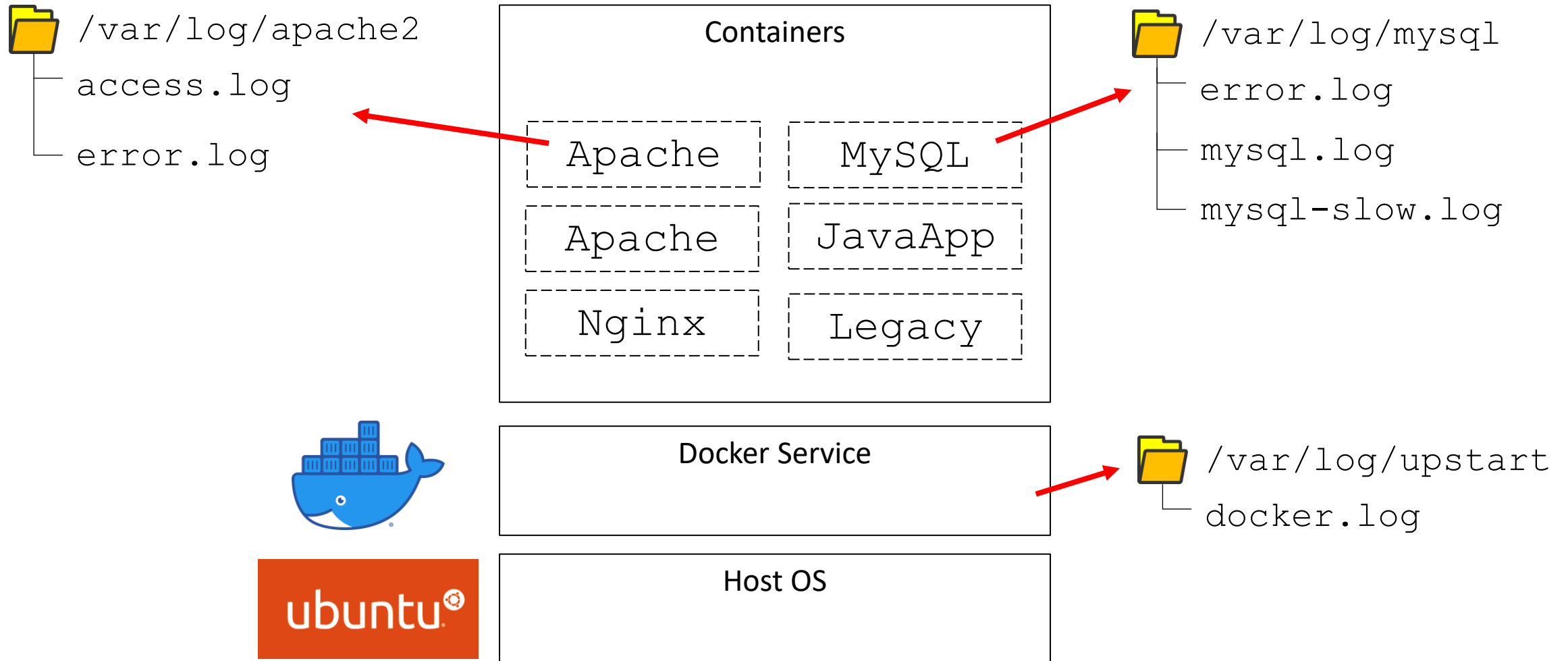
- Similar methodology for **S3**
- And **CloudWatch**
- CloudTrail can log to either S3 or CloudWatch
- Plugin access information into SIEM

```
input {  
  s3 {  
    tags => [ "s3", "dns" ]  
    access_key_id => "ACCESSKEYGOESHERE" 1  
    secret_access_key => "NOSECRETFORYOU" 2  
    region => "us-east-2" 3  
    bucket => "cisco-managed-us-east-2" 4  
    prefix => "FolderPrefixGoesHere/dnslogs/"  
    additional_settings => {  
      force_path_style => true  
      follow_redirects => false  
    }  
  }  
}
```

Cisco Umbrella log example



# Container Architecture



# Container Logging

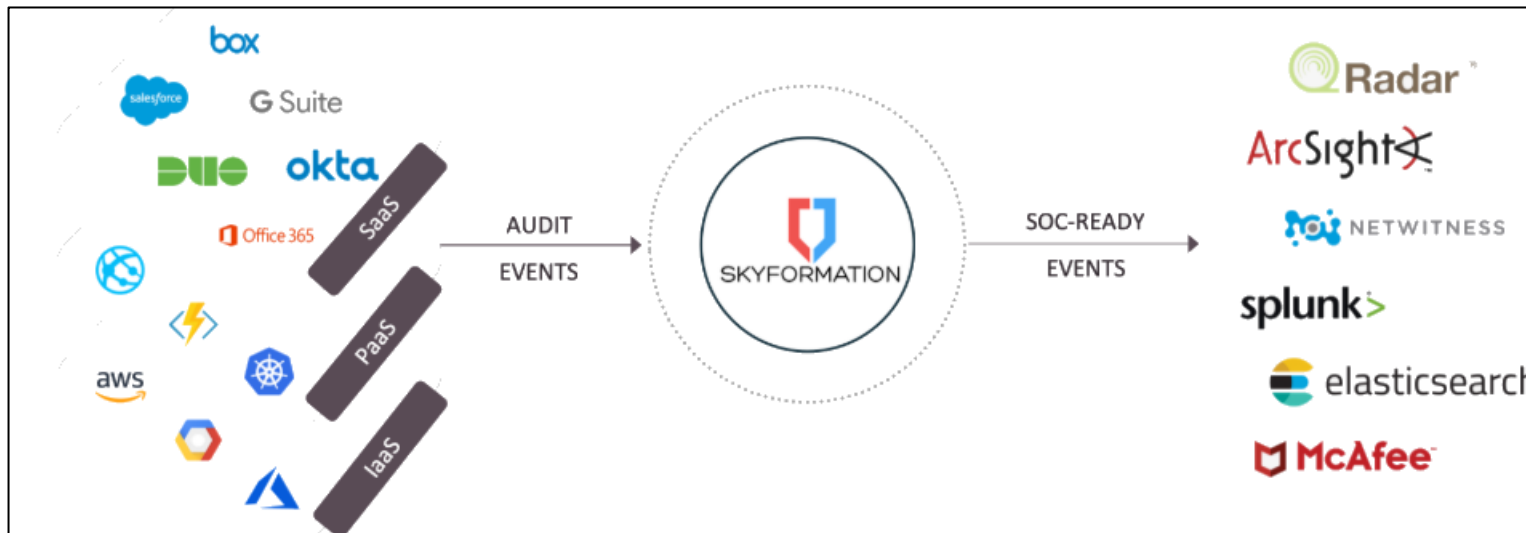
- Four common options for collecting logs from containers
- **Persistent data volume or bind mount**
  - Example: S3 bucket, google storage bucket, etc.
- **Application inside container** (custom coding)
- **Monitoring container (“sidecar”)**
- **Daemon log drivers**
  - Example: Amazon log driver to S3 or CloudWatch

# API Logging

- Major difficulty is dealing with APIs
- Each is custom to vendor
- Ideally, log solution natively supports application
- Some vendors support **Salesforce, Office 365**, etc.
- You likely will have SaaS applications without native support
- No support for SaaS application = custom coding

# API Logging Alternatives

- One option is to partner with **professional service** companies
- They design solution to pull logs and send off
- Also is possible to partner with **vendor solution** (can be pricey)
- Example: Sky Formation is a SaaS log agent



# Cloud Log Analysis Ideas

- Some things to consider:
  - Legacy or unauthorized logins
  - Changes to external sharing policies
  - Unauthorized accounts or adding to sensitive groups
  - Large amount of files accessed or large data pulls
  - Standard user behavior analytics
  - Standard application analytics

# MITRE ATT&CK Cloud Matrix

Windows

macOS

Linux

Cloud

AWS

GCP

Azure

Office 365

Azure AD

SaaS

Mobile

ICS

## Cloud Matrix

Below are the tactics and technique representing the MITRE ATT&CK Matrix™ for Enterprise covering cloud-based techniques. The Matrix contains information for the following platforms: AWS, GCP, Azure, Azure AD, Office 365, SaaS.

Last Modified: 2019-10-09 18:48:31.906000

Initial Access	Persistence	Privilege Escalation	Defense Evasion	Credential Access	Discovery	Lateral Movement	Collection	Exfiltration	Impact
Drive-by Compromise	Account Manipulation	Valid Accounts	Application Access Token	Account Manipulation	Account Discovery	Application Access Token	Data from Cloud Storage Object	Transfer Data to Cloud Account	Resource Hijacking
Exploit Public-Facing Application	Create Account		Redundant Access	Brute Force	Cloud Service Dashboard	Internal Spearphishing	Data from Information Repositories		
Spearphishing Link	Implant Container Image		Revert Cloud Instance	Cloud Instance Metadata API	Cloud Service Discovery	Web Session Cookie	Data from Local System		
Trusted Relationship	Office Application Startup		Unused/Unsupported Cloud Regions	Credentials in Files	Network Service Scanning		Data Staged		
Valid Accounts	Redundant Access		Valid Accounts	Steal Application Access Token	Network Share Discovery		Email Collection		
	Valid Accounts		Web Session Cookie	Steal Web Session Cookie	Permission Groups Discovery				
					Remote System Discovery				
					System Information Discovery				
					System Network Connections Discovery				