Google Cloud overview

This overview is designed to help you understand the overall landscape of Google Cloud. Here, you'll take a brief look at some of the commonly used features and get pointers to documentation that can help you go deeper. Knowing what's available and how the parts work together can help you make decisions about how to proceed. You'll also get pointers to some tutorials that you can use to try out Google Cloud in various scenarios.

Google Cloud resources

Google Cloud consists of a set of physical assets, such as computers and hard disk drives, and virtual resources, such as virtual machines (VMs), that are contained in Google's data centers around the globe. Each data center location is in a *region*. Regions are available in Asia, Australia, Europe, North America, and South America. Each region is a collection of *zones*, which are isolated from each other within the region. Each zone is identified by a name that combines a letter identifier with the name of the region. For example, zone a in the East Asia region is named asia-east1-a.

This distribution of resources provides several benefits, including redundancy in case of failure and reduced latency by locating resources closer to clients. This distribution also introduces some rules about how resources can be used together.

Accessing resources through services

In cloud computing, what you might be used to thinking of as software and hardware products, become *services*. These services provide access to the underlying resources. The list of available Google Cloud services is long, and it keeps growing. When you develop your website or application on Google Cloud, you mix and match these services into combinations that provide the infrastructure you need, and then add your code to enable the scenarios you want to build.

Global, regional, and zonal resources

Some resources can be accessed by any other resource, across regions and zones. These *global resources* include preconfigured disk images, disk snapshots, and networks. Some resources can be accessed only by resources that are located in the same region. These *regional resources* include static external IP addresses. Other resources can be accessed only by resources that are located in the same zone. These *zonal resources* include VM instances, their types, and disks.

The following diagram shows the relationship between global scope, regions and zones, and some of their resources:

The scope of an operation varies depending on what kind of resources you're working with. For example, creating a network is a global operation because a network is a global resource, while reserving an IP address is a regional operation because the address is a regional resource.

As you start to optimize your Google Cloud applications, it's important to understand how these regions and zones interact. For example, even if you could, you wouldn't want to attach a disk in one region to a computer in a different region because the latency you'd introduce would make for poor performance. Thankfully, Google Cloud won't let you do that; disks can only be attached to computers in the same zone.

Depending on the level of self-management required for the computing and hosting service you choose, you might or might not need to think about how and where resources are allocated.

For more information about the geographical distribution of Google Cloud, see Geography and Regions.

Projects

Any Google Cloud resources that you allocate and use must belong to a project. You can think of a project as the organizing entity for what you're building. A project is made up of the settings, permissions, and other metadata that describe your applications. Resources within a single project can work together easily, for example by communicating through an internal network, subject to the regions-and-zones rules. A project can't access another project's resources unless you use Shared VPC or VPC Network Peering.

Each Google Cloud project has the following:

- A project name, which you provide.
- A project ID, which you can provide or Google Cloud can provide for you.
- A project number, which Google Cloud provides.

As you work with Google Cloud, you'll use these identifiers in certain command lines and API calls. The following screenshot shows a project name, its ID, and number:

In this example:

- **Example Project** is the project name.
- **example-id** is the project ID.
- **123456789012** is the project number.

Each project ID is unique across Google Cloud. Once you have created a project, you can delete the project but its ID can never be used again.

When billing is enabled, each project is associated with one billing account. Multiple projects can have their resource usage billed to the same account.

A project serves as a namespace. This means every resource within each project must have a unique name, but you can usually reuse resource names if they are in separate projects. Some resource names must be globally unique. Refer to the documentation for the resource for details.

Ways to interact with the services

Google Cloud gives you three basic ways to interact with the services and resources.

Google Cloud Console

The Google Cloud Console provides a web-based, graphical user interface that you can use to manage your Google Cloud projects and resources. When you use the Cloud Console, you create a new project, or choose an existing project, and use the resources that you create in the context of that project. You can create multiple projects, so you can use projects to separate your work in whatever way makes sense for you. For example, you might start a new project if you want to make sure only certain team members can access the resources in that project, while all team members can continue to access resources in another project.

Command-line interface

If you prefer to work at the command line, you can perform most Google Cloud tasks by using the `gcloud` command-line tool. The `gcloud` tool lets you manage development workflow and Google Cloud resources in a terminal window.

For example, you can create a new Compute Engine virtual machine named `example-instance` using a command like the following example:

```
gcloud compute instances create example-instance \
   --image-family=rhel-8 \
   --image-project=rhel-cloud\
   --zone=us-central1-a
```

You can run `gcloud` commands in the following ways:

- You can install the Cloud SDK. The SDK includes the `gcloud` tool, so you can open a terminal window on your own computer and run commands to manage Google Cloud resources.
- You can use Cloud Shell, which is a browser-based shell. Because it runs in a browser window, you don't need to install anything on your own computer. You can open the Cloud Shell from the Google Cloud Console.

Cloud Shell provides the following:

- A temporary Compute Engine virtual machine instance.
- A built-in code editor.
- 5 GB of persistent disk storage.
- Pre-installed Cloud SDK and other tools.

- Language support for Java, Go, Python, Node.js, PHP, Ruby and .NET.
- Web preview functionality.
- Built-in authorization for access to Cloud Console projects and resources.

For a list of `gcloud` commands, see the `gcloud` reference.

For more information about Cloud Shell, see How Cloud Shell works.

Client libraries

The Cloud SDK includes client libraries that enable you to easily create and manage resources. Google Cloud client libraries expose APIs for two main purposes:

- *App APIs* provide access to services. App APIs are optimized for supported languages, such as Node.js and Python. The libraries are designed around service metaphors, so you can work with the services more naturally and write less boilerplate code. The libraries also provide helpers for authentication and authorization.
- *Admin APIs* offer functionality for resource management. For example, you can use admin APIs if you want to build your own automated tools.

You also can use the Google API client libraries to access APIs for products such as Maps, Drive, and YouTube.

Pricing

To browse pricing details for individual services, see the price list.

To estimate your total costs for running a specific workload on Google Cloud, see the pricing calculator.

About Google Cloud services

This overview introduces some of the commonly used Google Cloud services. For the full list of services, see the Products and services page.

This overview covers the following types of services:

- Computing and hosting
- Storage
- Databases
- Networking
- Big data
- Machine learning

Computing and hosting services

Google Cloud gives you options for computing and hosting. You can choose to do the following:

- Work in a serverless environment.
- Use a managed application platform.
- Leverage container technologies to gain lots of flexibility.
- Build your own cloud-based infrastructure to have the most control and flexibility.

You can imagine a spectrum where, at one end, you have most of the responsibilities for resource management and, at the other end, Google has most of those responsibilities.

Serverless computing

Cloud Functions, Google Cloud's *functions as a service* (FaaS) offering, provides a serverless execution environment for building and connecting cloud services. With Cloud Functions you write simple, single-purpose functions that are attached to events emitted from your cloud infrastructure and services. Your function is triggered when an event being watched is fired. Your code executes in a fully managed environment. There is no need to provision any infrastructure or worry about managing any servers.

Cloud Functions can be written using JavaScript, Python 3, Go, or Java. You can take your function and run it in any standard Node.js (Node.js 10), Python 3 (Python 3.7), Go (Go 1.11 or 1.13) or Java (Java 11) environment, which makes both portability and local testing a breeze.

Cloud Functions are a good choice for use cases that include the following:

- Data processing and ETL operations, for scenarios such as video transcoding and IoT streaming data.
- Webhooks to respond to HTTP triggers.
- Lightweight APIs that compose loosely coupled logic into applications.
- Mobile backend functions.

Application platform

App Engine is Google Cloud's *platform as a service* (PaaS). With App Engine, Google handles most of the management of the resources for you. For example, if your application requires more computing resources because traffic to your website increases, Google automatically scales the system to provide those resources. If the system software needs a security update, that's handled for you, too.

When you build your app on App Engine, you can:

- Build your app in Go, Java, .NET, Node.js, PHP, Python, or Ruby and use pre-configured runtimes, or use custom runtimes to write code in any language.
- Let Google manage app hosting, scaling, monitoring, and infrastructure for you.
- Connect with Google Cloud storage products, such as Cloud SQL, Firestore in Datastore mode, and Cloud Storage. You can also connect to managed Redis databases, and host third-party databases such as MongoDB and Cassandra on Compute Engine, another cloud provider, on-premises, or with a third-party vendor.
- Use Web Security Scanner to identify security vulnerabilities as a complement to your existing secure design and development processes.

For a complete list and description of App Engine features, see the App Engine documentation.

Containers

With container-based computing, you can focus on your application code, instead of on deployments and integration into hosting environments. Google Kubernetes Engine (GKE), Google Cloud's *containers as a service* (CaaS) offering, is built on the open source Kubernetes system, which gives you the flexibility of on-premises or hybrid clouds, in addition to Google Cloud's public cloud infrastructure.

When you build with GKE, you can:

- Create and manage groups of Compute Engine instances running Kubernetes, called *clusters*. GKE uses Compute Engine instances as *nodes* in a cluster. Each node runs the Docker runtime, a Kubernetes node agent that monitors the health of the node, and a simple network proxy.
- Declare the requirements for your Docker containers by creating a simple JSON configuration file.
- Use Container Registry for secure, private storage of Docker images. You can push images to your registry and then you can pull images to any Compute Engine instance or your own hardware by using an HTTP endpoint.
- Create single- and multi-container *pods*. Each pod represents a logical host that can contain one or more containers. Containers in a pod work together by sharing resources, such as networking resources. Together, a set of pods might comprise an entire application, a microservice, or one layer in a multi-tier application.
- Create and manage *replication controllers*, which manage the creation and deletion of pod replicas based on a template. Replication controllers help to ensure that your application has the resources it needs to run reliably and scale appropriately.
- Create and manage *services*. Services create an abstraction layer that decouples frontend clients from pods that provide backend functions. In this way, clients can work without concerns about which pods are being created and deleted at any given moment.
- Create an external network load balancer.

Virtual machines

Google Cloud's unmanaged compute service is Compute Engine. You can think of Compute Engine as providing an *infrastructure as a service* (IaaS), because the system provides a robust computing infrastructure, but you must choose and configure the platform components that you want to use. With Compute Engine, it's your responsibility to configure, administer, and monitor the systems. Google will ensure that resources are available, reliable, and ready for you to use, but it's up to you to provision and manage them. The advantage here is that you have complete control of the systems and unlimited flexibility.

When you build on Compute Engine, you can do the following:

- Use virtual machines (VMs), called *instances*, to build your application, much like you would if you had your own hardware infrastructure. You can choose from a variety of instance types to customize your configuration to meet your needs and your budget.
- Choose which global regions and zones to deploy your resources in, giving you control over where your data is stored and used.
- Choose which operating systems, development stacks, languages, frameworks, services, and other software technologies you prefer.
- Create instances from public or private images.
- Use Google Cloud storage technologies or any third-party technologies you prefer.
- Use Google Cloud Marketplace to quickly deploy pre-configured software packages. For example, you can deploy a LAMP or MEAN stack with just a few clicks.
- Create instance groups to more easily manage multiple instances together.
- Use autoscaling with an instance group to automatically add and remove capacity.
- Attach and detach disks as needed.
- Use SSH to connect directly to your instances.

Combining computing and hosting options

You don't have to stick with just one type of computing service. For example, you can combine App Engine and Compute Engine to take advantage of the features and benefits of each. For an example of using both App Engine and Compute Engine, see Reliable task scheduling on Compute Engine.

For a detailed look at options for serving websites, see Serving websites.

Storage services

Whatever your application, you'll probably need to store some media files, backups, or other file-like objects. Google Cloud provides a variety of storage services, including:

- Consistent, scalable, large-capacity data storage in Cloud Storage. Cloud Storage comes in several flavors:
- Standard Cloud Storage provides maximum availability.
- Cloud Storage Nearline provides low-cost archival storage ideal for data accessed less than once a month.

- Cloud Storage Coldline provides even lower-cost archival storage ideal for data accessed less than once a quarter.
- Cloud Storage Archive provides the lowest-cost archival storage for backup and disaster recovery ideal for data you intend to access less than once a year.
- Persistent disks on Compute Engine, for use as primary storage for your instances. Compute Engine offers both hard-disk-based persistent disks, called *standard persistent disks*, and solid-state persistent disks (SSD).
- Fully managed NFS file servers in Filestore. You can use Filestore instances to store data from applications running on Compute Engine VM instances or GKE clusters.

To understand the full range and benefits of storage services on Google Cloud, learn more about our storage options.

Database services

Google Cloud provides a variety of SQL and NoSQL database services:

- A SQL database in Cloud SQL, which provides either MySQL or PostgreSQL databases.
- A fully managed, mission-critical, relational database service in Cloud Spanner that offers transactional consistency at global scale, schemas, SQL querying, and automatic, synchronous replication for high availability.
- Two options for NoSQL data storage: Firestore, for document-like data, and Cloud Bigtable, for tabular data.

You can also choose to set up your preferred database technology on Compute Engine by using persistent disks. For example, you can set up MongoDB for NoSQL document storage.

To find out about the differences between our database services, read more about Google Cloud databases.

Networking services

While App Engine manages networking for you, and GKE uses the Kubernetes model, Compute Engine provides a set of networking services. These services help you to load-balance traffic across resources, create DNS records, and connect your existing network to Google's network.

Networks, firewalls, and routes

Virtual Private Cloud (VPC) provides a set of networking services that your VM instances use. An instance can have more than one interface, but each interface must be connected to a different network. Every VPC project has a *default network*. You can create additional networks in your project, but networks cannot be shared between projects.

Firewall rules govern traffic coming into instances on a network. The default network has a default set of firewall rules, and you can create custom rules, too.

A *route* lets you implement more advanced networking functions in your instances, such as creating VPNs. A route specifies how packets leaving an instance should be directed. For example, a route might specify that packets destined for a particular network range should be handled by a gateway virtual machine instance that you configure and operate.

Load balancing

If your website or application is running on Compute Engine, the time might come when you're ready to distribute the workload across multiple instances. Server-side load balancing features provide you with the following options:

- Network load balancing lets you distribute traffic among server instances in the same region based on incoming IP protocol data, such as address, port, and protocol. Network load balancing is a great solution if, for example, you want to meet the demands of increasing traffic to your website.
- HTTP(S) load balancing enables you to distribute traffic across regions so you can ensure that requests are routed to the closest region or, in the event of a failure or over-capacity limitations, to a healthy instance in the next closest region. You can also use HTTP(S) load balancing to distribute traffic based on content type. For example, you might set up your servers to deliver static content, such as images and CSS, from one server and dynamic content, such as PHP pages, from a different server. The load balancer can direct each request to the server that provides each content type.

Cloud DNS

You can publish and maintain Domain Name System (DNS) records by using the same infrastructure that Google uses. You can use the Google Cloud Console, the command line, or a REST API to work with managed zones and DNS records.

Advanced connectivity

If you have an existing network that you want to connect to Google Cloud resources, Google Cloud offers the following options for advanced connectivity:

- Cloud Interconnect enables you to connect your existing network to your VPC network through a highly available, low-latency, enterprise-grade connection. You can use Dedicated Interconnect to connect directly to Google, or use Partner Interconnect to connect to Google through a supported service provider.
- Cloud VPN enables you to connect your existing network to your VPC network through an IPsec connection. You can also use VPN to connect two Cloud VPN gateways to each other.
- Direct Peering enables you to exchange internet traffic between your business network and Google at one of Google's broad-reaching edge network locations. See Google's peering site for more information about edge locations.
- Carrier Peering enables you to connect your infrastructure to Google's network edge through highly available, lower-latency connections by using service providers. You can also

extend your private network into your private Virtual Private Cloud network over Carrier Peering links by using a VPN tunnel between the networks.

Big data services

Big data services enable you to process and query big data in the cloud to get fast answers to complicated questions.

Data analysis

BigQuery provides data analysis services. With BigQuery, you can:

- Create custom schemas that organize your data into datasets and tables.
- Load data from a variety of sources, including streaming data.
- Use SQL-like commands to query massive datasets very quickly. BigQuery is designed and optimized for speed.
- Use the web UI, command-line interface, or API.
- Load, query, export, and copy data by using jobs.
- Manage data and protect it by using permissions.

To try out BigQuery quickly and easily, try using the Web UI Quickstart to query a public dataset.

Batch and streaming data processing

Dataflow provides a managed service and set of SDKs that you can use to perform batch and streaming data processing tasks. Dataflow works well for high-volume computation, especially when the processing tasks can clearly and easily be divided into parallel workloads. Dataflow is also great for extract-transform-load (ETL) tasks, which are useful for moving data between different storage media, transforming data into a more desirable format, or loading data onto a new storage system.

Asynchronous messaging

Pub/Sub is an asynchronous messaging service. Your application can send messages as JSON data structures to a publishing unit called a *topic*. Because Pub/Sub topics are a global resource, other applications in projects that you own can subscribe to the topic to receive the messages in HTTP request or response bodies. To familiarize yourself with Pub/Sub, see the Pub/Sub quickstart.

Pub/Sub's usefulness isn't confined to big data. You can use Pub/Sub in many circumstances where you need an asynchronous messaging service. For an example that uses Pub/Sub to

coordinate App Engine and Compute Engine, see Reliable task scheduling on Compute Engine.

Machine learning services

AI Platform offers a variety of powerful machine learning (ML) services. You can choose to use APIs that provide pre-trained models optimized for specific applications, or build and train your own large-scale, sophisticated models using a managed TensorFlow framework.

Machine learning APIs

Google Cloud offers a variety of APIs that enable you to take advantage of Google's ML without creating and training your own models.

- Video Intelligence API lets you use video analysis technology that provides label detection, explicit content detection, shot-change detection, and regionalization features.
- Speech-to-Text lets you convert audio to text, recognizing over 110 languages and variants, to support your global user base. You can transcribe the text of users dictating to an application's microphone, enable command-and-control through voice, or transcribe audio files, among other use cases.
- Cloud Vision lets you easily integrate vision detection features, including image labeling, face and landmark detection, optical character recognition (OCR), and tagging of explicit content.
- Cloud Natural Language API lets you add sentiment analysis, entity analysis, entity-sentiment analysis, content classification, and syntax analysis.
- Cloud Translation lets you quickly translate source text into any of over a hundred supported languages. Language detection helps out in cases where the source language is not known.
- Dialogflow lets you build conversational interfaces for websites, mobile applications, popular messaging platforms, and IoT devices. You can use it to build interfaces, such as chatbots, that are capable of natural and rich interactions with humans.

AI Platform

AI Platform combines the managed infrastructure of Google Cloud with the power and flexibility of TensorFlow. You can use it to train your machine learning models at scale, and to host trained models to make predictions about new data in the cloud.

AI Platform enables you to train machine learning models by running TensorFlow training applications on Google Cloud, and hosts those trained models for you, so you can use them to get predictions about new data. AI Platform manages the computing resources that your training job needs to run, so you can focus more on your model than on hardware configuration or resource management.

COMPUTE PRODUCTS :

1) **Google Cloud Compute Engine**

- Compute Engine is described as virtual machines, disks, network. This is the most relatable Google Cloud product.

- Have a huge job? Maybe spin up dozens, hundreds, or even thousands of Compute Engine instances at the same time to get the job done faster, then stop paying for them when the job's done.

- *Analogy* : This is like the workhorse two by four brick of the Lego world. You can build almost anything with it if you have the time and you're okay with having a few rough edges. But in terms of building systems, this is a relatively small brick where you have to do a lot of work yourself to use it.

2) **Google Cloud Functions**

- Cloud Functions, just a little way down, is described as event-driven serverless functions. Cloud Functions is built even finer grained than Compute Engine is.

- This is by the 10th of a second, but it manages all of the scaling for you automatically.

- This is Google Cloud's “functions as a service product”, or FaaS, and just like Amazon's Lambda, this is a key building block of serverless systems.

- With this, instead of running server programs that sit around and wait for clients to connect to them, in Cloud Functions you set up an event and Google will run the code that you've given it every time that event happens.

- If no events are happening, you're not paying anything.

- And if a whole bunch of these events happen all at the same time, then Google just runs your code in parallel lots of times. But you don't even really have to think about that. Google just does it automatically.

- Use Case Eg : Now there are all sorts of events that you can hook up. They could be requests through HTTP, like connections being made from your web page or other micro services or something like that, or the event could be a new object being uploaded to Cloud Storage or a message being sent to Cloud Pub/Sub, or any number of other things, really. There are lots of different event types.

- *Analogy* : Now this isn't a perfect comparison, but if we were looking for something in the Lego world that would be like this, maybe it would be like a thin one by six brick. You can build lots of things with it, but it's particularly good at connecting all sorts of things together.

3) **Google Kubernetes Engine (GKE)**

- GKE, or Kubernetes Engine, can help alleviate some of the management burden that you would otherwise have to deal with with Compute Engine.

- GKE will install, run, and manage Kubernetes for you on clusters of Compute Engine instances.

- It also reaches out to the networking world and creates things like Cloud load balancers, which helps reduce even further the amount of stuff that you would have to do.

STORAGE PRODUCTS

1) Google Cloud Storage (GCS)

- Now, speaking of Cloud Storage, just like Cloud Functions, Cloud Storage is also a serverless product and they work very nicely together.

- Cloud Storage is like the most pure service at its core. It says give me some data, and I can give it back to you. No fuss, no muss, and incredibly reliable.

- You never really change any data it holds but you can give it some new data and tell it to forget about the old stuff. But unless you tell it to, it's really, really good at not losing your data.

- And it's also amazing at handling however much load you slam it with without even breaking a sweat. That's why it's described as object storage and serving.

- Nearline and Coldline are really just versions of Cloud Storage that are priced and optimized for less frequent data access patterns, so they're perfect for backups. That's why their descriptions are archival occasional access storage and archival rare access storage.

2) Persistent Disk

- Persistent Disk is a special kind of storage, and its description gives that away - VM-attached disks.

- Now, the Cloud Storage product is object storage, but Persistent Disk is block storage instead, so it works much like a hard drive does.

- And it only connects to Compute Engine instances.

- *Analogy* : In a way, Persistent Disk is kind of like the Lego pieces with a different kind of connector on the side. It only goes together with Compute Engine. They were made for each other.

- You don't usually even think too much about the Persistent Disks. They're just one aspect of the Compute Engine instance, but if you've decided to take on the burden of managing data storage yourself, which I don't generally recommend by the way, then you'll get more familiar with this product and some of the cool things that it can do beyond what a normal hard drive could do.

3) Cloud Firestore

- Cloud Filestore sits between Cloud Storage and Persistent Disk, and it is neither object-based, nor block-based. Cloud File Store is file-based.

- Being a managed NFS server, it's a bit more flexible to connect to than Persistent Disk and may work better with certain kinds of applications than the object-based Cloud Storage.

AI / ML PRODUCTS

1) **TensorFlow Processing Unit (TPU)**

- Cloud TPU is a specialized hardware for machine learning. So it's kind of like a Compute Engine instance, but this one is purpose-built for tensor flow processing instead of for general purpose processing.

- Now, if you're not familiar with it, TensorFlow is an open source platform for doing machine learning stuff. It's quite broadly popular, but it's actually another one of the things that Google originally created and then released as open source.

2) **Cloud Speech-To-Text** - will convert audio to text.

3) **Cloud Text-To-Speech** - will convert text to audio.

4) **Cloud Natural Language** - is for text parsing and analysis so you can give it some text and then it'll tell you both the parts of speech and the emotions that are represented in it, stuff like that.

5) **Cloud Machine Learning Engine** - which is a managed platform for machine learning, where it will run TensorFlow for you.

6) **Cloud Deep Learning VM Image** - is a pre-configured virtual machine for deep learning. So you would run that image as a Compute Engine instance.

7) **Cloud AutoML products** - where the basic machine learning models are set up but you can train them on your organization's own set of data, whether those be text documents, or images, or translations, or what have you.

DATABASE PRODUCTS

1) **Cloud SQL**

Managed MySQL and PostgreSQL. Here, no need to manage the Compute Engine instances and persistent disks yourself and installing MySQL, upgrading it, applying patches, all that good stuff; here, Google can do all of that for youand can even manage things like read replicas.

2) **Cloud Spanner**

A horizontally scalable relational database. This thing is seriously massive scale, and it lets you keep all of the things you love about relational databases while still scaling well beyond what you would ever conCIDR normal for that.

3) **Cloud Firestore**

- One of the Firebase family of products that shows up in the Mobile Products section.

- It's a strongly consistent serverless document database and it has some really cool things built in, like web socket connections for updates.

4) Cloud Data Store

- A horizontally scalable document database, and one of the nice things about Cloud Data Store is that you just pay for what you use.
- You don't have to provision and manage a certain capacity.

5) Cloud BigTable

If you do have a more predictable and high volume, then Cloud Bigtable might be a better choice, since it is a petabyte-scale, low-latency, nonrelational database.

DATA AND ANALYTICS SECTION

1) Dataproc

- Dataproc is managed Spark and Hadoop. So if you're already running Hadoop work somewhere else, Cloud Dataproc could be a good way to move that work on to the Google Cloud.
- This is for processing large amounts of data, and it uses Compute Engine under the hood, just like Kubernetes Engine does.

2) Dataflow

- Cloud Dataflow also processes large amounts of data, but this is newer and better than Cloud Dataproc, and it handles both stream and batch data processing.
- The key technology behind Cloud Dataflow is something that Google released as the open source Apache Beam.

3) Genomics

- Google Genomics is very specialized, and it's only really useful if you're working with that kind of medical data.
- *Analogy* : Comparing this to Lego pieces, this is kind of like a helicopter rotor Lego piece. It's not very general purpose, but it's very useful if you have that particular need.

4) Cloud Pub/Sub

- Cloud Pub/Sub is probably the most flexible product in all of Google Cloud.
- This is described as global real-time messaging, and it can be used to connect almost anything to almost anything else.
- In Cloud Pub/Sub, you create topics, and then you have one part of your system

- publish messages to that topic, and another part of your system subscribe to messages from that topic.

- So for example, you might have a Cloud Function that creates some data and publishes it, and then you might have that same data flow out to both Cloud Dataflow, and BigQuery.

5) **BigQuery**

- Crown jewel in Google Cloud.

- It's described as data warehouse/analytics, but I think that undersells it quite a lot.

- BigQuery is serverless, in that when you're not using it, you're not paying for it. And then when you slam it with a huge complicated SQL query, it just scales up automatically and handles it without a problem.

- You can store tons of data in BigQuery and still get incredibly fast responses to your queries.

NETWORKING SECTION

1) **Virtual Private Cloud (VPC)**

It's the umbrella for the software-defined networking, and most of the other networking services hang off of that one.

2) **Dedicated Interconnect** - to have a dedicated private network connection between that VPC and your external data center.

3) **Cloud NAT** - to have network address translation to connect out from that VPC

4) **Cloud Load Balancing**

- Deals with multi-region load distribution into that global Virtual Private Cloud network.

- Cloud Load Balancing is pretty special, especially when you combine it with a premium network service tier. That can let you have your clients who are connecting from all over the world connect to one single IP address and still have all of those connections be made to whatever server inside of your VPC is actually physically closest to them.

- That means that your clients can have much faster connections, and faster connections makes for happier clients.

- It also handles situations where there isn't something close by to serve that request because it's failed or something like that, and it can just reroute that inside of Google's global network to some server that actually can still handle that request. But this is all transparent to the client.

- So fail overs like that can happen almost instantaneously.

5) **Cloud Armor** - is distributed denial of service protection and a web application firewall.

6) **Cloud CDN** - is a content delivery network that can cache data much closer to where users are physically located.

7) **Cloud DNS**

- Lets you take advantage of Google's programmable DNS serving network where their domain name system has a 100% uptime guarantee.

- When your users are connecting to mycoolwebsite.com, or whatever it is, you definitely want that name translation to succeed. Otherwise, they won't be able to connect to your service at all.

MANAGEMENT TOOLS SECTION

Stackdriver is an important family of services.

1) **Cloud Monitoring** – is infrastructure and application monitoring, so you can watch metrics of what's going on in your system.

2) **Cloud Logging**

- Centralized logging, so that all the different parts of your system, whether they be micro services, or just multiple instances, or completely different parts of your system, they can all log to one central place.

- Centralized logging is really important to running a system, because if you have to go out and try and gather up log information from every individual piece of your system just to make sense of what's been happening, you're going to get so bogged down in that overhead, that you won't actually be able to make any progress.

IDENTITY AND SECURITY SECTION

1) **Cloud Identity** - which, if you're not using G Suite, is where you manage users, devices, and apps

2) **Cloud IAM**

- Stands for identity and access management, is the resource access control part of things.

- Cloud IAM ties everything together from a security perspective. That's where you define who can do what to which things.

3) **Cloud HSM** - is the hardware security module service. You can use that to manage your encryption keys and certificates,

4) **Cloud Data Loss Prevention API** - is actually a machine learning service that can classify and redact sensitive data from what you pointed at.

---

gcloud command-line tool overview

This page contains an overview of the gcloud command-line tool and its common command patterns and capabilities.

What is the gcloud command-line tool?

The gcloud command-line interface is the primary CLI tool to create and manage Google Cloud resources. You can use this tool to perform many common platform tasks either from the command line or in scripts and other automations.

For example, you can use the gcloud CLI to create and manage:

- Google Compute Engine virtual machine instances and other resources
- Google Cloud SQL instances
- Google Kubernetes Engine clusters
- Google Cloud Dataproc clusters and jobs
- Google Cloud DNS managed zones and record sets
- Google Cloud Deployment manager deployments

You can also use the gcloud CLI to deploy App Engine applications, manage authentication, customize local configuration, and perform other tasks. Read the gcloud CLI reference to learn more about the capabilities of this tool.

The gcloud command-line tool cheat sheet

For a quick introduction to the gcloud command-line tool, a list of commonly used commands, and a look at how these commands are structured, refer to the gcloud command-line tool cheat sheet.

The gcloud CLI and Cloud SDK

The gcloud CLI is a part of the Google Cloud SDK. You must download and install the SDK on your system and initialize it before you can use the gcloud command-line tool.

By default, the SDK installs those gcloud CLI commands that are at the General Availability level only. Additional functionality is available in SDK components named alpha and beta. These components allow you to use the gcloud CLI to work with Google Cloud Bigtable, Google Cloud Dataflow and other parts of the Cloud Platform at earlier release levels than General Availability.

The gcloud CLI releases have the same version number as the SDK. The current SDK version is 319.0.0. You can download and install previous versions of the SDK from the download archive.

Note: The gcloud command-line tool is available automatically in Google Cloud Shell. If you are using Cloud Shell, you do not need to install the gcloud CLI manually in order to use it.

Downloading the gcloud command-line tool

You can download the latest version of Cloud SDK, which includes the gcloud command-line tool, from the download page.

Release levels

The gcloud CLI commands have the following release levels:

Release level	Label	Description
General Availability	None	Commands are conCIDRed fully stable and available for production use. Advance warnings will be made for commands that break current functionality and documented in the release notes.
Beta	beta	Commands are functionally complete, but may still have some outstanding issues. Breaking changes to these commands may be made without notice.
Alpha	alpha	Commands are in early release and may change without notice.

The alpha and beta components are not installed by default when you install the SDK. You must install these components separately using the gcloud components install command. If you try to run an alpha or beta command and the corresponding component is not installed, the gcloud command-line tool will prompt you to install it.

Command groups

Within each release level, gcloud CLI commands are organized into a nested hierarchy of command groups, each of which represents a product or feature of the Cloud Platform or its functional subgroups.

For example:

Command group	Description
gcloud compute	Commands related to Compute Engine in general availability
gcloud compute instances	Commands related to Compute Engine instances in general availability
gcloud beta compute	Commands related to Compute Engine in Beta
gcloud alpha app	Commands related to managing App Engine deployments in Alpha

Running gcloud CLI commands

You can run gcloud CLI commands from the command line in the same way you use other command-line tools. You can also run gcloud CLI commands from within scripts and other automations, for example, when using Jenkins to automate Cloud Platform tasks.

Note: gcloud CLI reference documentation and examples use backslashes, ****, to denote long commands. You can execute these commands as-is (Windows users can use **^** instead of ****). If you'd like to remove the backslashes, be sure to remove newlines as well to ensure the command is read as a single line.

Properties

gcloud CLI properties are settings that affect the behavior of the gcloud CLI and other Cloud SDK tools. Some of these properties can be set by either global or command flags - in which case, the value set by the flag takes precedence.

Enabling accessibility features

For a more streamlined screen reader experience, the gcloud command-line tool comes with an accessibility/screen_reader property.

To enable this property, run:

```
gcloud config set accessibility/screen_reader true
```

For more details about the accessibility features that come with the gcloud command-line tool, refer to the Enabling accessibility features guide.

Configurations

A configuration is a named set of gcloud CLI properties. It works like a *profile*, essentially.

Starting off with Cloud SDK, you'll work with a single configuration named default and you can set properties by running either gcloud init or gcloud config set. This single default configuration is suitable for most use cases.

If you'd like to work with multiple projects or authorization accounts, you can set up multiple configurations with gcloud config configurations create and switch among them accordingly. Within these configurations, you can customize properties. For example, to set your project within an active configuration use the project property: gcloud config set project <project-id>.

For a detailed account of these concepts, see the Configurations guide.

Global flags

The gcloud CLI provides a set of gcloud CLI-wide flags that govern the behavior of commands on a per-invocation level. Flags override any values set in SDK properties.

Positional Arguments and Flags

While both positional arguments and flags affect the output of a gcloud CLI command, there is a subtle difference in their use cases. A positional argument is used to define an entity on which a command operates while a flag is required to set a variation in a command's behaviour.

Use of stdout and stderr

Successful output of gcloud CLI commands is written to stdout. All other types of responses - prompts, warnings, and errors - are written to stderr. Note that anything written to stderr is not stable and should not be scripted against.

For a definitive list of guidelines on handling output, refer to the Scripting guide.

Prompting

To protect against unintended destructive actions, the gcloud CLI will confirm your intentions before executing commands such as `gcloud projects delete`.

You can also expect prompts if you were to create a Google Compute Engine virtual machine instance, say 'test-instance', using `gcloud compute instances create test-instance`. You will be asked to choose a zone to create the instance in.

To disable prompting, use the `--quiet` flag.

Note, the wording of prompts can change and should not be scripted against.

Suppressing prompting, writing to the terminal, and logging

The `--quiet` flag (also, `-q`) for the gcloud CLI disables all interactive prompts when running gcloud CLI commands and comes in handy when scripting. In the event input is needed, defaults will be used. If there aren't any, an error will be raised.

To suppress printing of command output to standard output and standard error in the terminal, use the `--no-user-output-enabled` flag.

To adjust verbosity of logs instead, use the `--verbosity` flag with an appropriate level (debug, info, warning, error, critical, or none).

Determining output structure

By default, when a gcloud CLI command returns a list of resources, they are pretty-printed to standard output. To produce more meaningful output, the format, filter and projection flags allow you to finetune your output.

If you'd like to define just the format of your output, use the `--format` flag to produce a tabulated or flattened version of your output (for interactive display) or a machine-readable version of the output (`json`, `csv`, `yaml`, `value`).

To format a list of keys that select resource data values, use projections. To further refine your output to criteria you define, use filter.

If you'd like to work through a quick interactive tutorial to help get you familiar with filter and format functionality, follow the link below.

The gcloud command-line tool cheat sheet

The gcloud **cheat sheet**

A roster of go-to gcloud commands for the gcloud tool, Google Cloud's primary command-line tool.

Also included: introductory primer, understanding commands, and a printable PDF).

Cheat sheet

Getting started

Get going with the gcloud command-line tool.

- gcloud init: Initialize, authorize, and configure the gcloud tool.
- gcloud version: Display version and installed components.
- gcloud components install: Install specific components.
- gcloud components update: Update your Cloud SDK to the latest version.
- gcloud config set project: Set a default Google Cloud project to work on.
- gcloud info: Display current gcloud tool environment details.

Help

Cloud SDK is happy to help.

- gcloud help: Search the gcloud tool reference documents for specific terms.
- gcloud feedback: Provide feedback for the Cloud SDK team.
- gcloud topic: Supplementary help material for non-command topics like accessibility, filtering, and formatting.

Personalization

Make the Cloud SDK your own; personalize your configuration with properties.

- gcloud config set: Define a property (like compute/zone) for the current configuration.
- gcloud config get-value: Fetch value of a Cloud SDK property.

- `gcloud config list`: Display all the properties for the current configuration.
- `gcloud config configurations create`: Create a new named configuration.
- `gcloud config configurations list`: Display a list of all available configurations.
- `gcloud config configurations activate`: Switch to an existing named configuration.

Credentials

Grant and revoke authorization to Cloud SDK

- `gcloud auth login`: Authorize Google Cloud access for the `gcloud` tool with Google user credentials and set current account as active.
- `gcloud auth activate-service-account`: Like `gcloud auth login` but with service account credentials.
- `gcloud auth list`: List all credentialed accounts.
- `gcloud auth print-access-token`: Display the current account's access token.
- `gcloud auth revoke`: Remove access credentials for an account.

Projects

Manage project access policies

- `gcloud projects describe`: Display metadata for a project (including its ID).
- `gcloud projects add-iam-policy-binding`: Add an IAM policy binding to a specified project.

Identity & Access Management

Configuring Cloud Identity & Access Management (IAM) preferences and service accounts

- `gcloud iam list-grantable-roles`: List IAM grantable roles for a resource.
- `gcloud iam roles create`: Create a custom role for a project or org.
- `gcloud iam service-accounts create`: Create a service account for a project.
- `gcloud iam service-accounts add-iam-policy-binding`: Add an IAM policy binding to a service account.
- `gcloud iam service-accounts set-iam-policy-binding`: Replace existing IAM policy binding.
- `gcloud iam service-accounts keys list`: List a service account's keys.

Docker & Google Kubernetes Engine (GKE)

Manage containerized applications on Kubernetes

- gcloud auth configure-docker: Register the gcloud tool as a Docker credential helper.
- gcloud container clusters create: Create a cluster to run GKE containers.
- gcloud container clusters list: List clusters for running GKE containers.
- gcloud container clusters get-credentials: Update kubeconfig to get kubectl to use a GKE cluster.
- gcloud container images list-tags: List tag and digest metadata for a container image.

Virtual Machines & Compute Engine

Create, run, and manage VMs on Google infrastructure

- gcloud compute zones list: List Compute Engine zones.
- gcloud compute instances describe: Display a VM instance's details.
- gcloud compute instances list: List all VM instances in a project.
- gcloud compute disks snapshot: Create snapshot of persistent disks.
- gcloud compute snapshots describe: Display a snapshot's details.
- gcloud compute snapshots delete: Delete a snapshot.
- gcloud compute ssh: Connect to a VM instance by using SSH.

Serverless & App Engine

Build highly scalable applications on a fully managed serverless platform

- gcloud app deploy: Deploy your app's code and configuration to the App Engine server.
- gcloud app versions list: List all versions of all services deployed to the App Engine server.
- gcloud app browse: Open the current app in a web browser.
- gcloud app create: Create an App Engine app within your current project.
- gcloud app logs read: Display the latest App Engine app logs.

Miscellaneous

Commands that might come in handy

- gcloud kms decrypt: Decrypt ciphertext (to a plaintext file) using a Cloud Key Management Service (Cloud KMS) key.

- `gcloud logging logs list`: List your project's logs.
- `gcloud sql backups describe`: Display info about a Cloud SQL instance backup.
- `gcloud sql export sql`: Export data from a Cloud SQL instance to a SQL file.

Introductory primer

A quick primer for getting started with the gcloud command-line tool.

Installing the Cloud SDK

Install the Cloud SDK with these installation instructions.

Flags, arguments, and other wondrous additions

Arguments can be Positional args or Flags

- **Positional args:** Set after command name; must respect order of positional args.
- **Flags:** Set after positional args; order of flags doesn't matter.

A flag can be either a:

- *Name-value pair* (`--foo=bar`), or
- *Boolean* (`--force/no-force`).

Additionally, flags can either be:

- *Required*
- *Optional:* in which case, the default value is used, if the flag is not defined

Global flags

Some flags are available throughout the gcloud command-line tool experience, like:

- `--help`: For when in doubt; display detailed help for a command.
- `--project`: If using a project other than the current one.
- `--quiet`: Disabling interactive prompting (and applying default values for inputs).
- `--verbosity`: Can set verbosity levels at `debug`, `info`, `warning`, `error`, `critical`, and `none`.
- `--version`: Display `gcloud` version information.
- `--format`: Set output format
  as `config`, `csv`, `default`, `diff`, `disable`, `flattened`, `get`, `json`, `list`, `multi`, `none`, `object`, `table`, `text`, `v alue`, or `yaml`.

Cleaning up results

Extricate the most from your output with the filter, format, limit, and sort-by flags.

For Compute Engine instances with prefix `us` and not machine type `f1-micro`:

```
gcloud compute instances list --filter="zone ~ ^us AND -machineType:f1-micro"
```

For a list of projects created on or after 15 January 2018, sorted from oldest to newest, presented as a table with project number, project id and creation time columns with dates and times in local timezone:

```
gcloud projects list --format="table(projectNumber,projectId,createTime.date(tz=LOCAL))"
--filter="createTime>=2018-01-15T12:00:00" --sort-by=createTime
```

For a list of ten Compute Engine instances with a label `my-label` (of any value):

```
gcloud compute instances list --filter="labels.my-label:*" --limit=10
```

Understanding commands

The underlying patterns for `gcloud` commands; to aid self-discovery of commands.

Finding gcloud commands

The gcloud command-line tool is a tree; non-leaf nodes are command groups and leaf nodes are commands. (Also, tab completion works for commands and resources!)

Most gcloud commands follow the following format:

```
gcloud + release level (optional) + component + entity + operation + positional args + flags
```

For example: `gcloud + compute + instances + create + example-instance-1 + --zone=us-central1-a`

Release level

Release Level refers to the command's release status.

Example: `alpha` for alpha commands, `beta` for beta commands, no release level needed for GA commands.

Component

Component refers to the different Google Cloud services.

Example: compute for Compute Engine, app for App Engine, etc.

Entity

Entity refers to the plural form of an element or collection of elements under a component.

Example: disks, firewalls, images, instances, regions, zones for compute

Operation

Operation refers to the imperative verb form of the operation to be performed on the entity.

Example: Common operations
are describe, list, create/update, delete/clear, import, export, copy, remove, add, reset, rest art, restore, run, and deploy.

Positional args

Positional args refer to the required, order-specific arguments needed to execute the command.

Example: <INSTANCE_NAMES> is the required positional argument for gcloud compute instances create.

Flags

Flags refer to the additional arguments, --flag-name(=value), passed in to the command after positional args.

Example: --machine-type=<MACHINE_TYPE> and --preemptible are optional flags for gcloud compute instances create.

Google Cloud Billing Account

A Cloud Billing account defines who pays for a given set of Google Cloud resources. To use Google Cloud services, you must have a valid Cloud Billing account, and must link it to your Google Cloud projects. Your project's Google Cloud usage is charged to the linked Cloud Billing account.

You must have a valid Cloud Billing account even if you are in your free trial period or if you only use Google Cloud resources that are covered by the Google Cloud Free Tier.

You also need a Cloud Billing account to pay for your use of the Google Maps Platform APIs.

Interacting with GCP

There are four ways you can interact with Google Cloud Platform :

- Console,
- SDK and Cloud Shell
- Mobile App
- APIs.

1) The GCP Console and Cloud Shell

i. A web-based administrative interface.
ii. If you build an application in GCP, you'll use it.
iii. Although, the end users of your application won't.
iv. It lets you view and manage all your projects and all the resources they use.
v. It also lets you enable, disable and explore the APIs of GCP services.
vi. And it gives you access to Cloud Shell. That's a command-line interface to GCP that's easily accessed from your browser. From Cloud Shell, you can use the tools provided by the Google Cloud Software Development kit SDK, without having to first install them somewhere.

2) Software Development Kit

i. The Google Cloud SDK is a set of tools that you can use to manage your resources and your applications on GCP.
ii. These include the gcloud tool, which provides the main command line interface for Google Cloud Platform products and services.
iii. There's also gsutil which is for Google Cloud Storage and bq which is for BigQuery.
iv. The easiest way to get to the SDK commands is to click the Cloud Shell button on a GCP Console. You get a command line in your web browser on a virtual machine with all these commands already installed.
v. You can also install the SDK on your own computers - your laptop, your on-premise servers of virtual machines and other clouds.
vi. The SDK is also available as a docker image, which is a really easy and clean way to work with it.

3) APIs

i. The services that make up GCP offer application programming interfaces so that the code you write can control them.
ii. These APIs are what's called RESTful. In other words they follow the representational state transfer paradigm. Basically, it means that your code can use Google services in much the same way that web browsers talk to web servers.
iii. The APIs name resources and GCP with URLs. Your code can pass information to the APIs using JSON, which is a very popular way of passing textual information over the web.
iv. And there's an open system for user log in and access control.

v. The GCP Console lets you turn on and off APIs. Many APIs are off by default, and many are associated with quotas and limits. These restrictions help protect you from using resources inadvertently. You can enable only those APIs you need and you can
vi. request increases in quotas when you need more resources.
vii. *For example :* if you're writing an application that needs to control GCP resources, you'll need to get your use of the APIs just right. And to do that, you'll use APIs Explorer.
viii. **APIs Explorer**
ix. The GCP Console includes a tool called the APIs Explorer that helps you learn about the APIs interactively.
x. With APIs Explorer you can :
    a. Browse quickly through available APIs and versions
    b. See methods available for each API and what parameters they support along with inline documentation
    c. Execute requests for any method and see responses in real time
    d. Easily make authenticated and authorized API calls
xi. Suppose you have explored an API and you're ready to build an application that uses it. We do not need to start coding from scratch. Google provides client libraries that take a lot of the drudgery out of the task of calling GCP from your code.
xii. There are two kinds of libraries.
    a. The **Cloud Client Libraries** are Google clouds latest and recommended libraries for its APIs. They adopt the native styles and idioms of each language. On the other hand, sometimes a Cloud Client Library doesn't support the newest services and features.
    b. In that case, you can use the **Google API Client Library** for your desired languages. These libraries are designed for generality and completeness.

4) Mobile App

i. For Android and iOS that lets you examine and manage the resources you're using in GCP.
ii. Manage VMs and db instances
iii. Manage apps in Google App Engine (GAE)
iv. Manage your billing
v. Visualize your projects with a customizable dashboard

Cloud Marketplace (formerly known as Cloud Launcher)

i. Get started with GCP with minimal effort.
ii. It's a tool for quickly deploying functional software packages on Google Cloud Platform.
iii. There's no need to manually configure the software, virtual machine instances, storage or network settings. Although, you can modify many of them before you launch if you like.
iv. Most software packages in Cloud Launcher are at no additional charge beyond the normal usage fees for GCP resources.
v. Some Cloud Launcher images charge users fees, particularly those published by third parties with commercially licensed software. But they all show you estimates of their monthly charges before you launch them. Be aware that these estimates are just that, estimates. In

particular, they don't attempt to estimate networking costs since those will vary based on how you use the applications.

vi. A second note of caution. GCP updates the base images for these software packages to fix critical issues and vulnerabilities. But it doesn't update the software after it's been deployed. Fortunately, you'll have access to the deployed systems, so you can maintain them.

Note : In Google Cloud IAM: if a policy applied at the project level gives you Owner permissions, your access to an individual resource in that project might be restricted to View permission if someone applies a more restrictive policy directly to that resource : False.

Reason : Your permissions precede permissions of those higher in chain and you gave no one lower than you in chain the role to work with these permissions so naturally, you can't be restricted

1. Google Compute Engine

- Just like computers you might buy from the store, you can run whatever you want on them, do virtually anything with them. But you rent these ones by the second.
- One of the nice things about virtual machines is that they have the power and generality of a full-fledged operating system in each. You configure a virtual machine much like you build out a physical server by specifying its amounts of CPU power and memory, its amounts and types of storage and its operating system.
- You can flexibly reconfigure them and a VM running on Google's cloud has unmatched worldwide network connectivity.
- Compute Engine lets you create and run virtual machines on Google infrastructure.
- There are no upfront investments and you can run thousands of virtual CPUs on a system that is designed to be fast and to offer consistent performance.
- You can create a virtual machine instance by using the Google Cloud Platform console or the GCloud command line tool.
- Your VM can run Linux and Windows Server images provided by Google or customized versions of these images, and you can even import images for many of your physical servers. When you create a VM, pick a machine type which determines how much memory and how many virtual CPUs it has. These types range from very small to very large indeed.
- If you can't find a predefined type that meets your needs perfectly, you can make a custom VM.

- Speaking of processing power, if you have workloads like machine learning and data processing that can take advantage of GPUs, many GCP zones have GPU's available for you.

- Just like physical computers need disks, so do VM.

- You can choose **two kinds of persistent storage; standard or SSD**. If your application needs high-performance scratch space, you can attach a local SSD, but be sure to store data of permanent value somewhere else because local SSDs content doesn't last past when the VM terminates. That's why the other kinds are called persistent disks. Anyway, most people start off with standard persistent disks and that's the default.

- You'll also choose a boot image. GCP offers lots of versions of Linux and Windows ready to go and you can import your own images too.

- Lots of GCP customers want their VMs to always come up with certain configurations like installing software packages on first boot. It's very common to pass GCP VM startup scripts that do just that. You can also pass in other kinds of metadata too.

- Once your VMs are running, it's easy to take a durable snapshot of their disks. You can keep these as backups or use them when you need to migrate a VM to another region.

- Suppose you have a workload that no human being is sitting around waiting to finish, say a batch job analyzing large dataset, you can save money by choosing **preemptible VMs** to run the job. A preemptible VM is different from an ordinary Compute Engine VM in only one respect. You've given compute engine permission to terminate it if it's resources are needed elsewhere. You can save a lot of money with preemptible VMs, although be sure to make your job able to be stopped and restarted.

- You can choose the machine properties of your instances such as the number of virtual CPUs and the amount of memory by using a set of predefined machine types or by creating your own custom machine types.

- Compute Engine has a feature called auto scaling that lets you add and take away VMs from your application based on load metrics.

- The other part of making that work is balancing the incoming traffic across the VMs, and Google VPC supports several different kinds of load balancing.

2. Google Virtual Private Cloud

- Define their own **Virtual Private Cloud** inside their first GCP project, or they can simply choose the default VPC and get started with that.

- Your VPC networks connect your Google Cloud platform resources to each other and even isolate them from each other and wrt to the internet.

- You can segment your networks, use **firewall** rules to restrict access to instances, and create static routes to forward traffic to specific destinations.

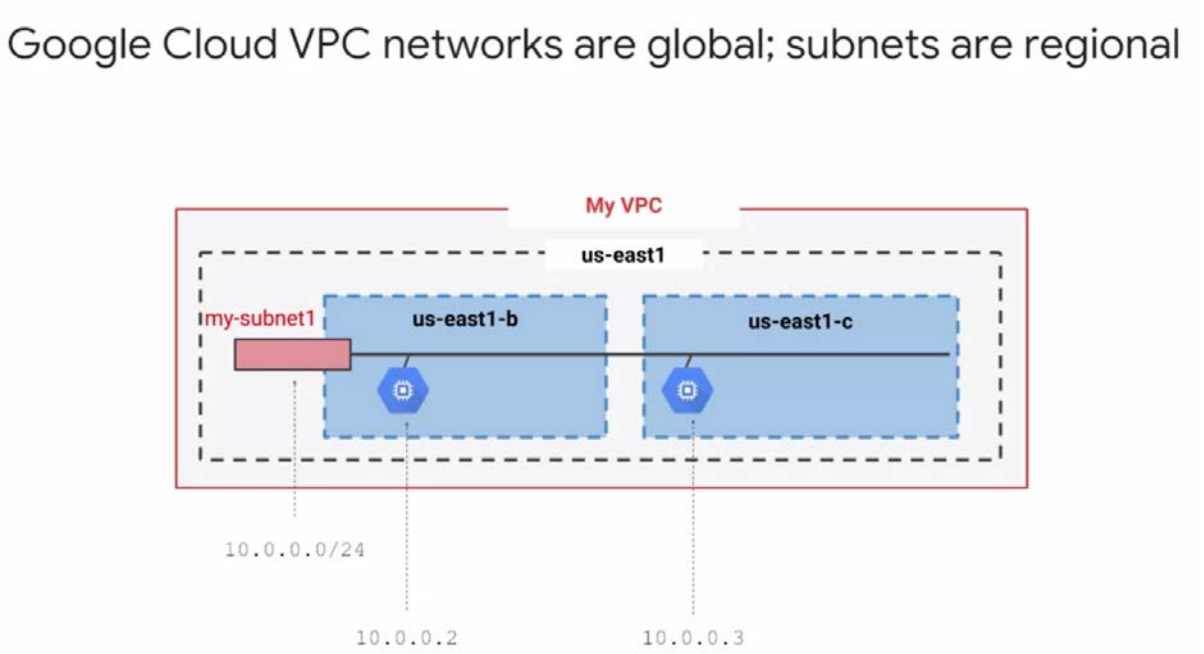
- The Virtual Private Cloud networks that you define have global scope. They can have subnets in any GCP region worldwide and subnets can span the zones that make up a region.

- This architecture makes it easy for you to define your own network layout with global scope.

- You can also have resources in different zones on the same subnet.

- You can dynamically increase the size of a subnet in a custom network by expanding the range of IP addresses allocated to it. Doing that doesn't affect already configured VMs.

- *Example :* your VPC has one network. So far, it has one subnet defined in GCP us-east1 region. Notice that it has two Compute Engine VMs attached to it. They're neighbors on the same subnet even though they are in different zones. You can use this capability to build solutions that are resilient but still have simple network layouts.



- Much like physical networks, VPCs have routing tables. These are used to forward traffic from one instance to another instance within the same network. Even across sub-networks and even between GCP zones without requiring an external IP address. VPCs routing tables are built in, you don't have to provision or manage a router.

- Another thing you don't have to provision or manage for GCP, a firewall instance. VPCs give you a **global distributed firewall**. You can control to restrict access to instances, both incoming and outgoing traffic. You can define firewall rules in terms of metadata tags on Compute Engine instances, which is really convenient.

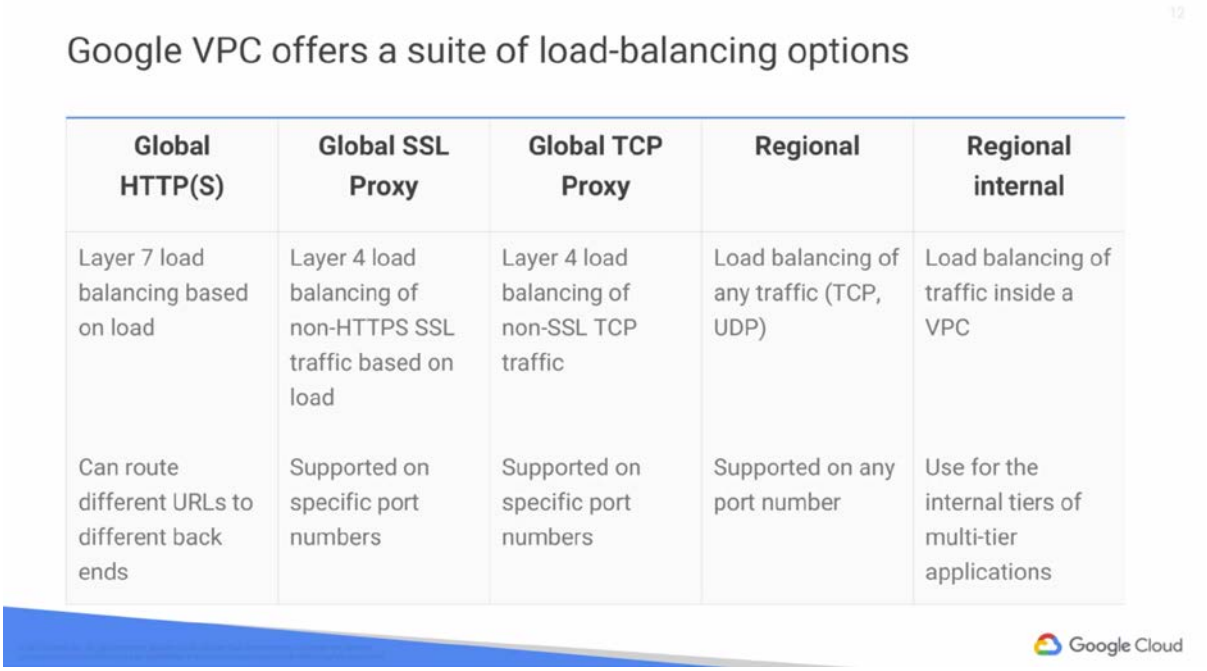
- Firewall and routers are managed by Google

- *For example*, you can tag all your web servers with say, "web," and write a firewall rule saying that traffic on ports 80 or 443 is allowed into all VMs with the "web" tag, no matter what their IP address happens to be.

- If you simply want to establish a peering relationship between two VPCs so that they can exchange traffic, that's what **VPC Peering** does. On the other hand, if you want to use the full power of IAM to control who and what in one project can interact with a VPC in another, that's what **Shared VPC** is for.

- **Cloud Load Balancing** is a fully distributed, software-defined managed service for all your traffic. And because the load balancers don't run in VMs you have to manage, you don't have to worry about scaling or managing them.

- You can put Cloud Load Balancing in front of all your traffic - HTTP and HTTPS, other TCP and SSL traffic, and UDP traffic too.



- With Cloud Load Balancing, a single anycast IP frontends all your backend instances in regions around the world. It provides cross-region load balancing, including automatic multi-region failover, which gently moves traffic in fractions if backends become unhealthy.

- Cloud Load Balancing reacts quickly to changes in users, traffic, backend health, network conditions, and other related conditions.

- If you need cross regional load balancing for a web application, use **HTTPS load balancing.**

- For Secure Sockets Layer traffic that is **not HTTP, use the global SSL proxy** load balancer.

- If it's **other TCP traffic that does not use Secure Sockets Layer, use the global TCP proxy load balancer**. Those two proxy services only work for specific port numbers, and they only work for TCP.

- If you want to load balance **UDP traffic** or traffic on any port number, you can still load balance cross a GCP region with the **regional load balancer**. Finally, what all those services have in common is that they're intended for traffic coming into the Google network from the internet.

- But what if you want to load balance traffic inside your project? Say, between the presentation layer and the business logic layer of your application? For that, use the **internal load balancer.** It accepts traffic on a GCP internal IP address and load balances it across Compute Engine VMs.

- One of the most famous Google services that people don't pay for is 8.8.8.8, which provides a public domain name service to the world. DNS is what translates internet host names to addresses. Google has a highly developed DNS infrastructure. It makes 8.8.8.8 available so that everybody can take advantage of it.

- But what about the internet host names and addresses of applications you build in GCP? GCP offers **Cloud DNS** to help the world find them. It's a managed DNS service running on the same infrastructure as Google. It has low latency and high availability and it's a cost-effective way to make your applications and services available to your users.

- The DNS information you publish is served from redundant locations around the world. Cloud DNS is also programmable. You can publish and manage millions of DNS zones and records using the GCP console, the command line interface or the API.

- Google has a global system of edge caches. You can use this system to accelerate content delivery in your application using **Google Cloud CDN**. Your customers will experience lower network latency. The origins of your content will experience reduced load and you can save money too.

- Once you've set up HTTPS load balancing, simply enable Cloud CDN with a single checkbox.

- There are lots of other CDNs out there of course. What if you're already using one? Chances are, your CDN is a part of GCPs, CDN interconnect partner program and you can continue to use it.

- Lots of GCP customers want to interconnect their other networks to their Google VPCs, such as on-premises networks or their networks in other clouds.

- There are many good choices. Many customers start with a **Virtual Private Network** connection over the internet using the IPSEC protocol.

- To make that dynamic, they use a GCP feature called **Cloud Router**. Cloud Router lets your other networks and your Google VPC exchange route information over the VPN using the **Border Gateway Protocol.**

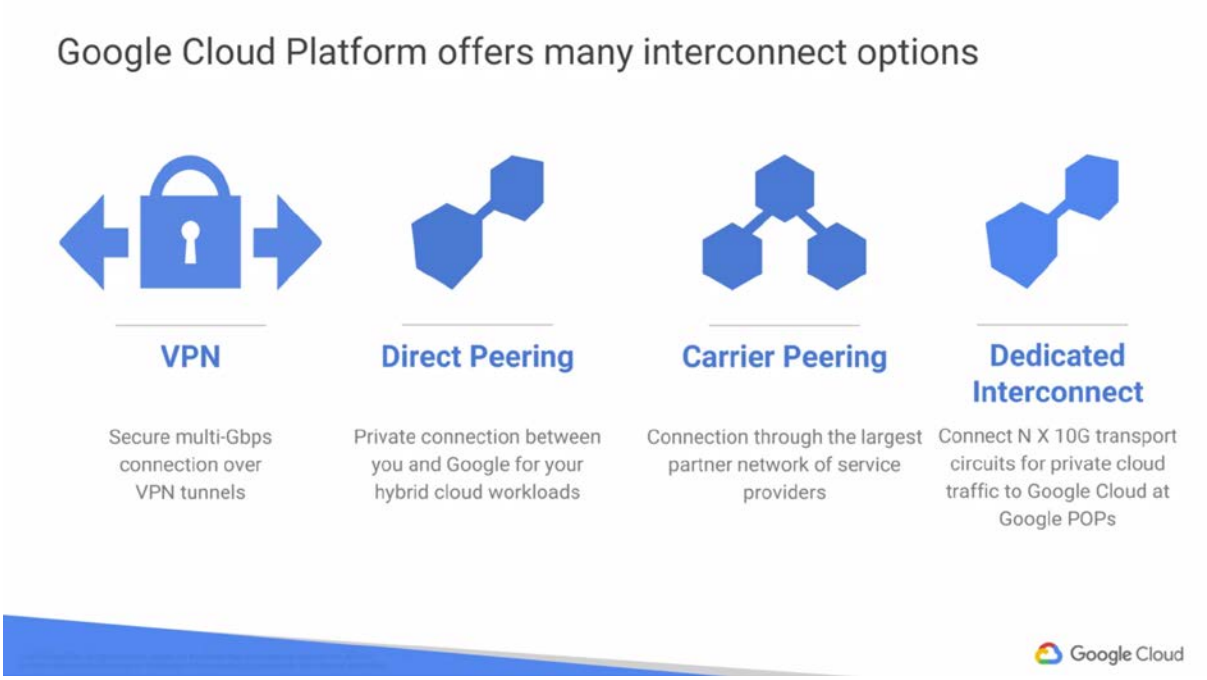
- For instance, if you add a new subnet to your **Google VPC**, your on-premises network will automatically get routes to it. But some customers don't want to use the internet, either because of security concerns or because they need more reliable bandwidth.

- They can conCIDR peering with Google using **Direct Peering**. Peering means putting a router in the same public data center as a Google point of presence and exchanging traffic. Google has more than 100 points of presence around the world.

- Customers who aren't already in a point of presence can contract with a partner in the carrier peering program to get connected.

- One downside of peering though is that it isn't covered by a Google service level agreement. Customers who want the highest uptimes for their interconnection with Google should use **Dedicated Interconnect**, in which customers get one or more direct private connections to Google. If these connections have topologies that meet Google's specifications,

- they can be covered by up to a 99.99 percent SLA. These connections can be backed up by a VPN for even greater reliability.



NOTE : A GCP customer wants to load-balance traffic among the back-end VMs that form part of a multi-tier application. Which load-balancing option should this customer choose? : An internal load balancer

3. GCP Storage

- Different applications and workloads required different storage database solutions.
- Can store data on your VM's persistent disk.
- Google Cloud Platform has other storage options to meet your needs for structured, unstructured, transactional and relational data.
- Core storage options:
    - Cloud Storage,
    - Cloud SQL,
    - Cloud Spanner,
    - Cloud Data Store
    - Google Big Table.
- Depending on your application, you might want to use one or several of these services to get the job done.

4. Google Cloud Storage

- **Object storage** : It's not the same as file storage, (in which you manage your data as a hierarchy of folders.) It's not the same as block storage(in which your operating system manages your data as chunks of disk.) Instead, object storage means you save to your storage here, **you keep this arbitrary bunch of bytes I give you and the storage lets you**

address it with a unique key. Often these unique keys are in the form of URLs which means object storage interacts nicely with Web technologies.

- It's a fully managed scalable service. That means that you don't need to provision capacity ahead of time. Just make objects and the service stores them with high durability and high availability. You can use Cloud Storage for lots of things: serving website content, storing data for archival and disaster recovery, or distributing large data objects to your end users via **Direct Download**.

- Cloud Storage is not a file system because each of your objects in Cloud Storage has a URL. Each feels like a file in a lot of ways and that's okay to use the word "file" informally to describe your objects, but still it's not a file system. You would not use Cloud Storage as the root file system of your Linux box.

- Instead, Cloud Storage is comprised of buckets you create and configure and use to hold your storage objects. The storage objects are **immutable,** which means that you do not edit them in place but instead you create new versions.

- Cloud Storage always encrypts your data on the server side before it is written to disk and you don't pay extra for that. Also by default, data in-transit is encrypted using HTTPS.

- **Transferring Data** : there are services you can use to get large amounts of data into Cloud Storage conveniently. Once they are in Cloud Storage, you can move them onwards to other GCP storage services.

- Cloud Storage files are organized into buckets. When you create a bucket, you give it a globally unique name. You specify a geographic location where the bucket and its contents are stored and you choose a default storage class.

- Pick a location that minimizes latency for your users. In other words, if most of your users are in Europe, you probably want to pick a European location.

- Speaking of your users, there are several ways to control access to your objects and buckets. For most purposes, Cloud IAM is sufficient. Roles are inherited from project to bucket to object.

- If you need finer control, you can create access control lists - ACLs - that offer finer control. ACLs define who has access to your buckets and objects as well as what level of access they have.

- Each ACL consists of two pieces of information, a scope which defines who can perform the specified actions, for example, a specific user or group of users and a permission which defines what actions can be performed for example, read or write.

- Cloud Storage objects are immutable. You can turn on object versioning on your buckets if you want. If you do, Cloud Storage keeps a history of modifications. That is, it overrides or deletes all of the objects in the bucket.

- You can list the archived versions of an object, restore an object to an older state or permanently delete a version as needed. If you don't turn on object versioning, new always overrides old.

- Cloud Storage also offers lifecycle management policies. For example, you could tell Cloud Storage to delete objects older than 365 days. Or you could tell it to delete objects created before January 1, 2013 or keep only the three most recent versions of each object in a bucket that has versioning enabled.

Your Cloud Storage files are organized into buckets

Bucket attributes	Bucket contents
Globally unique name	Files (in a flat namespace)
Storage class	
Location (region or multi-region)	
IAM policies or Access Control Lists	Access Control Lists
Object versioning setting	
Object lifecycle management rules	

Choosing among Cloud Storage classes

	Multi-regional	Regional	Nearline	Coldline
Intended for data that is...	Most frequently accessed	Accessed frequently within a region	Accessed less than once a month	Accessed less than once a year
Availability SLA	99.95%	99.90%	99.00%	99.00%
Access APIs	***Consistent APIs***			
Access time	***Millisecond access***			
Storage price	Price per GB stored per month			
Retrieval price				Total price per GB transferred
Use cases	Content storage and delivery	In-region analytics, transcoding	Long-tail content, backups	Archiving, disaster recovery

- Cloud Storage lets you choose among four different types of storage classes: Regional,
- Multi-regional, Nearline, and Coldline.
- **Regional storage** lets you store your data in a specific GCP region: US Central one, Europe West one or Asia East one. It's cheaper than Multi-regional storage but it offers less redundancy.

    Use : People use regional to store data close to their Compute Engine, virtual machines, or their Kubernetes engine clusters. That gives better performance for data-intensive computations.

- **Multi-regional storage** on the other hand, cost a bit more but it's Geo-redundant. That means you pick a broad geographical location like the United States, the European Union, or Asia and cloud storage stores your data in at least two geographic locations separated by at least 160 kilometers.

  Use : Multi-regional storage is appropriate for storing frequently accessed data. For example, website content, interactive workloads, or data that's part of mobile and gaming applications.

- **Nearline storage** is a low-cost, highly durable service for storing infrequently accessed data. The storage class is a better choice than Multi-regional storage or Regional storage in scenarios where you plan to read or modify your data once a month or less on average.

  For example, if you want to continuously add files to cloud storage and plan to access those files once a month for analysis, Nearline storage is a great choice.

- **Coldline storage** is a very low cost, highly durable service for data archiving, online backup, and disaster recovery. Coldline storage is the best choice for data that you plan to access -at most - once a year. This is due to its slightly lower availability, 90-day minimum storage duration, costs for data access, and higher per operation costs.

  For example, if you want to archive data or have access to it in case of a disaster recovery event.

- As for **pricing**, all storage classes incur a cost per gigabyte of data stored per month,

- with Multi-regional having the highest storage price and Coldline the lowest storage price. Egress and data transfer charges may also apply. Nearline and Coldline storage assess additional retrieval fees

- In addition to those charges, Nearline storage also incurs an access fee per gigabyte of data read and Coldline storage incurs a higher fee per gigabyte of data read.

- Regardless of which storage class you choose, there are several ways to bring data into cloud storage.

    - Many customers simply use gsutil which is the cloud storage command from this cloud SDK.

    - You can also move data in with a drag and drop in the GCP console, if you use the Google Chrome browser.

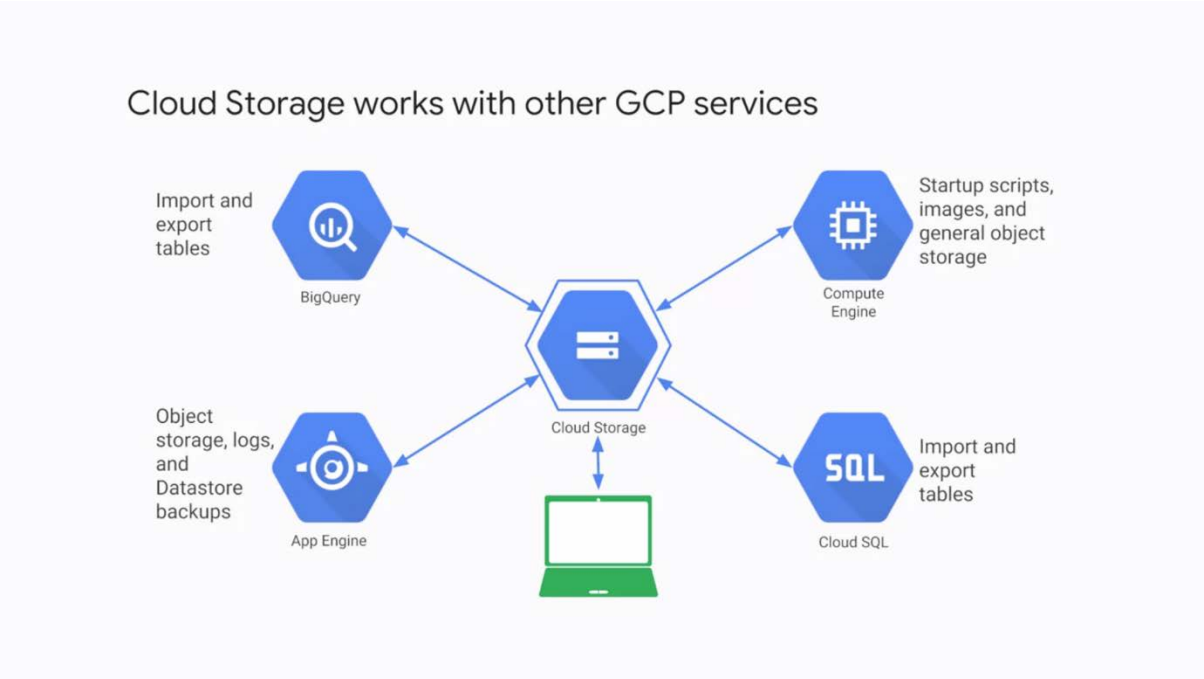
    - Google Cloud platform offers the online storage transfer service and the offline transfer appliance to help. The storage transfer service lets you schedule and manage batch transfers to cloud storage from another cloud provider from a different cloud storage region or from an HTTPS endpoint.

    - The transfer appliance is a rackable, high-capacity storage server that you lease from Google Cloud. You simply connect it to your network, load it with data, and then ship it to an upload facility where the data is uploaded to cloud storage. This service enables you to securely transfer up to a petabyte of data on a single appliance.

- There are other ways of getting your data into cloud storage as this storage option is tightly integrated with many of the Google cloud platform products and services.

- *For example* : you can import and export tables from and to BigQuery as well as Cloud SQL.

- You can also store App Engine logs, cloud data store backups, and objects used by App Engine applications like images. Cloud storage can also store instant startup scripts, Compute Engine images, and objects used by Compute Engine applications. In short, cloud storage is often the ingestion point for data being moved into the cloud and is frequently the long-term storage location for data.



5. Google Cloud BigTable

- Cloud Bigtable is Google's NoSQL, big data database service. (not all the rows might need to have the same columns.). And in fact, the database might be designed to take advantage of that by sparsely populating the rows. That's part of what makes a NoSQL database what it is.

- Databases in Bigtable are sparsely populated tables that can scale to billions of rows and thousands of columns allowing you to store petabytes of data. **GCP fully manages** the surface, so you don't have to configure and tune it.

- It's ideal for data that has a single lookup key. Some applications developers think of Bigtable as a persistent hash table.

- Cloud Bigtable is ideal for storing large amounts of data with very low latency. It supports high throughput, both read and write, so it's a great choice for both operational and analytical applications including Internet of Things, user analytics and financial data analysis.

- Cloud Bigtable is offered through the same open source API as HBase, which is the native database for the Apache Hadoop project. Having the same API enables portability of applications between HBase and Bigtable.

- **Apache Hadoop** is a collection of open-source software utilities that facilitates using a network of many computers to solve problems involving massive amounts of data and computation. It provides a software framework for distributed storage and processing of big data using the MapReduce programming model.

- **HBase** is an open-source non-relational distributed database modeled after Google's Bigtable and written in Java. It is developed as part of Apache Software Foundation's Apache Hadoop project and runs on top of HDFS or Alluxio, providing Bigtable-like capabilities for Hadoop.

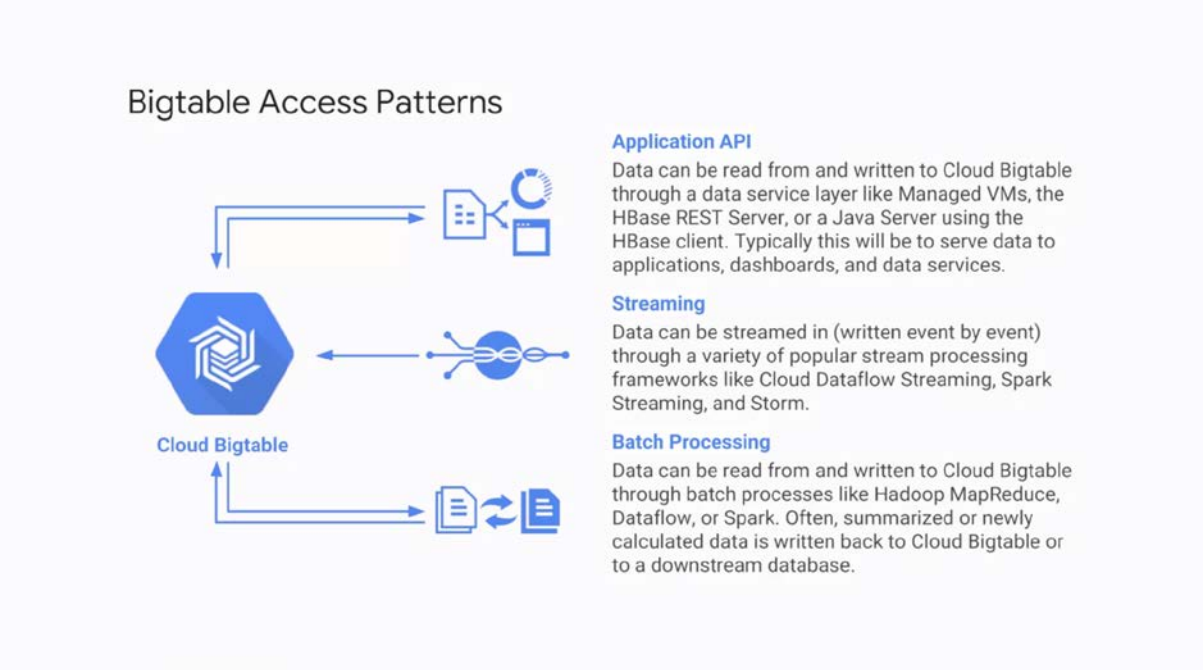
- *Scalability* : If you manage your own Hbase installation, scaling past a certain rate of queries per second is going to be tough, but with Bigtable you can just increase your machine count which doesn't even require downtime.

- Also, Cloud Bigtable handles administration tasks like upgrades and restarts transparently.

- All data in Cloud Bigtable is encrypted in both in-flight and at rest. You can even use IAM permissions to control who has access to Bigtable data.

- One last reference point. Bigtable is actually the same database that powers many of Google's core services including search, analytics, maps and Gmail.

- As Cloud Bigtable is part of the GCP ecosystem, it can interact with other GCP services and third-party clients.



6. GOOGLE CLOUD DATASTORE

- Another highly scalable NoSQL database choice for your applications

- One of its main use cases is to store structured data from App Engine apps. You can also build solutions that span App Engine and Compute Engine with Cloud Datastore as the integration point.

- Fully-managed service : Cloud Datastore automatically handles sharding and replication, providing you with a highly available and durable database that scales automatically to handle load.

- Unlike Cloud Bigtable, it also offers transactions that affect multiple database rows, and it lets you do SQL-like queries.

- To get you started, unlike BigTable, Cloud Datastore has a free daily quota that provides storage, reads, writes, deletes and small operations at no charge.

7. CLOUD SQL

- Relational database services - these services use a database schema to help your application keep your data consistent and correct.
- Another feature of relational database services that helps with the same goal - transactions.
- Your application can designate a group of database changes as all or nothing.
- Either they all get made, or none do. Without database transactions, your online bank wouldn't be able to offer you the ability to move money from one account to another.
- Classically, relational databases are a lot of work to set up, maintain, manage, and administer.
- If that doesn't sound like a good use of your time but you still want the protections of a relational database, conCIDR Cloud SQL. It offers you your choice of the MySQL or PostgreSQL database engines as a fully managed service.
- Cloud SQL offers both MySQL and PostgreSQL databases that are capable of handling terabytes of storage.
- Of course, you could always run your own database server inside a Compute Engine virtual machine, which a lot of GCP customers do. But there are some benefits of using the Cloud SQL managed service instead.
- First, Cloud SQL provides several replica services like read, failover, and external replicas.
- This means that if an outage occurs, Cloud SQL can replicate data between multiple zones with automatic failover. Cloud SQL also helps you backup your data with either on-demand or scheduled backups.
- It can also scale both vertically by changing the machine type, and horizontally via read replicas.
- From a security perspective, Cloud SQL instances include network firewalls, and customer data is encrypted when on Google's internal networks, and when stored in database tables, temporary files, and backups.
- The dbs are accessible by other GCP services and even external services. You can authorize Compute Engine instances for access Cloud SQL instances and configure the Cloud SQL instance to be in the same zone as your virtual machine.
- Cloud SQL also supports other applications and tools that you might be used to, like SQL WorkBench, Toad, and other external applications using standard MySQL drivers.

8. CLOUD SPANNER

- If Cloud SQL does not fit your requirements because you need horizontal scaleability, conCIDR using Cloud Spanner.
- It offers transactional consistency at a global scale, schemas, SQL, and automatic synchronous replication for high availability. And, it can provide petabytes of capacity.
- ConCIDR using Cloud Spanner if you have outgrown any relational database, or sharding your databases for throughput high performance, need transactional consistency, global data and strong consistency, or just want to consolidate your database.
- Natural use cases include, financial applications, and inventory applications.

Comparing storage options: technical details

	Cloud Datastore	Bigtable	Cloud Storage	Cloud SQL	Cloud Spanner	BigQuery
Type	NoSQL document	NoSQL wide column	Blobstore	Relational SQL for OLTP	Relational SQL for OLTP	Relational SQL for OLAP
Transactions	Yes	Single-row	No	Yes	Yes	No
Complex queries	No	No	No	Yes	Yes	Yes
Capacity	Terabytes+	Petabytes+	Petabytes+	Terabytes	Petabytes	Petabytes+
Unit size	1 MB/entity	~10 MB/cell ~100 MB/row	5 TB/object	Determined by DB engine	10,240 MiB/ row	10 MB/row

Comparing storage options: technical details

	Cloud Datastore	Cloud Bigtable	Cloud Storage	Cloud SQL	Cloud Spanner	BigQuery
Type	NoSQL document	NoSQL wide column	Blobstore	Relational SQL for OLTP	Relational SQL for OLTP	Relational SQL for OLAP
Best for	Semi-structured application data, durable key-value data	"Flat" data, Heavy read/write, events, analytical data	Structured and unstructured binary or object data	Web frameworks, existing applications	Large-scale database applications (> ~2 TB)	Interactive querying, offline analytics
Use cases	Getting started, App Engine applications	AdTech, Financial and IoT data	Images, large media files, backups	User credentials, customer orders	Whenever high I/O, global consistency is needed	Data warehousing

9. KUBERNETES

- Kubernetes is an open-source orchestrator for containers so you can better manage and scale your applications. Kubernetes offers an API that lets people, that is authorized people, not just anybody, control its operation through several utilities.

- **Cluster** is a set of master components that control the system as a whole and a set of nodes that run containers.

- In Kubernetes, a node represents a computing instance. In Google Cloud, nodes are virtual machines running in Compute Engine.

- To use Kubernetes, you can describe a set of applications and how they should interact with each other, and Kubernetes figures out how to make that happen.

- You can always build one yourself on your own hardware, or in any environment that provides virtual machines, but that's work. And if you've built it yourself, you have to maintain it. That's even more toil.

- Because that effort is not always a valuable use of your time, Google Cloud provides Kubernetes Engine, which is Kubernetes as a managed service in the cloud. You can create a Kubernetes cluster with **Kubernetes Engine** using the GCP console or the g-cloud command that's provided by the Cloud SDK.

- GKE clusters can be customized, and they support different machine types, numbers of nodes and network settings. Here's a sample command for building a Kubernetes cluster using GKE.

gcloud container clusters create k1.

- When this command completes, you will have a cluster called K1, complete, configured and ready to go.

- Whenever Kubernetes deploys a container or a set of related containers, it does so inside an abstraction called a pod.

- A **pod** is the smallest deployable unit in Kubernetes. Think of a pod as if it were a running process on your cluster.

- It could be one component of your application or even an entire application.

- It's common to have only one container per pod. But if you have multiple containers with a hard dependency, you can package them into a single pod. They'll automatically share networking and they can have disk storage volumes in common.

- Each pod in Kubernetes gets a unique IP address and set of ports for your containers. Because containers inside a pod can communicate with each other using the localhost network interface, they don't know or care which nodes they're deployed on.

- One way to run a container in a pod in Kubernetes is to use the kubectl run command.

- Running the kubectl run command starts a deployment with a container running a pod. In this example, the container running inside the pod is an image of the popular nginx open source web server. The kubectl command is smart enough to fetch an image of nginx of the version we request from a container registry.

- A deployment represents a group of replicas of the same pod. It keeps your pods running even if a node on which some of them run on fails. You can use a deployment to contain a component of your application or even the entire application. In this case, it's the nginx web server.

- To see the running nginx pods, run the command **kubectl get pods**. By default, pods in a deployment or only accessible inside your cluster, but what if you want people on the Internet to be able to access the content in your nginx web server? To make the pods in your deployment publicly available, you can connect a **load balancer** to it by running the

  kubectl expose command.

- Kubernetes then creates a service with a fixed IP address for your pods. A service is the fundamental way Kubernetes represents load balancing. To be specific, you requested Kubernetes to attach an external load balancer with a public IP address to your service so that others outside the cluster can access it.

- In GKE, this kind of load balancer is created as a **network load balancer**. This is one of the managed load balancing services that Compute Engine makes available to virtual machines. GKE makes it easy to use it with containers.

- Any client that hits that IP address will be routed to a pod behind the service. In this case, there is only one pod, your simple nginx pod.

- A **service** groups a set of pods together and provides a stable endpoint for them. In our case, a public IP address managed by a network load balancer, although there are other choices.

- As deployments create and destroy pods, pods get their own IP addresses, but those addresses don't remain stable over time. Services provide that stable endpoint you need.

- As you learn more about Kubernetes, you'll discover other service types that are suitable for internal application back ends. The **kubectl get services** command shows you your service's public IP address. Clients can use this address to hit the nginx container remotely.

- To scale a deployment, **run the kubectl scale command.**

- Now our deployment has 3 nginx web servers, but they're all behind the service and they're all available through one fixed IP address.

- You could also use **auto scaling** with all kinds of useful parameters.

- *For example*, here's how to auto scale a deployment based on CPU usage. In the command shown, you specify a minimum number of pods, 10, a maximum number of pods, 15, and the criteria for scaling up. In this case, Kubernetes will scale up the number of pods when CPU usage hits 80% of capacity.

- Instead of issuing commands, you provide a configuration file that tells Kubernetes what you want your desired state to look like and Kubernetes figures out how to do it. These

configuration files then become your management tools. To make a change, edit the file and then present the changed version to Kubernetes.

- And you can save them in a version control system to keep track of the changes you made to your infrastructure. In this case, the deployment configuration file declares that you want 3 replicas of your nginx pod. It defines a selector field, so your deployment knows how to group specific pods as replicas. It works because all of those specific pods share a **label**. Their app is tagged as nginx.

- To illustrate the flexibility of this declarative method, in order to run 5 replicas instead of 3, all you need to do is edit the deployment config file, changing 3 to 5. And then run the **kubectl apply command** to use the updated config file.

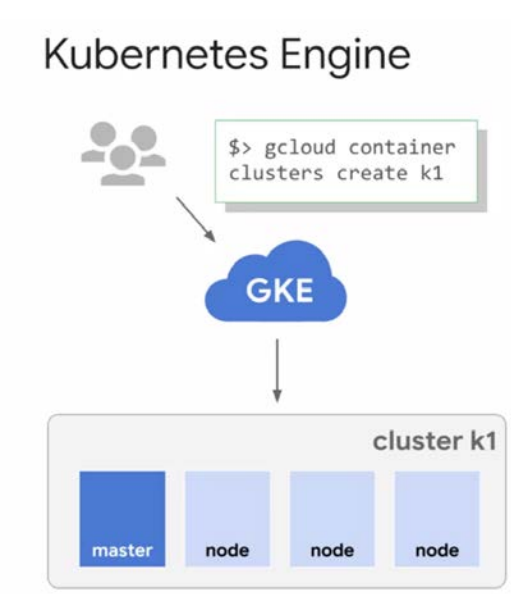
- Now use the **kubectl get replicasets** command to view your replicas and see their updated state. Then use the **kubectl get pods** command to watch the pods come online. In this case, all 5 are ready and running. Finally, let's check the deployments to make sure the proper number of replicas are running using **kubectl get deployments.**

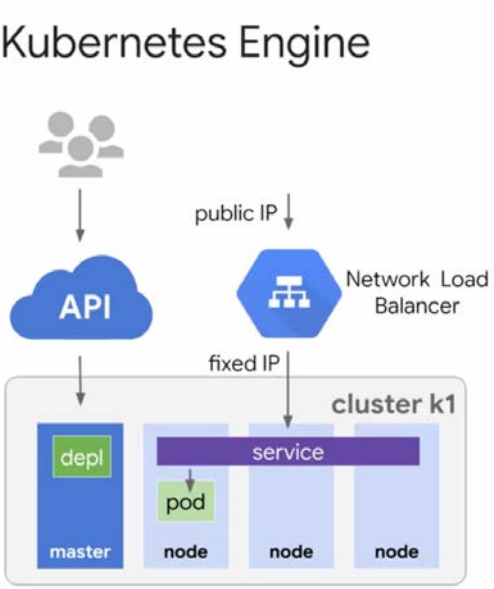
- In this case, all 5 pod replicas are available. And clients can still hit your endpoint, just like before. The **kubectl get services** command confirms that the external IP of the service is unaffected.

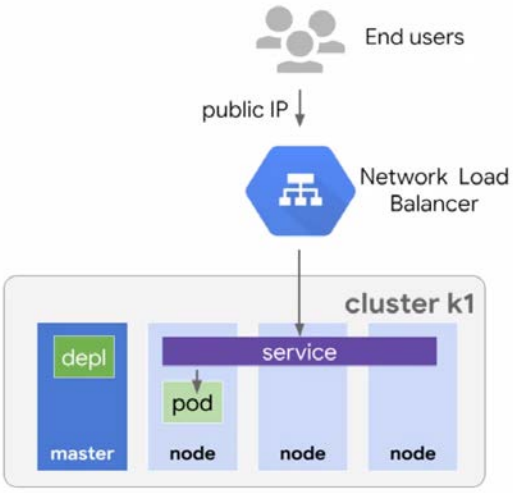
- Now you have 5 copies of your nginx pod running in GKE, and you have a single service that's proxying the traffic to all 5 pods. This technique allows you to share the load and scale your service in Kubernetes.

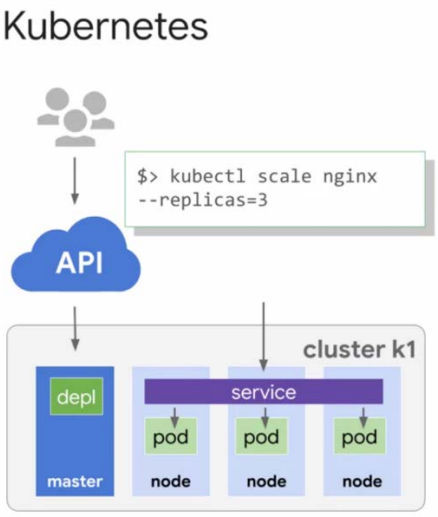
- You will definitely want to update your container and get the new code out in front of your users as soon as possible, but it could be risky to roll out all those changes at once.You do not want your users to experience downtime while your application rebuilds and redeploys. That's why one attribute of a deployment is its **update strategy**. Here's an example, a rolling update.

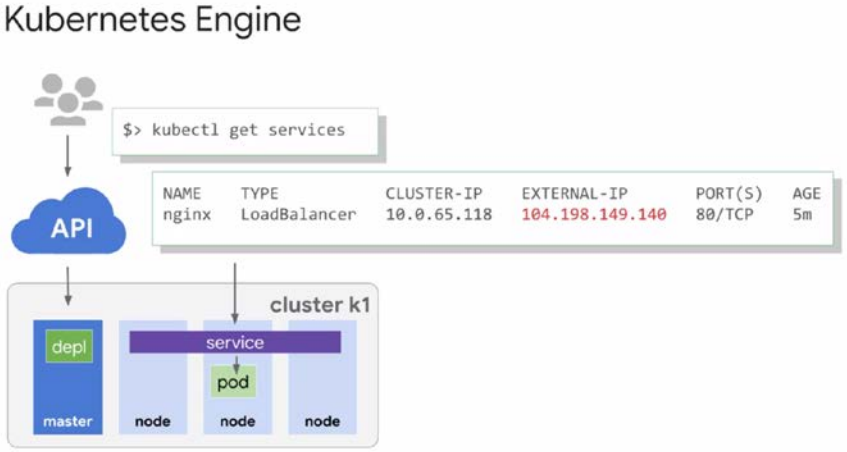
- When you choose a rolling update for a deployment and then give it a new version of the software that it manages, Kubernetes will create pods of the new version one-by-one, waiting for each new version pod to become available before destroying one of the old version pods. Rolling updates are a quick way to push out a new version of your application while still sparing your users from experiencing downtime.


Kubernetes Engine
$> gcloud container clusters create k1
GKE
cluster k1
master
node
node
node

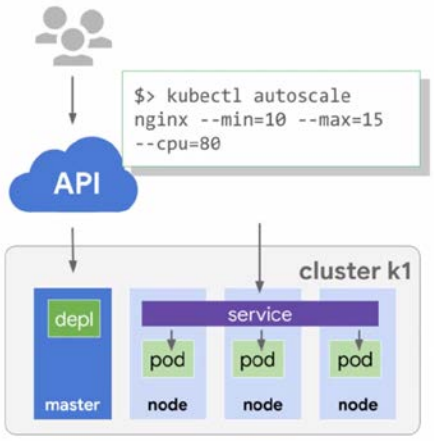

Kubernetes Engine
public IP
API
Network Load Balancer
fixed IP
cluster k1
depl
service
pod
master
node
node
node


End users
public IP
Network Load Balancer
cluster k1
depl
service
pod
master
node
node
node

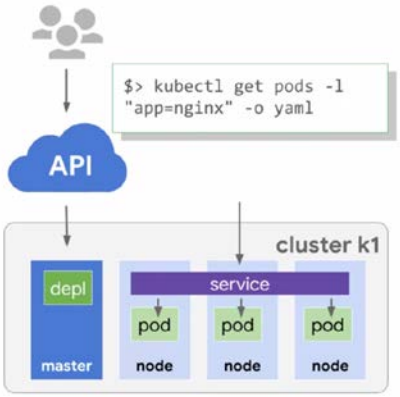

Kubernetes
$> kubectl scale nginx --replicas=3
API
cluster k1
depl
service
pod
pod
pod
master
node
node
node


Kubernetes Engine
$> kubectl get services
NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE
nginx LoadBalancer 10.0.65.118 104.198.149.140 80/TCP 5m
API
cluster k1
depl
service
pod
master
node
node
node

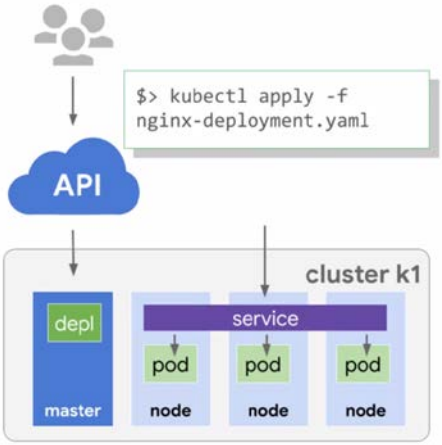
Kubernetes



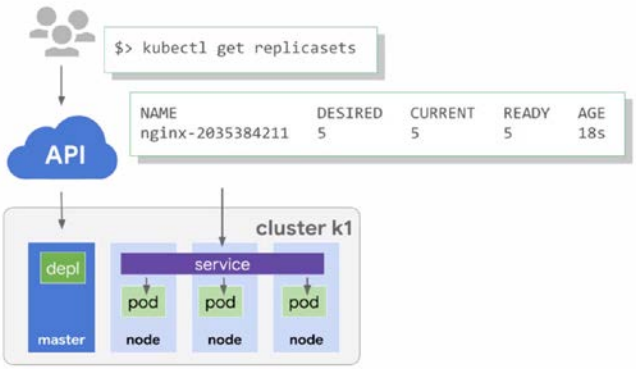
Kubernetes



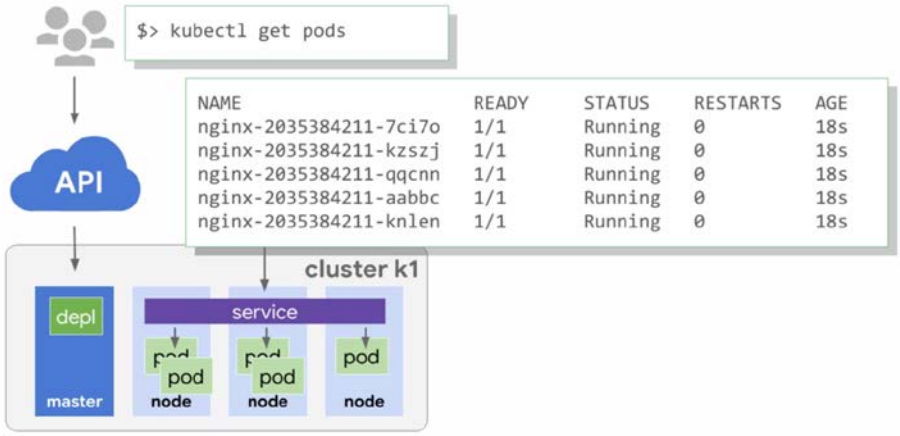
Kubernetes



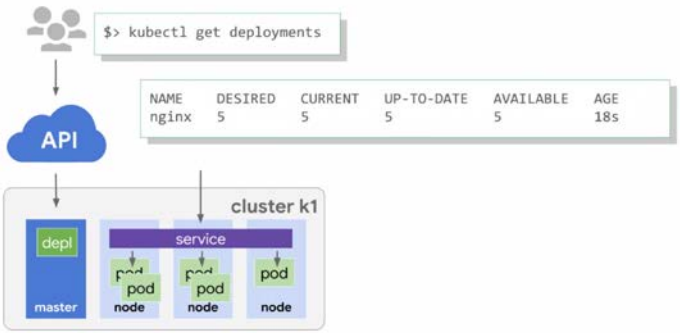
Kubernetes



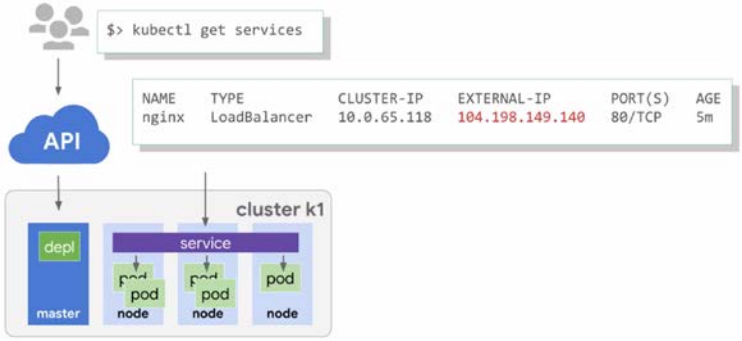
Kubernetes



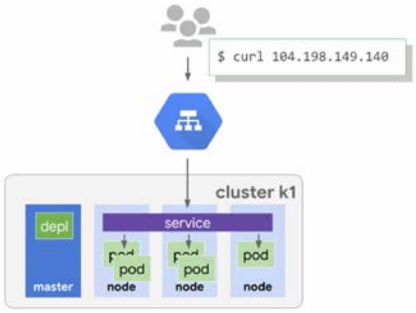
Kubernetes



Kubernetes

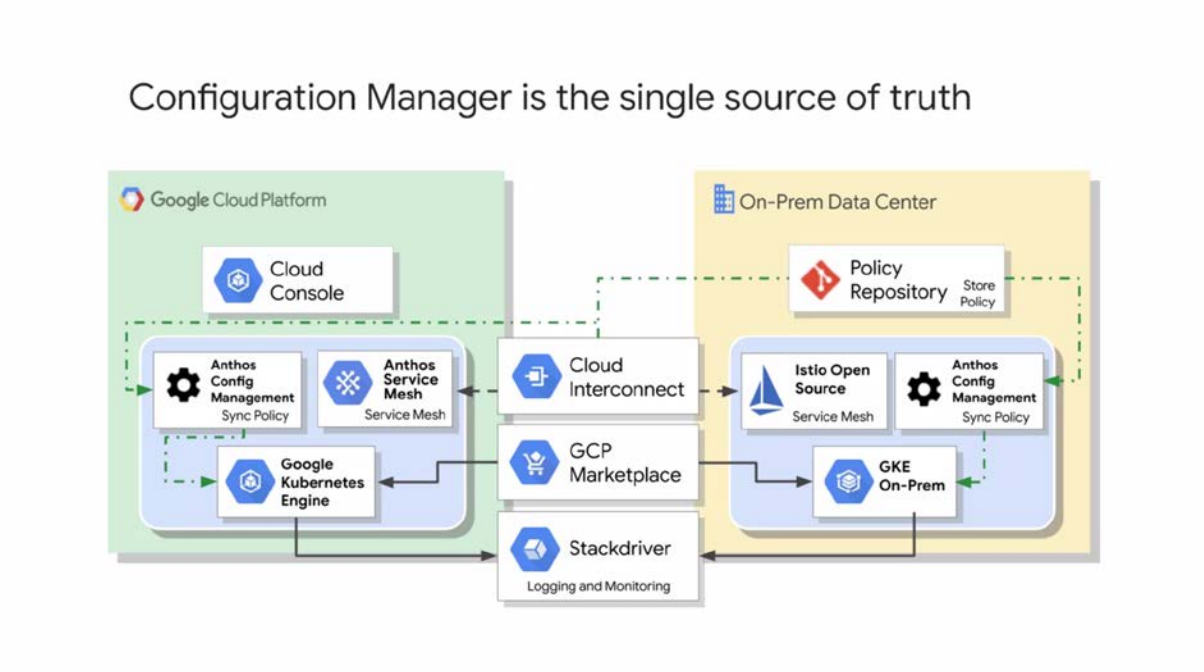


Kubernetes Engine




```
spec:
  # ...
  replicas: 5
  strategy:
    rollingUpdate:
      maxSurge: 1
      maxUnavailable: 0
    type: RollingUpdate
  # ...
```

10. HYBRID CLOUD USING GKE, ANTHOS, ISTIOS AND CLOUD INTERCONNECT



- **Modern hybrid or multi-cloud architecture** can help. To summarize, it allows you to keep parts of your systems infrastructure on-premises while moving other parts to the Cloud, creating an environment that is uniquely suited to your company's needs.

- Move only specific workloads to the Cloud at your own pace because a full scale migration is not required for it to work.

- Take advantage of the flexibility, scalability, and lower computing costs offered by cloud services for running the workloads you decide to migrate. Add specialized services such as machine learning, content caching, data analysis, long-term storage, and IoT to your computing resources tool kit.

- **Anthos** is a hybrid and multi-cloud solution powered by the latest innovations in distributed systems, and service management software from Google. The Anthos framework rests on **Kubernetes** and **Google Kubernetes engine** deployed on-prem which provides the foundation for an architecture that is fully integrated with centralized management through a central control plane that supports policy based application lifecycle delivery across hybrid and multi-cloud environments.

- Anthos also provides a rich set of tools for monitoring and maintaining the consistency of your applications across all of your network, whether on-premises, in the Cloud, or in multiple clouds.

- **Google Kubernetes Engine on the Cloud** site of your hybrid network : GKE is a managed production-ready environment for deploying containerized applications. Operates seamlessly with high availability and an SLA. Runs certified Kubernetes ensuring portability across clouds and on-premises. Includes auto-node repair, and auto-upgrade, and auto-scaling. Uses regional clusters for high availability with multiple masters. Node storage replication across multiple zones.

- Its counterpart on the on-premises side of a hybrid network is **Google Kubernetes Engine deployed on-prem**. GKE deployed on-prem is a turn-key production-grade conformed version of Kubernetes with the best practice configuration already pre-loaded. Provides an

easy upgrade path to the latest Kubernetes releases that have been validated and tested by Google.

- Provides access to container services on Google Cloud platform, such as **Cloud build** (Google's container registry, kinda like Google's version of DockerHub), container registry, audit logging, and more. It also integrates with Istio, Knative and Marketplace Solutions. Ensures a consistent Kubernetes version and experience across Cloud and on-premises environments.

- As mentioned, both Google Kubernetes Engine in the Cloud and Google Kubernetes Engine deployed on-premises integrate with **Marketplace**, so that all of the clusters in your network, whether on-premises or in the Cloud, have access to the same repository of containerized applications. This allows you to use the same configurations on both the sides of the network, reducing the time spent developing applications.

- It's like the right ones replicate anywhere and maintaining conformity between your clusters.

- Enterprise applications may use hundreds of microservices to handle computing workloads. Keeping track of all of these services and monitoring their health can quickly become a challenge.

- **Anthos**, an **Istio** Open Source service mesh take all of these guesswork out of managing and securing your microservices. These service mesh layers communicate across the hybrid network using **Cloud interconnect**, as shown to sync and pass their data.

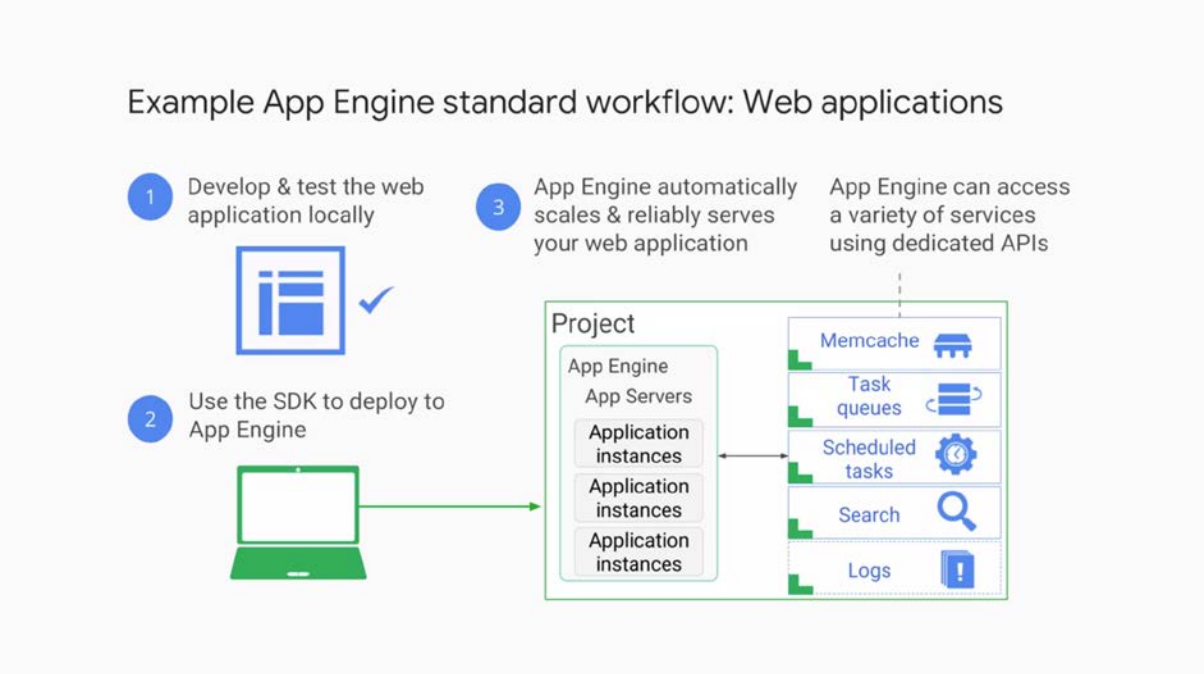
- **Stackdriver** is the built-in logging and monitoring solution for Google Cloud. Stackdriver offers a fully managed logging, metrics collection, monitoring dashboarding, and alerting solution that watches all sides of your hybrid on multi-cloud network. Stackdriver is the ideal solution for customers wanting a single easy to configure powerful cloud-based observability solution, that also gives you a single pane of glass dashboard to monitor all of your environments.

- Lastly, **Anthos Configuration Management** provides a single source of truth for your clusters configuration. That source of truth is kept in the policy repository, which is actually a git repository. In this illustration, this repository is happen to be located on-premises, but it can also be hosted in the Cloud.

- The Anthos Configuration Management agents use the policy repository to enforce configurations locally in each environment, managing the complexity of owning clusters across environments. Anthos Configuration Management also provides administrators and developers the ability to deploy code changes with a single repository commit. And the option to implement configuration inheritance, by using namespaces.

11. APP ENGINE

- PaaS is a platform as a service.
- The App Engine platform manages the hardware and networking infrastructure required to run your code. To deploy an application on App Engine, you just hand App Engine your code and the App Engine service takes care of the rest.
- App Engine provides you with a built-in services that many web applications need. NoSQL databases, in-memory caching, load balancing, health checks, logging and a way to authenticate users.
- You code your application to take advantage of these services and App Engine provides them. App engine will scale your application automatically in response to the amount of traffic it receives.
- So you only pay for those resources you use. There are no servers for you to provision or maintain.
- That's why App Engine is especially suited for applications where the workload is highly variable or unpredictable like web applications and mobile backend.
- App Engine offers two environments: standard and flexible.



- Of the two App Engine Environments, **Standard** is the simpler. It offers a simpler deployment experience than the **Flexible** environment and fine-grained auto-scale.
- It also offers a free daily usage quota for the use of some services. What's distinctive about the Standard Environment though, is that low utilization applications might be able to run at no charge.
- Google provides App Engine **software development kits** in several languages, so that you can test your application locally before you upload it to the real App Engine service. The SDKs also provide simple commands for deployment.
- App Engine's term for this kind of binary is the **runtime**. In App Engine Standard Environment, you use a runtime provided by Google.
- App Engine Standard Environment provides runtimes for specific versions of Java, Python, PHP and Go. The runtimes also include libraries that support App Engine APIs. And for many applications, the Standard Environment runtimes and libraries may be all you need.

Comparing the App Engine environments

	Standard Environment	Flexible Environment
Instance startup	Milliseconds	Minutes
SSH access	No	Yes (although not by default)
Write to local disk	No	Yes (but writes are ephemeral)
Support for 3rd-party binaries	No	Yes
Network access	Via App Engine services	Yes
Pricing model	After free daily use, pay per instance class, with automatic shutdown	Pay for resource allocation per hour; no automatic shutdown

Deploying Apps: Kubernetes Engine vs App Engine

	Kubernetes Engine	App Engine Flexible	App Engine Standard
Language support	Any	Any	Java, Python, Go, PHP
Service model	Hybrid	PaaS	PaaS
Primary use case	Container-based workloads	Web and mobile applications, container-based workloads	Web and mobile applications

Toward managed infrastructure *Toward dynamic infrastructure*

- If you want to code in another language, Standard Environment is not right for you. You'll want to conCIDR the Flexible Environment.
- The Standard Environment also enforces restrictions on your code by making it run in a so-called **"Sandbox".** That's a software construct that's independent of the hardware, operating system, or physical location of the server it runs on.
- The Sandbox is one of the reasons why App Engine Standard Environment can scale and manage your application in a very fine-grained way. Like all Sandboxes, it imposes some constraints.
- *For example*, your application can't write to the local file system. It'll have to write to a database service instead if it needs to make data persistent. Also, all the requests your application receives has a 60-second timeout, and you can't install arbitrary third party software. If these constraints don't work for you, that would be a reason to choose the Flexible Environment.
- You'll develop your application and run a test version of it locally using the App Engine SDK.
- Then when you're ready, you'll use the SDK to deploy it.

- Each App Engine application runs in a GCP project.
- App Engine automatically provisions server instances and scales and load balances them.
- Meanwhile, your application can make calls to a variety of services using dedicated APIs.
- *Here are a few examples*: a NoSQL data store to make data persistent, caching of that data using Memcache, searching logging, user logging, and the ability to launch actions not triggered by direct user requests, like task queues and a task scheduler.

12. HANDLING APIs - CLOUD ENDPOINTS & APIGEE

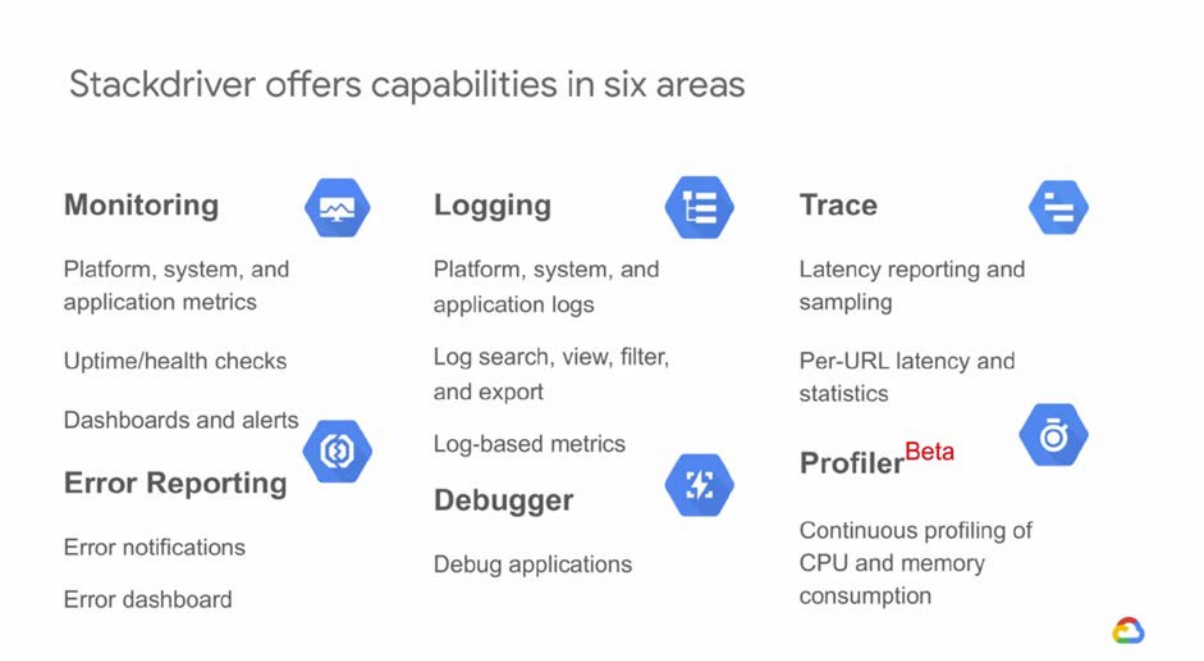
- API : a clean, well-defined interface, structured by app developers that abstracts away needless details and then they document that interface. The underlying implementation can change as long as the interface doesn't and other pieces of software that use the API don't have to know or care.
- Sometimes you have to change an API, say to add or deprecate a feature. To make this kind of API change cleanly, developers version their APIs. Version two of an API might contain calls that version one does not. Programs that consume the API can specify the API version that they want to use in their calls.
- Supporting an API is a very important task and Google Cloud platform provides two API management tools. They approach related problems in a different way and each has a particular strength.
- Suppose you're developing a software service and one of GCP's backends. You'd like to make it easy to expose this API. You'd like to make sure it's only consumed by other developers whom you trust. You'd like an easy way to monitor and log its use. You'd like for the API to have a single coherent way for it to know which end user is making the call. That's when you use **Cloud Endpoints**.
- It implements these capabilities and more using an easy to deploy proxy in front of your software service, and it provides an API console to wrap up those capabilities in an easy-to-manage interface. Cloud Endpoints supports applications running in GCP's compute platforms in your choice of languages and your choice of client technologies.
- **Apigee Edge** is also a platform for developing and managing API proxies. It has a different orientation though. It has a focus on business problems like rate limiting, quotas, and analytics.
- Many users of Apigee Edge are providing a software service to other companies and those features come in handy.
- Because of the backend services for Apigee Edge need not be in GCP, engineers often use it when they are "taking apart" a legacy application. Instead of replacing a monolithic application in one risky move, they can instead use Apigee Edge to peel off its services one by one, standing up microservices to implement each in turn, until the legacy application can be finally retired.

13. DEVELOPMENT IN THE CLOUD – CLOUD SOURCE REPOS & CLOUD FUNCS

- **Cloud Source Repositories** : a way to keep code private to a GCP project and use IAM permissions to protect it, but not have to maintain the Git instance yourself. It provides Git version control to support your team's development of any application or service, including those that run on App Engine, Compute Engine, and Kubernetes Engine.
- With Cloud Source Repositories, you can have any number of private Git repositories, which allows you to organize the code associated with your cloud project in whatever way works best for you.
- Cloud Source Repositories also contains a source viewer so that you can browse and view repository files from within the GCP console.

- Many applications contain event-driven parts. For example, maybe you have an application that lets users upload images. Whenever that happens, you need to process that image in various ways: convert it to a standard image format, thumbnail into various sizes, and store each in a repository. You could always integrate this function into your application, but then you have to worry about providing compute resources for it, no matter whether it happens once a day or once a millisecond.
- **Cloud Functions** : a single purpose function that doesthe necessary manipulations and then arrange for it to automatically run on being “triggered” (here, whenever a new image gets uploaded.)
- You don't have to worry about servers or runtime binaries. You just write your code in JavaScript for a Node.js environment that GCP provides and then configure when it should fire.
- There's no need for you to pay for servers either. You just pay whenever your functions run, in 100 millisecond intervals.
- Cloud Functions can trigger on events in Cloud Storage, Cloud Pub/Sub, or in HTTP call.
- You choose which events you care about. For each event type, you tell Cloud Functions you're interested in it. These declarations are called triggers. Then you attach JavaScript functions to your triggers. From now on, your functions will respond whenever the events happen.
- Some applications, especially those that have microservices architecture, can be implemented entirely in Cloud Functions.
- People also use Cloud Functions to enhance existing applications without having to worry about scaling. Your code executes whenever an event triggers it, no matter whether it happens rarely or several times per minute. You don’t have to provision compute resources to handle these operations

14. LOGGING AND MONITORING – STACKDRIVER



- Monitoring lets you figure out whether the changes you made were good or bad.

- **Stackdriver** is GCP's tool for monitoring, logging and diagnostics. Stackdriver gives you access to many different kinds of signals from your infrastructure platforms, virtual machines, containers, middleware and application tier, logs, metrics and traces. It gives

you insight into your application's health, performance and availability. So if issues occur, you can fix them faster.

- Here are the core components of Stackdriver: Monitoring, Logging, Trace, Error Reporting and Debugging.

- **Stackdriver Monitoring** checks the endpoints of web applications and other Internet accessible services running on your cloud environment.

  You can configure uptime checks associated with URLs, groups or resources such as Instances and load balancers.

  You can set up alerts on interesting criteria, like when health check results or uptimes fall into levels that need action. You can use Monitoring with a lot of popular notification tools. And you can create dashboards to help you visualize the state of your application.

- **Stackdriver Logging** lets you view logs from your applications and filter and search on them.

  Logging also lets you define metrics, based on log contents that are incorporated into dashboards and alerts. You can also export logs to BigQuery, Cloud Storage and Cloud PubSub.

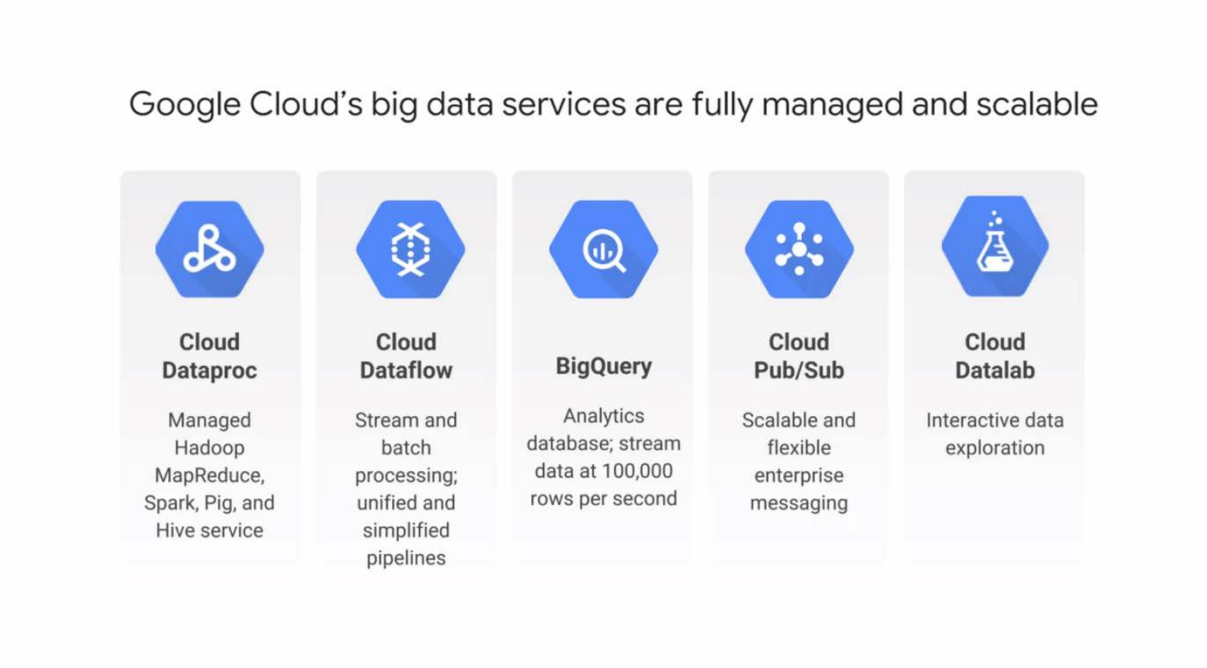
- **Stackdriver Error Reporting** tracks and groups the errors in your cloud apps. And it notifies you when new errors are detected.

- With **Stackdriver Trace**, you can sample the latency of app engine applications and report Per-URL statistics.

- **Stackdriver Debugger** connects your applications production data to your source code. So you can inspect the state of your application at any code location in production. That means you can view the application stage without adding logging statements.

  Stackdriver Debugger works best when your application source code is available, such as in Cloud Source repositories. Although it can be in other repositories too.

15. GOOGLE BIG DATA PLATFORM

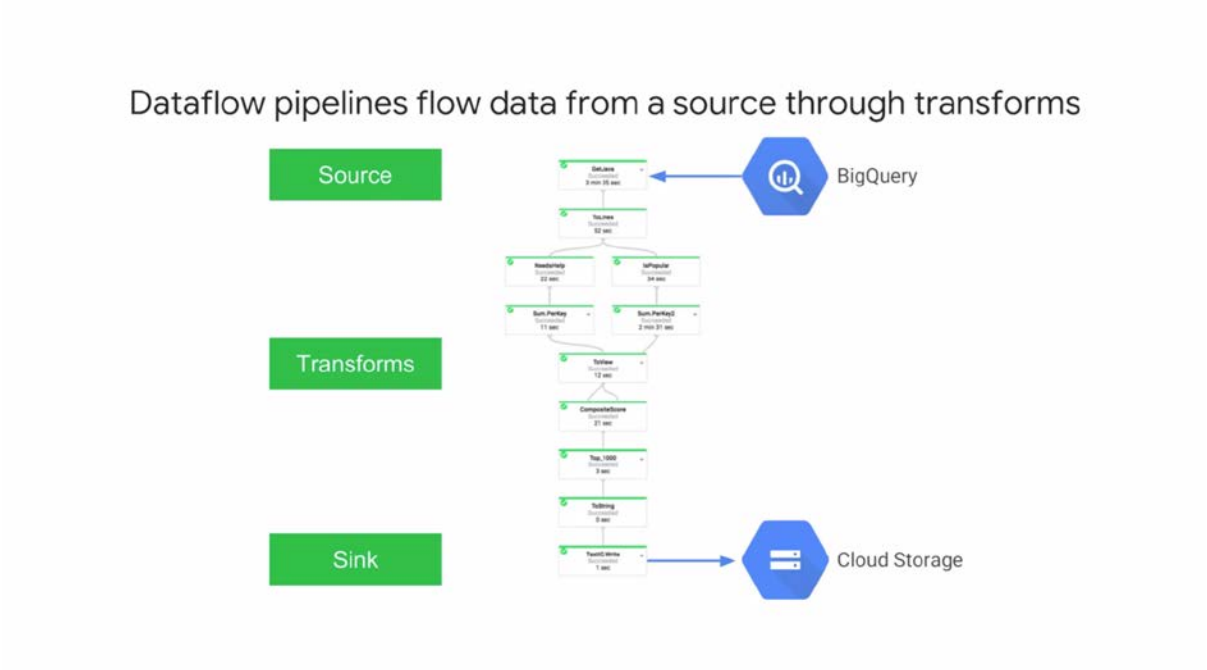


CLOUD DATAPROC

- Google Cloud Big Data Solutions are designed to help you transform your business and user experiences with meaningful data insights.
- Integrated Serverless Platform : Serverless means you don't have to worry about
- provisioning Compute Instances to run your jobs. The services are fully managed, and you pay only for the resources you consume. The platform is integrated, so that GCP data services work together to help you create custom solutions.
- Apache Hadoop is an open source framework for big data. It is based on the MapReduce programming model which Google invented and published. The MapReduce model is, at its simplest, means that one function, traditionally called the "Map function," runs in parallel with a massive dataset to produce intermediate results. And another function, traditionally called the "Reduce function," builds a final result set based on all those intermediate results.
- The term "Hadoop" is often used informally to encompass Apache Hadoop itself, and related projects such as Apache Spark, Apache Pig, and Apache Hive.
- **Cloud Dataproc** is a fast, easy, managed way to run Hadoop, Spark, Hive, and Pig on Google Cloud Platform. All you have to do is request a Hadoop cluster. It will be built for you in 90 seconds or less, on top of Compute Engine virtual machines whose number and type you control.
- If you need more or less processing power while your cluster is running, you can scale it up or down. You can use the default configuration for the Hadoop software in your cluster or you can customize it.
- And you can monitor your cluster using Stackdriver.
- Running on-premises, Hadoop jobs requires a capital hardware investment. Running these jobs in Cloud Dataproc, allows you to only pay for hardware resources used during the life of the cluster you create.
- Although the rate for pricing is based on the hour, Cloud Dataproc is billed by the second. Cloud Dataproc clusters are billed in one-second clock-time increments, subject to a one minute minimum billing. So, when you're done with your cluster, you can delete it, and billing stops. This is much more agile use of resources than on-premise hardware assets.
- You can also save money, by telling Cloud Dataproc to use preemptible Compute Engine instances for your batch processing. You have to make sure that your jobs can be restarted cleanly, if they're terminated, and you get a significant break in the cost of the instances.

- Be aware that the costs of the Compute Engine instances isn't the only component of the cost of a Dataproc cluster, but it's a significant one.
- Once your data is in a cluster, you can use Spark and Spark SQL to do data mining. And you can use MLib, which is Apache Spark's machine learning libraries to discover patterns through machine learning.

GOOGLE CLOUD BIG DATA– CLOUD DATAFLOW



- While Cloud Dataproc is great when you have a data set of known size or when you want to manage your cluster size yourself. But what if your data shows up in real time or it's of unpredictable size or rate?
- That's where **Cloud Dataflow** is particularly a good choice. It's both a unified programming model and a managed service and it lets you develop and execute a big range of data processing patterns: **extract, transform, and load** batch computation and **continuous computation.**
- You use Dataflow to build data pipelines. And the same pipelines work for both batch and streaming data. There's no need to spin up a cluster or to size instances.
- Cloud Dataflow fully **automates** the management of whatever processing resources are required.
- Cloud Dataflow frees you from operational tasks like resource management and performance optimization.
- *In this example*, Dataflow pipeline reads data from a big query table, the Source, processes it in a variety of ways, the Transforms, and writes its output to a cloud storage, the Sink. Some of those transforms you see here are map operations and some are reduce operations.
- You can build really expressive pipelines. Each step in the pipeline is elastically scaled. There is no need to launch and manage a cluster.
- Instead, the service provides all **resources on demand**. It has automated and optimized worked partitioning built in, which can dynamically rebalance lagging work. That reduces the need to worry about hotkeys. That is, situations where disproportionately large chunks of your input get mapped to the same cluster.
- People use Dataflow in a variety of use cases. As we've discussed, it's a general purpose ETL tool and its use case as a data analysis engine comes in handy in things like fraud detection and financial services, IoT analytics and manufacturing, healthcare and logistics and click stream, point of sale and segmentation analysis in retail.

- And because those pipelines, we saw can orchestrate multiple services even external services.
- It can be used in real time applications such as personalizing gaming user experiences.

GOOGLE CLOUD BIG DATA– BIGQUERY

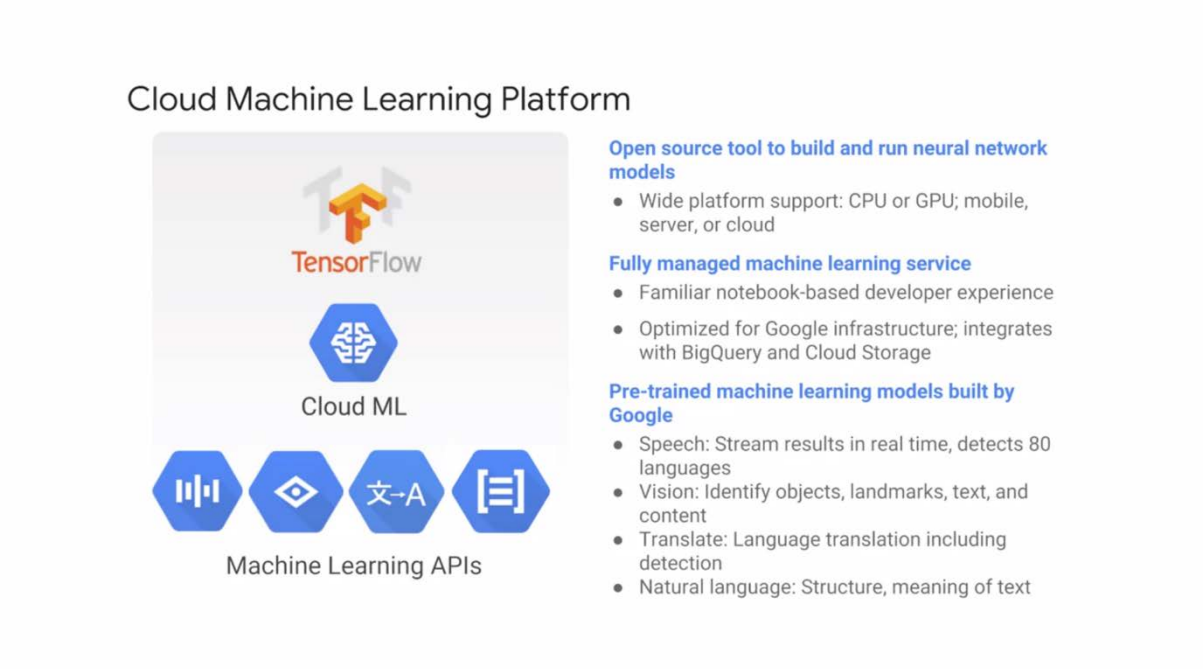
- Suppose, instead of a dynamic pipeline, your data needs to run more in the way of exploring a vast sea of data. You want to do **ad-hoc SQL queries** on a massive data set.
- That's what BigQuery is for. It's Google's fully-managed, petabyte-scale, low-cost analytics data warehouse.
- Because there's no infrastructure (eg : clusters, GCEs) to manage, you can focus on analyzing data to find meaningful insights, use familiar SQL and take advantage of our pay-as-you-go model. It's easy to get data into BigQuery.
- You can load it from cloud storage or cloud data store, or stream it into BigQuery at up to 100,000 rows per second. Once it's in there, you can run super-fast SQL queries against
- multiple terabytes of data in seconds using the processing power of Google's infrastructure.
- In addition to SQL queries, you can easily read and write data in BigQuery via Cloud Dataflow, Hadoop, and Spark.
- BigQuery is used by all types of organizations from startups to Fortune 500 companies - smaller organizations like Big Query's free monthly quotas, bigger organizations like its seamless scale, and it's available 99.9 percent service level agreement.
- Google's infrastructure is global and so is BigQuery. BigQuery lets you specify the region where your data will be kept.
- So, for example, if you want to keep data in Europe, you don't have to go set up a cluster in Europe. Just specify the EU location where you create your data set. US and Asia locations are also available.
- Because BigQuery separates storage and computation, you pay for your data storage separately from queries. That means, you pay for queries only when they are actually running.
- You have full control over who has access to the data stored in BigQuery, including sharing data sets with people in different projects.
- If you share data sets, that won't impact your cost or performance.
- People you share with pay for their own queries, not you.
- Long-term storage pricing is an automatic discount for data residing in BigQuery for extended periods of time. When the age of your data reaches 90 days in BigQuery, Google will automatically drop the price of storage.

GOOGLE CLOUD BIG DATA– CLOUD PUB/SUB

- Cloud Pub/Sub : a simple, reliable, scalable foundation for stream analytics. You can use it to let independent applications you build send and receive messages. That way they're decoupled, so they scale independently. The Pub in Pub/Sub is short for publishers and Sub is short for subscribers.
- Applications can publish messages in Pub/Sub and one or more subscribers receive them. Receiving messages doesn't have to be synchronous. That's what makes Pub/Sub great for decoupling systems. It's designed to provide "at least once" delivery at low latency. When we say "at least once delivery," we mean that there is a small chance some messages might be delivered more than once. So, keep this in mind when you write your application.
- Cloud Pub/Sub offers on-demand scalability to one million messages per second and beyond. You just choose the quota you want.
- Cloud Pub/Sub builds on the same technology Google uses internally.
- It's an important building block for applications where data arrives at high and unpredictable rates, like Internet of Things systems. If you're analyzing streaming data,

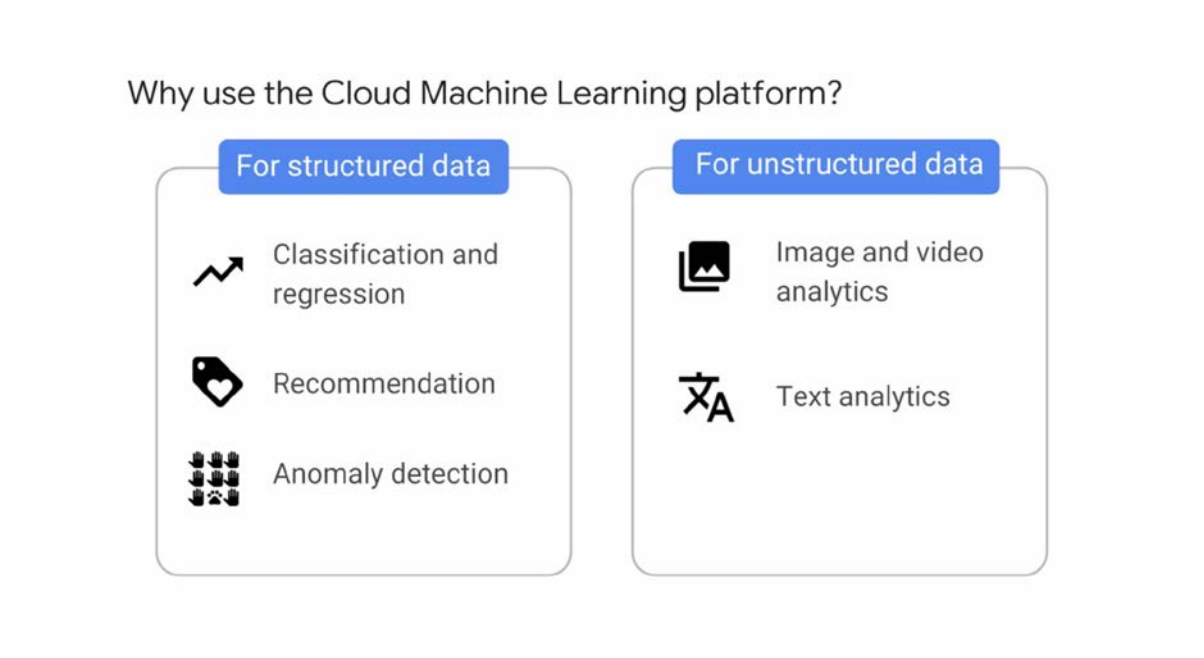
- Cloud Dataflow is a natural pairing with Pub/Sub. Pub/Sub also works well with applications built on GCP's Compute Platforms. You can configure your subscribers to receive messages on a push or pull basis.
- In other words, subscribers can get notified when new messages arrive for them or they can check for new messages at intervals.
- Scientists have long used lab notebooks to organize their thoughts and explore their data. For data science, the lab notebook metaphor works really well, because it feels natural to intersperse data analysis with comments about their results.
- A popular environment for hosting those is **Project Jupyter**. It lets you create and maintain web-based notebooks containing Python code and you can run that code interactively and view the results. And **Cloud Datalab** takes the management work out of this natural technique. It runs in a Compute Engine virtual machine.
- To get started, you specify the virtual machine type you want and what GCP region it should run in. When it launches, it presents an interactive Python environment that's ready to use. And it orchestrates multiple GCP services automatically, so you can focus on exploring your data.
- You only pay for the resources you use. There is no additional charge for Datalab itself.
- It's integrated with BigQuery, Compute Engine, and Cloud Storage, so accessing your data doesn't run into authentication hassles.
- When you're up and running, you can visualize your data with **Google Charts** or **map plot line** and because there's a vibrant interactive Python community, you can learn from published notebooks. There are many existing packages for statistics, machine learning, and so on.

16. GOOGLE CLOUD MACHINE LEARNING



- Major Google applications use machine learning like YouTube, Photos, the Google Mobile App and Google Translate.
- The Google Machine Learning Platform is now available as a cloud service so that you can add innovative capabilities to your own applications.
- Cloud Machine Learning Platform provides modern machine learning services with pre-trained models and a platform to generate your own tailored models. As with other GCP products, there's a range of services that stretches from the highly general to the pre-customized.
- **TensorFlow** is an open source software library that's exceptionally well suited for machine learning applications like neural networks. It was developed by Google Brain for Google's internal use and then open source so that the world could benefit.

- You can run TensorFlow wherever you like but GCP is an ideal place for it because machine learning models need lots of on-demand compute resources and lots of training data. TensorFlow can also take advantage of **Tensor Processing Units**, which are hardware devices designed to accelerate machine learning workloads with TensorFlow.
- GCP makes them available in the cloud with Compute Engine virtual machines. Each cloud TPU provides up to 180 teraflops of performance.
- And because you pay for only what you use, there's no upfront capital investment required.
- Suppose you want a more managed service. Google Cloud Machine Learning Engine lets you easily build machine learning models that work on any type of data of any size. It can take any TensorFlow model and perform large-scale training on a managed cluster.
- Finally, suppose you want to add various machine learning capabilities to your applications without having to worry about the details of how they are provided. Google Cloud also offers a range of machine learning APIs suited to specific purposes.
- People use the Cloud Machine Learning Platform for lots of applications. Generally, they fall into two categories, depending on whether the data they work on is structured or unstructured.
- Based on **structured data**, you can use ML for various kinds of classification and regression tasks like customer churn analysis, product diagnostics and forecasting. It can be the heart of a recommendation engine for content personalization and cross-sells and up-sells. You can use ML to detect anomalies, as in fraud detection, sensor diagnostics or log metrics.
- Based on **unstructured data**, you can use ML for image analytics such as identifying damaged shipment, identifying styles and flagging content. You can do text analytics too, like a call center, blog analysis, language identification, topic classification and sentiment analysis.
- In many of the most innovative applications for machine learning, several of these kinds of applications are combined.
- What if whenever one of your customers posted praise for one of your products on social media, your application could automatically reach out to them with a customized discount on another product they'll probably like? The Google Cloud Machine Learning Platform makes
- that kind of interactivity well within your grasp.



IMAGE, LANGUAGE, VIDEO AND AUDIO INTELLIGENCE

- The **Cloud Vision API** enables developers to understand the content of an image. It quickly lassifies images into thousands of categories - sailboat, lion, Eiffel Tower - detects individual objects within images, and finds and reads printed words contained within images.
  It can do the following stuff :
  a. Logo Detection, label detection etc.
  b. Gain insights from images
  c. Detect inappropriate content
  d. Analyze sentiment
  e. Extract text
  Like the other APIs I'm describing here, encapsulates powerful machine learning models behind an easy-to-use API. You can use it to build metadata on your image catalog, moderate offensive content or even do image sentiment analysis.
- The **Cloud Speech API** enables developers to convert audio to text. Because you have an increasingly global user base, The API recognizes over 80 languages and variants. You can transcribe the text of users, dictating in an applications' microphone, enable command and control through voice or transcribe audio files.
- The **Cloud Natural Language API** offers a variety of natural language understanding technologies to developers.
  It's some functions include :
  a. Can return text in real time
  b. Highly accurate, even in noisy environments
  c. Access from any device
  d. Use ML models to reveal structure and meaning of text
  e. Extract information about items mentioned in text docs, news articles and blog posts
  It can do syntax analysis, breaking down sentences supplied by our users into tokens, identify the nouns, verbs, adjectives, and other parts of speech and figure out the relationships among the words.
  It can do entity recognition. In other words, it can parse text and flag mentions of people, organizations, locations, events, products, and media.
  It can understand the overall sentiment expressed in a block of text.
  It has these capabilities in multiple language, including English, Spanish, and Japanese.
- Cloud **Translation API** provides a simple, programmatic interface for translating an arbitrary string into a supported language. When you don't know the source language, the API can detect it.
  Functions :
  a. Translate arbitrary strings between 1000s of language pairs
  b. Programmatically detect a doc's language
  c. Supports for dozens of languages
- The **Cloud Video Intelligence API** lets you annotate videos in a variety of formats. It helps you identify key entities - that is, nouns - within your video and when they occur. You can use it to make video content searchable and discoverable.
  a. Annotate the contents of videos
  b. Detect scene changes
  c. Flag inapt content
  d. Support for a variety of video formats

Comparing compute options

	Compute Engine	Kubernetes Engine	App Engine Flex	App Engine Standard	Cloud Functions[Beta]
Service model	IaaS	Hybrid	PaaS	PaaS	Serverless
Use cases	General computing workloads	Container-based workloads	Web and mobile applications; container-based workloads	Web and mobile applications	Ephemeral functions responding to events

Toward managed infrastructure *Toward dynamic infrastructure*

Comparing load-balancing options

Global HTTP(S)	Global SSL Proxy	Global TCP Proxy	Regional	Regional internal
Layer 7 load balancing based on load	Layer 4 load balancing of non-HTTPS SSL traffic based on load	Layer 4 load balancing of non-SSL TCP traffic	Load balancing of any traffic (TCP, UDP)	Load balancing of traffic inside a VPC
Can route different URLs to different back ends	Supported on specific port numbers	Supported on specific port numbers	Supported on any port number	Use for the internal tiers of multi-tier applications

Comparing interconnect options

VPN

Secure multi-Gbps connection over VPN tunnels

Direct Peering

Private connection between you and Google for your hybrid cloud workloads

Carrier Peering

Connection through the largest partner network of service providers

Dedicated Interconnect

Connect N X 10G transport circuits for private cloud traffic to Google Cloud at Google POPs

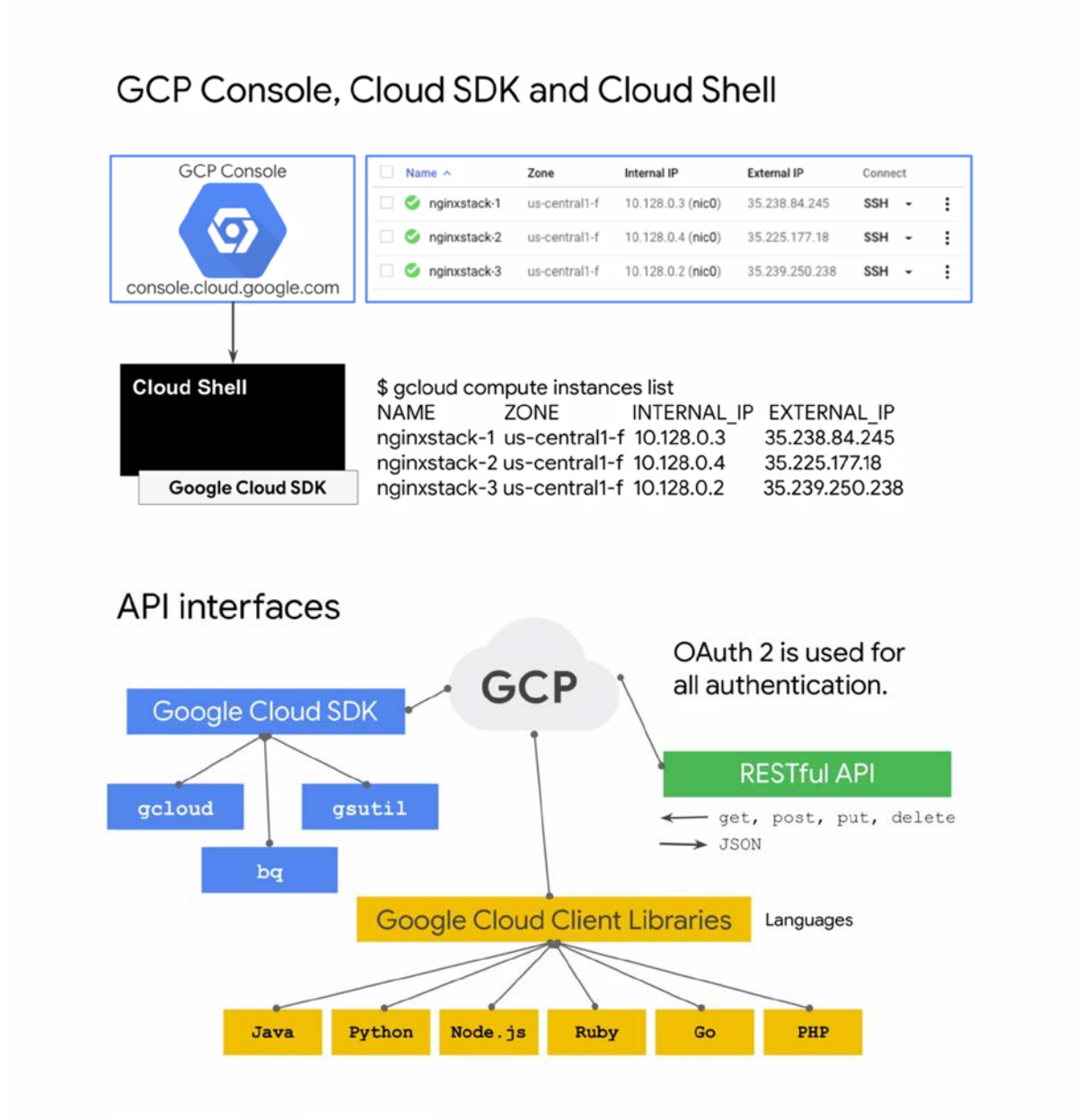
Comparing storage options

	Cloud Datastore	Cloud Bigtable	Cloud Storage	Cloud SQL	Cloud Spanner	BigQuery
Type	NoSQL document	NoSQL wide column	Blobstore	Relational SQL for OLTP	Relational SQL for OLTP	Relational SQL for OLAP
Best for	Getting started, App Engine applications	"Flat" data, Heavy read/write, events, analytical data	Structured and unstructured binary or object data	Web frameworks, existing applications	Large-scale database applications (> ~2 TB)	Interactive querying, offline analytics
Use cases	Getting started, App Engine applications	AdTech, Financial and IoT data	Images, large media files, backups	User credentials, customer orders	Whenever high I/O, global consistency is needed	Data warehousing

Choosing among Google Cloud Storage classes

	Multi-regional	Regional	Nearline	Coldline
Intended for data that is...	Most frequently accessed	Accessed frequently within a region	Accessed less than once a month	Accessed less than once a year
Availability SLA	99.95%	99.90%	99.00%	99.00%
Access APIs	***Consistent APIs***			
Access time	***Millisecond access***			
Storage price	Price per GB stored per month			
Retrieval price				Total price per GB transferred
Use cases	Content storage and delivery	In-region analytics, transcoding	Long-tail content, backups	Archiving, disaster recovery

17. **Deployment Manager** is a Google Cloud service that uses templates written in a combination of YAML, python, and Jinja2 to automate the allocation of Google Cloud resources and perform setup tasks. Behind the scenes a virtual machine has been created. A startup script was used to install and configure software, and network Firewall Rules were created to allow traffic to the service.



18. GCP VPC

- Projects are the key organizer of infrastructure resources in GCP. A project associates objects and services with billing. Now, it's unique that projects actually contain entire networks. The default quota for each project is five networks but you can simply request additional quota using the GCP console. These networks can be shared with other projects or they can be peered with networks in other projects.

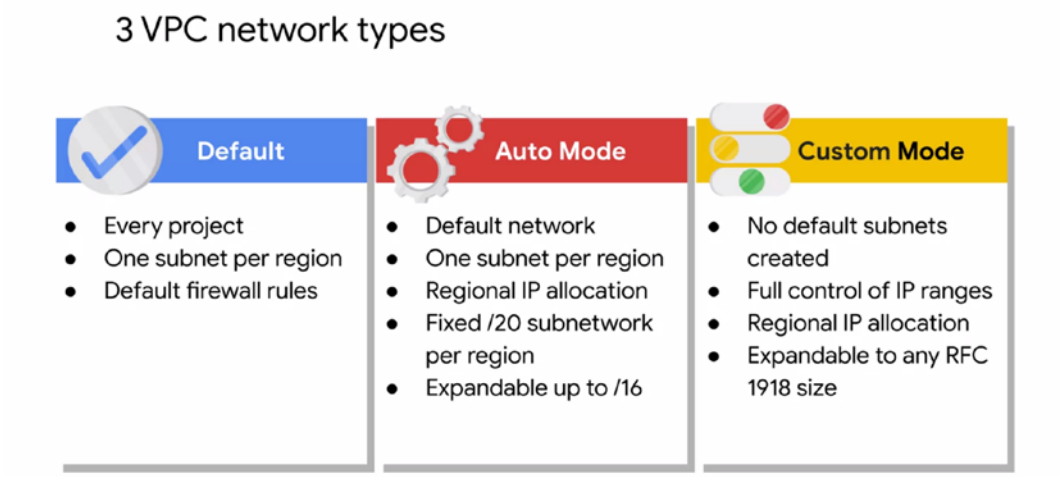
- These networks do not have IP ranges but are simply a construct of all of the individual IP addresses and services within that network. GCP networks are global spending all available regions across the world that I showed earlier. So you can have one network that later exists anywhere in the world, Asia, Europe, Americas, all simultaneously.

- Inside a network you can segregate your resources with regional subnetworks.

- There are different types of networks: **default, auto, and custom.**

- Every project is provided with a default VPC network with presets subnets and firewall rules.

- Specifically a subnet is allocated for each region with non-overlapping CIDR blocks and firewall rules that allow ingress traffic from ICMP, RDP, and SSH traffic from anywhere, as well as ingress traffic from within the default network for all protocols and ports.



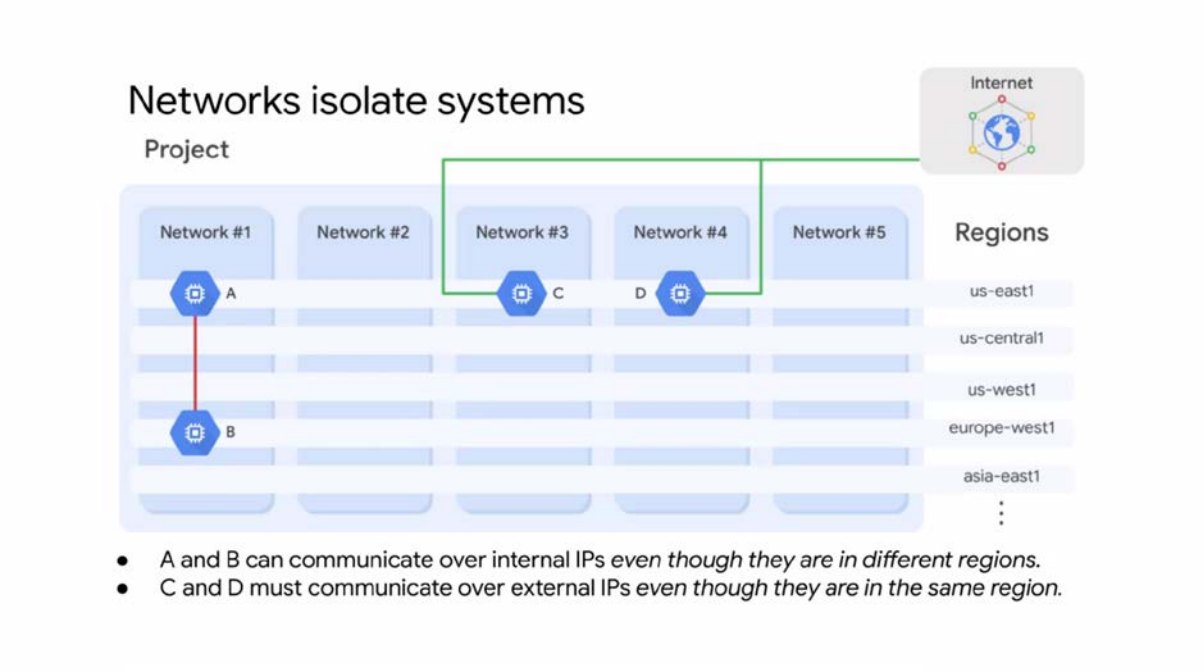
- In an **Auto Mode** network, one subnets from each region is automatically created within it. The default network is actually an auto mode network. These automatically created subnets uses set of predefined IP ranges with a /20 mask that can be expanded to a /16. All of these subnets fit within the 10.128.0.0/9 cider block.

- Therefore, as new GCP regions become available, new subnets and dose regions are automatically added to automotive networks using an IP range from that block.

- A Custom Mode network does not automatically create subnets. This type of network provides you with complete control over its subnets and IP ranges. You decide which subnets to create in regions you choose and using IP ranges you specify within the RFC 1918 address space. These IP ranges cannot overlap between subnets of the same network.

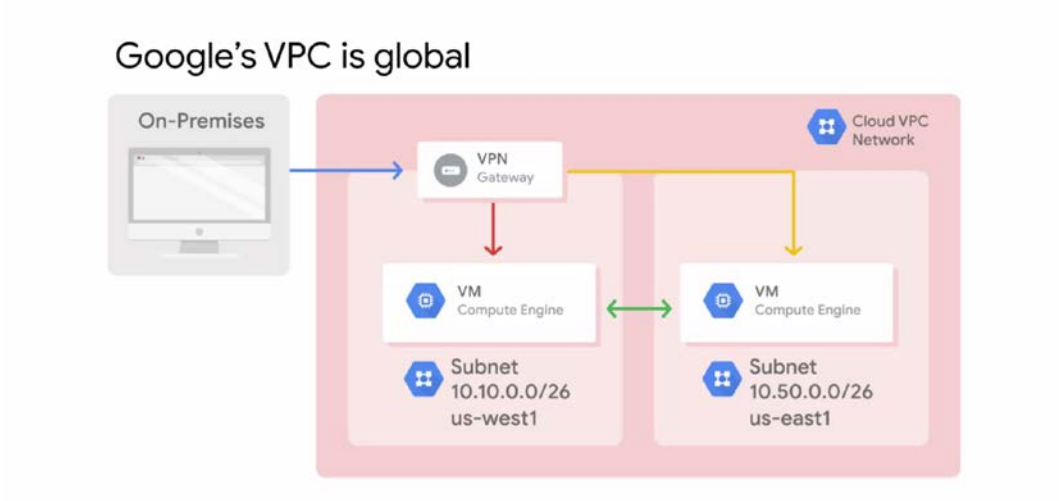
- Now, you can convert an auto mode network to a custom mode network to take advantage of the control that custom mode networks provide. However, this conversion is one way. Meaning that custom mode networks cannot be changed to auto mode nodes. So carefully review the conCIDRation for auto mode networks to help you

- decide which type of network meets your needs.



- Example : a project that contains five networks. All of these networks span multiple regions across the world as you can see on the right. Each network contains separate virtual machines: A, B, C, and D. Because VM's A and B are in the same network, Network 1, they can communicate using their internal IP address even though they are in different regions.

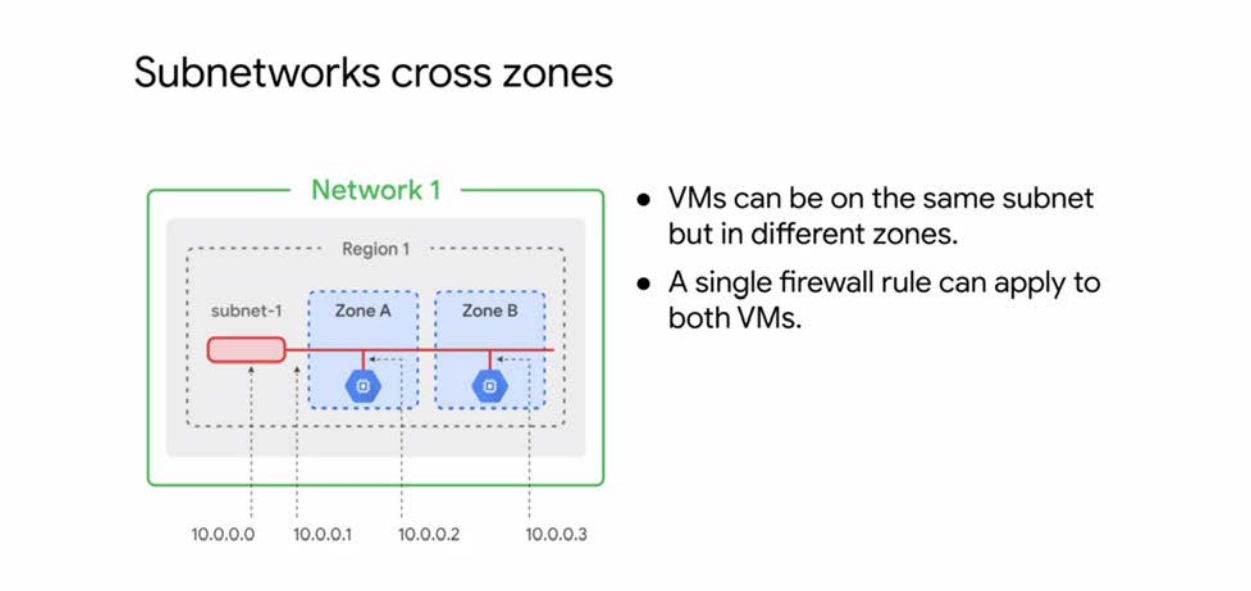
- Essentially your virtual machines even if they exist in different locations across the world, take advantage of Google's global fiber network. Those Virtual Machines appear as though they're sitting in the same rack, when it comes to a network configuration protocol. VM C and D however are not in the same network. Therefore by default these VM's must communicate using their external IP addresses even though they are in the same region. The traffic between VM C and D isn't actually touching the public Internet but is going through the Google edge routers. This has different billing and security ramifications



- Because VM instances within a VPC network can communicate privately on a global scale, a single VPN can securely connect your on-premises network to a GCP network as shown in this diagram.

- Even though the two VM instances are in separate regions, US-West 1 and US-East 1, they leverage Google's private network to communicate between each other and to an on-premises network through a VPN gateway. This reduces cost and network management complexity.

- I mentioned that subnetworks work on a regional scale. Because a region contains several zones, subnetworks can cross zones.



- This slide has a region, Region 1 with two zones: zones A and B. Subnetworks can extend across these zones within the same region such as subnet-1. The subnet is simply an IP address range and you can use IP addresses within that range. Notice that the first and second addresses in the range 10.0.0.0 and 10.0.0.1 are reserved for the network and these subnets gateway respectively. This makes the first and second available addresses 10.0.0.2 and 10.0.0.3 which are assigned to the VM instances.

- The other reserved addresses in every subnets are the second-to-last address in the range and the last address which is reserved as the broadcast address. So to summarize, every subnet has four reserved IP addresses in its primary IP range.

- Now, even though the two Virtual Machines in this example are in different zones, they still communicate with each other using the same subnet IP address. This means that a single firewall rule can be applied to both VM's even though they are in different zones.

- Speaking of IP addresses of a subnet, Google Cloud VPC's let you increase the IP address space of any subnets without any workload shutdown or downtime.

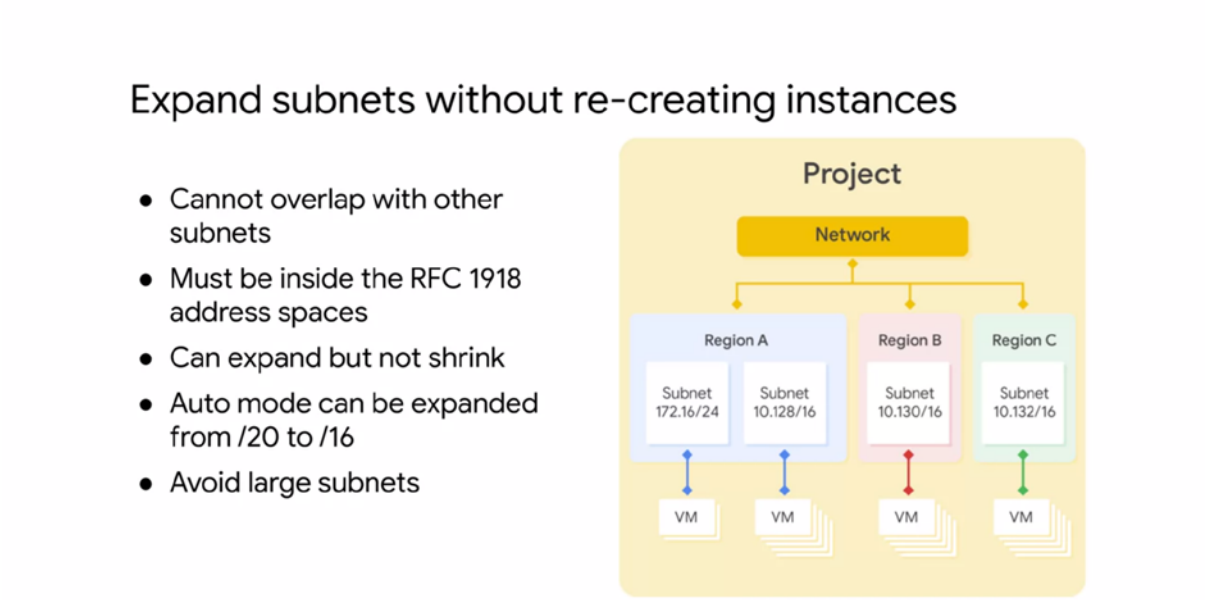
- This diagram illustrates a network with subnets that have different subnet masks allowing for more instances in some subnets than others.

- This gives you flexibility and growth options to meet your needs but there are some things to remember. The new subnet must not overlap with other subnets in these same VPC network in any region.

- Also, the new subnets must stay inside the RFC 1918 address spaces. The new network range must be larger than the original which means the prefix length value must be a smaller number. In other words, you cannot undo an expansion.

- Now, auto mode subnets start with a /20 IP range. They can be expanded to a /16 IP range but no larger. Alternatively, you can convert the auto mode subnetwork to a custom mode subnetwork to increase IP range further.

- Also avoid creating large subnets. Overly large subnets are more likely to cause site arrange collisions when using multiple network interfaces and VPC network peering or when configuring a VPN or other connections to an on-premises network. Therefore, do not scale your subnet beyond what you actually need.



- In GCP, each virtual machine can have two IP addresses assigned.

- One of them is an internal IP address, which is going to be assigned via DHCP internally. Every VM that starts up and any service that depends on virtual machines gets an internal IP address.

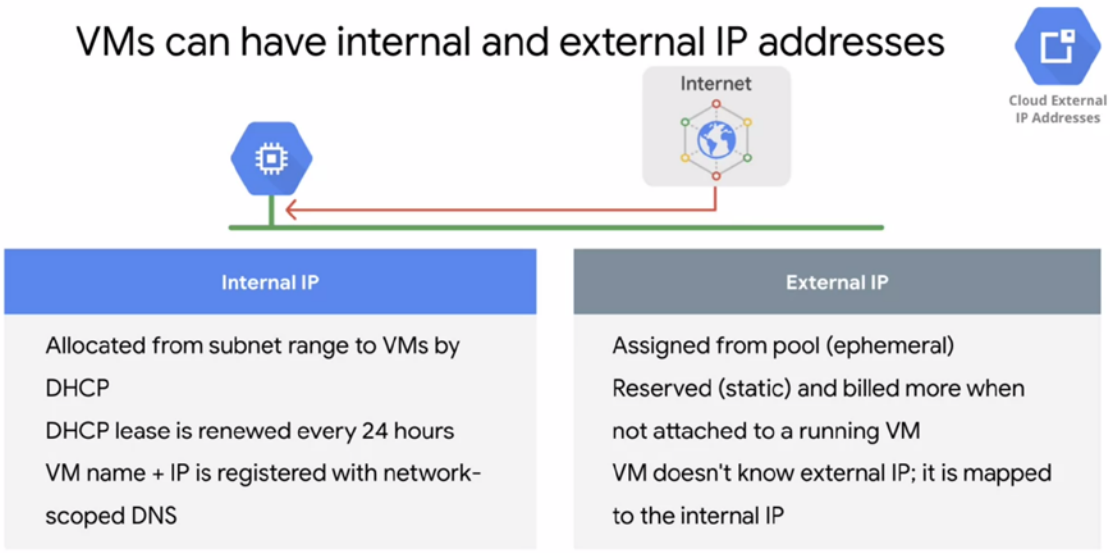
- Example of such services are App Engine and Kubernetes Engine, which are explored in other courses.

- When you create a VM in GCP, it's symbolic name is registered with an internal DNS service that translates the name to the internal IP address.

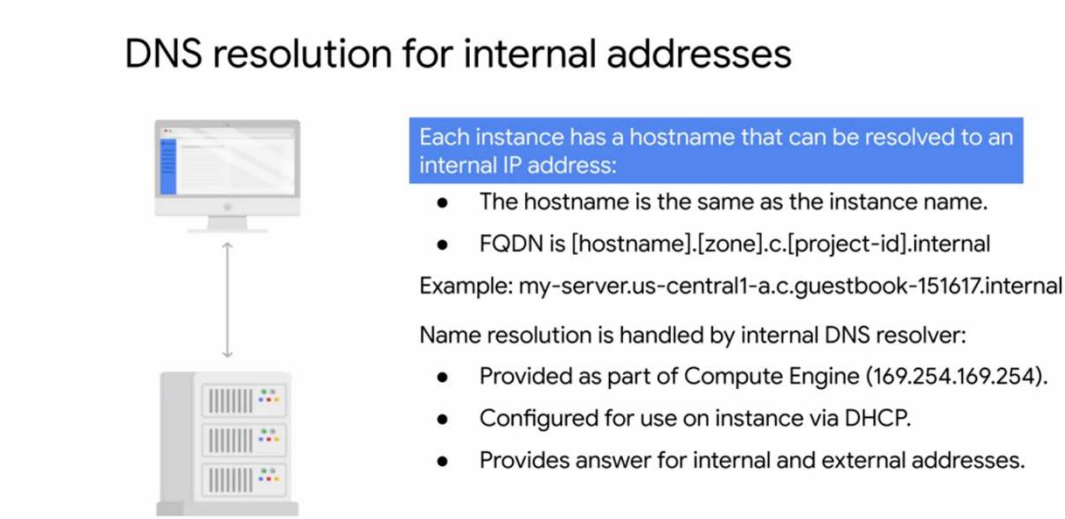
- DNS is scoped to the network, so it you can translate web URLs and VM names of hosts in the same network, but it can't translate host names from VMs in a different network.

- The other IP address is the external IP address, but this one is optional.

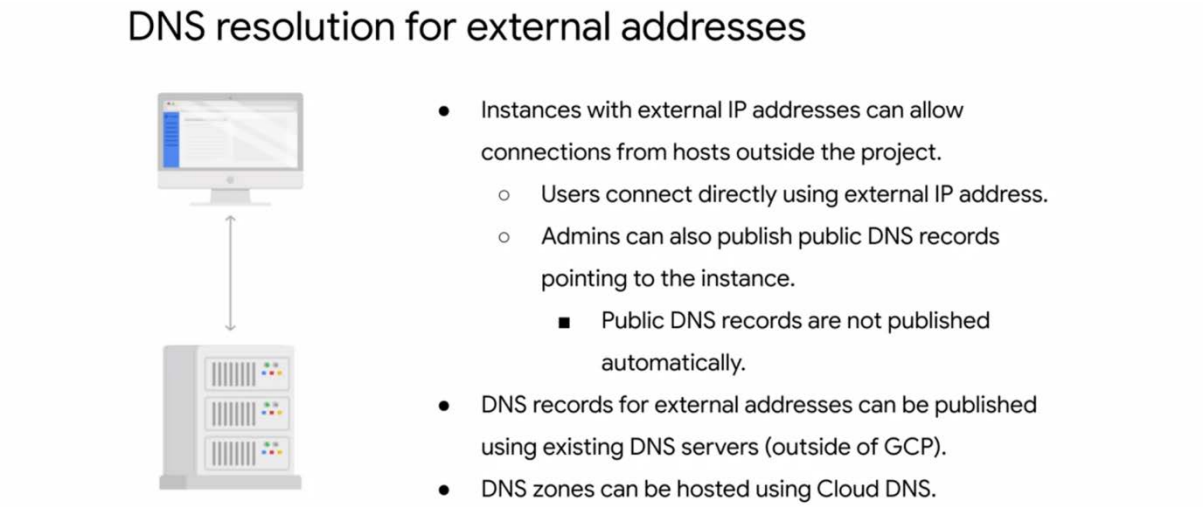
- You can assign an external IP address if your device or your machine is externally facing. That external IP address can be assigned from a pool, making it ephemeral, or it can be assigned a reserved external IP address, making it static. If you preserve a static external IP address and do not assign it to a resource such as a VM instance or a forwarding rule, you are charged at a higher rate than for static and ephemeral external IP addresses that are in use.



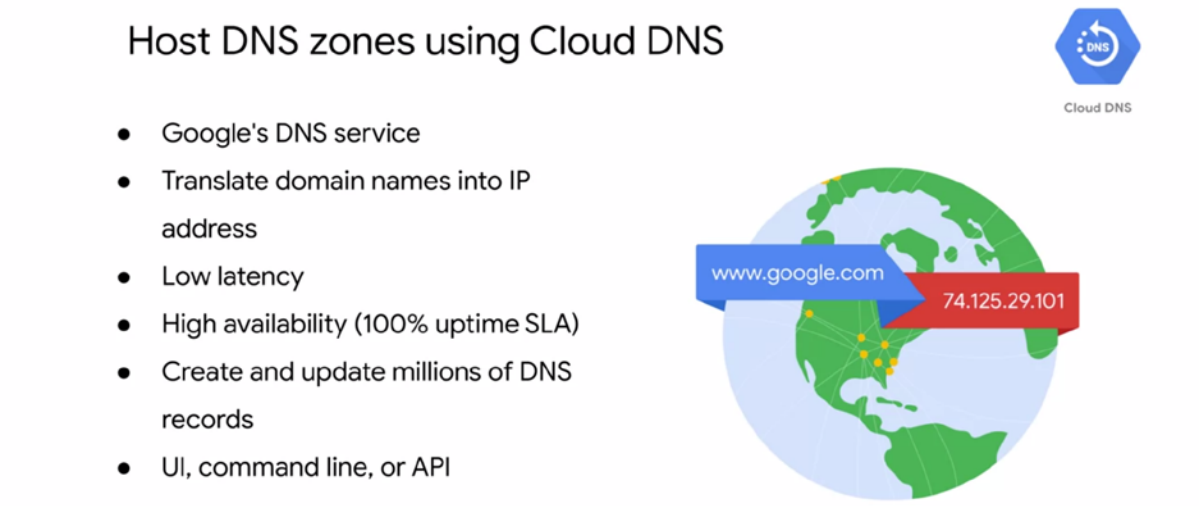
- Regardless of whether you use an ephemeral or static IP address, the external address is unknown to the OS of the VM. The external IP address is mapped to the VMs internal address transparently by VPC.



- **Internal addresses** : each in Instance has a host name that can be resolved to an internal IP address. This hostname is the same as the instance name.
- There's also an internal fully qualified domain name or fqdn for an instance that uses the format shown.
- If you delete and recreate an instance, the internal IP address can change. This change can disrupt connections from other compute engine resources, which must obtain the new IP address before they can connect again. However, the DNS name always points to specific instance no matter what the internal IP address is.
- Each instance has a metadata server that also acts as a DNS resolver for that instance.
- The metadata server handles all DNS queries for local network resources and routes all other queries to Google's public DNS servers for public name resolution.
- Instance is not aware of any external IP address assigned to it. Instead, the network stores a lookup table that matches external IP addresses with the internal IP addresses of the relevant instances.

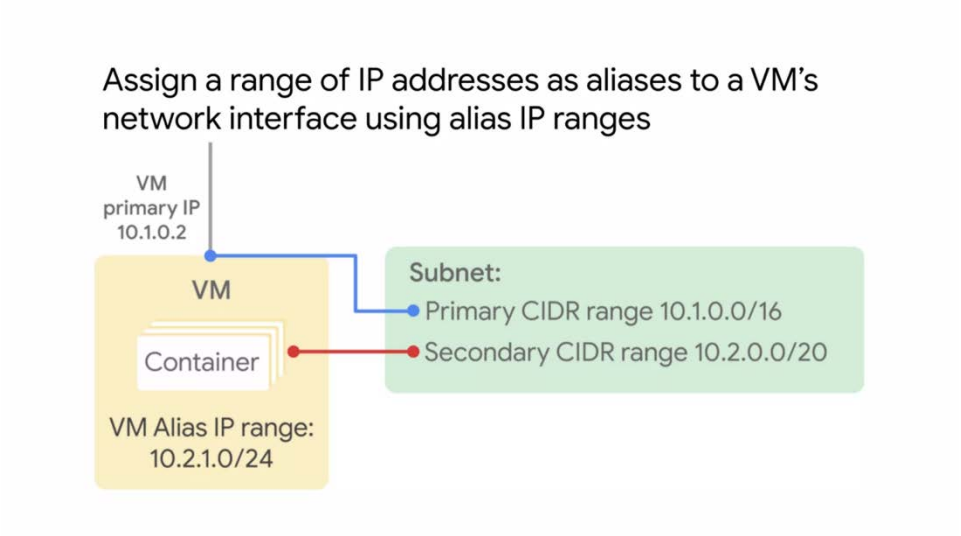


- **External Addresses** : Instances with external IP addresses can allow connections from hosts outside of the project. Users can do so directly using the external IP address. Public DNS records pointing to instances are not published automatically. However, admins can publish these using existing DNS servers. Domain and servers can be hosted on gcp using Cloud DNS. This is a managed service that's definitely worth conCIDRing



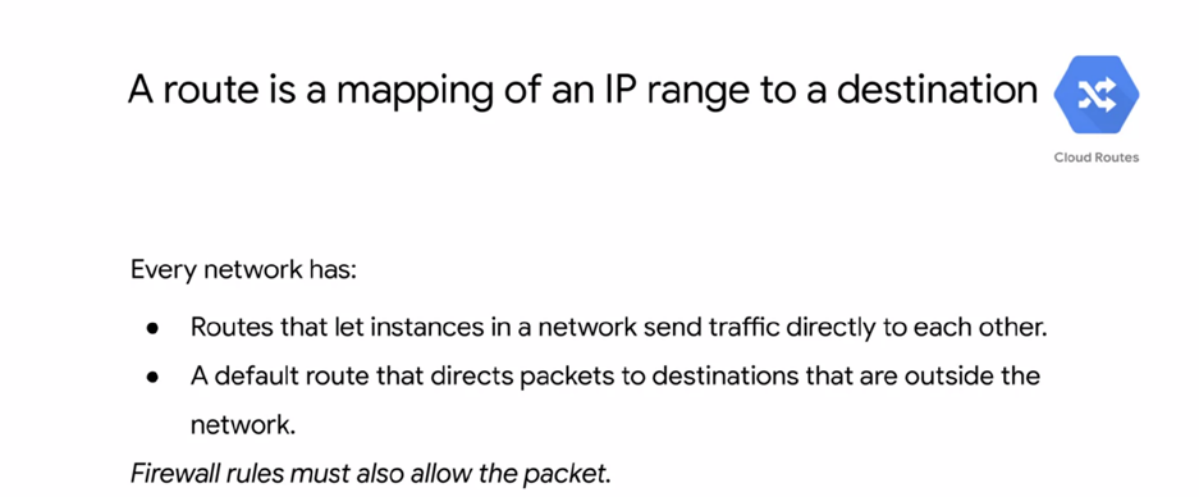
- **Cloud DNS** is a scalable, reliable and managed authoritive domain name system or DNS service running on the same infrastructure as Google. Cloud DNS translate requests for domain names like google.com into IP addresses. Cloud DNS uses Google's Global Network of any cast name servers to serve your DNS zones from a download locations around the world providing lower latency and high availability for your users.
- High availability is very important because if you can't look up a domain name the internet might as well be down. That's why gcp offers a 100% up-time service level agreement or SLA for domains configured in cloud DNS.
- Cloud DNS lets you create and update millions of DNS records without the burden of managing your own DNS service and software. Instead, you use a simple user interface, command line face or API.
- Another networking feature of gcp is alias IP ranges, alias IP ranges that you assign a range of internal addresses as an alias to Virtual machine’s network interface. This is useful if you have multiple services running on a VM and you want to assign a different IP address to each service. In essence, you can configure multiple IP addresses representing containers or applications hosted in a VM.

- Without having to define a separate network interface. You just draw the alias IP range from the local subnets primary or secondary side arranges.



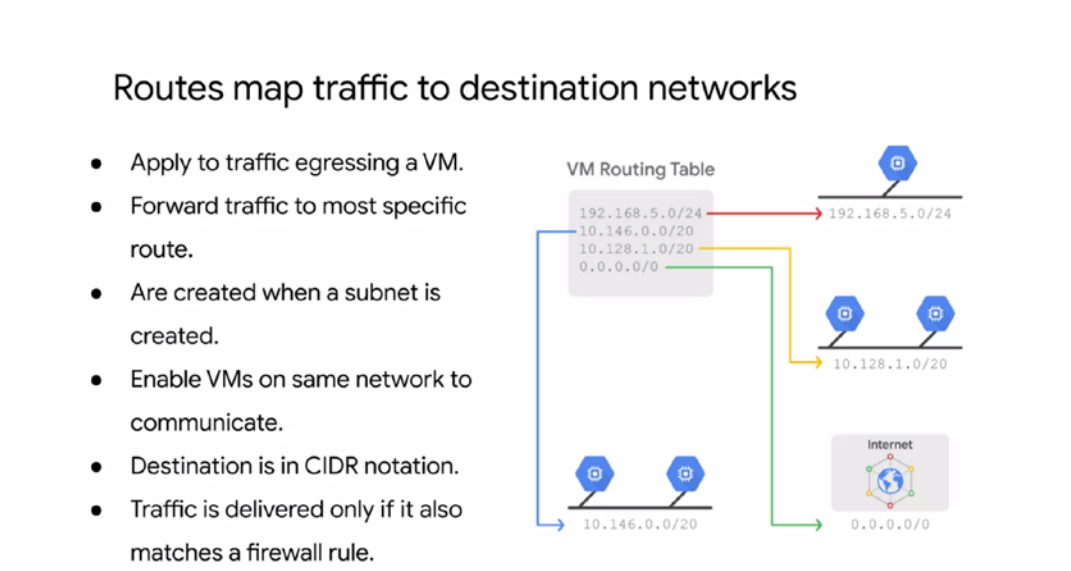
19. ROUTES AND FIREWALL RULES

By default, every network has routes that let instances in a network send traffic directly to each other even across subnets. In addition, every network has a default route that directs packets to destinations that are outside the network.



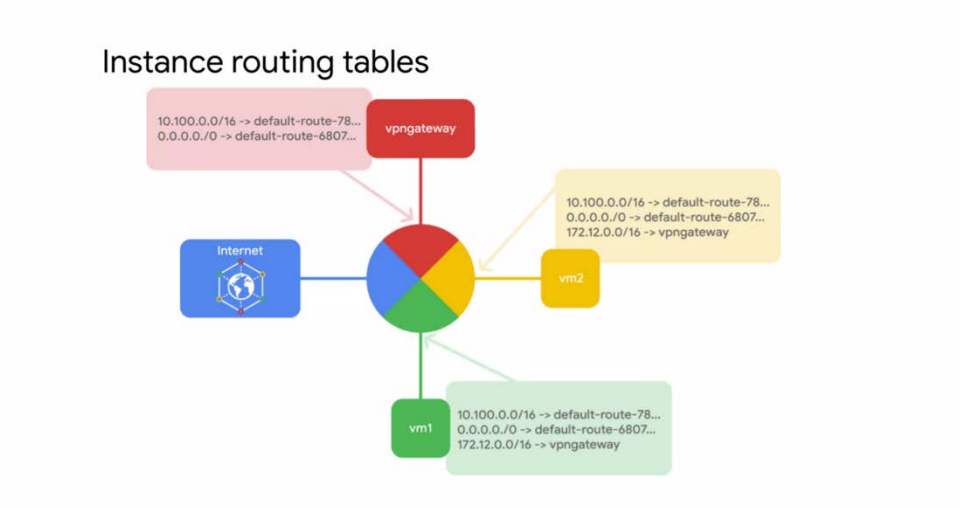
- Although these routes cover most of your normal routing needs, you can also create special routes that overwrite these routes.
- Just creating a route does not ensure that your packet will be received by the specified next top. Firewall rules must also allow the packet. The default network has preconfigured firewall rules that allow all instances in the network to talk with each other.
- Manually created networks do not have such rules, so you must create them
- Routes match packets by destination IP addresses. However, no traffic will flow without also matching a firewall rule.

- A route is created when a network is created, enabling traffic delivery from anywhere. Also, a route is created when a subnet is created. This is what enables VM's on the same network to communicate.



- Each route in the routes collection may apply to one or more instances. A route applies to an instance if the network and instance tags match. If the network matches and there are no instance tags specified, the route applies to all instances in that network.

- Compute engine then uses the routes collection to create individual read-only routing tables for each instance.
- This diagram shows a massively scalable virtual router at the core of each network.



- Every virtual machine instance in the network is directly connected to this router, and all packets leaving a virtual machine instance are first handled at this layer before they are forwarded to the next hop. The virtual network router selects the next hop for a packet by consulting the routing table for that instance.

- GCP firewall rules to protect you virtual machine instances from unapproved connections both inbound and outbound known as ingress and egress respectively.

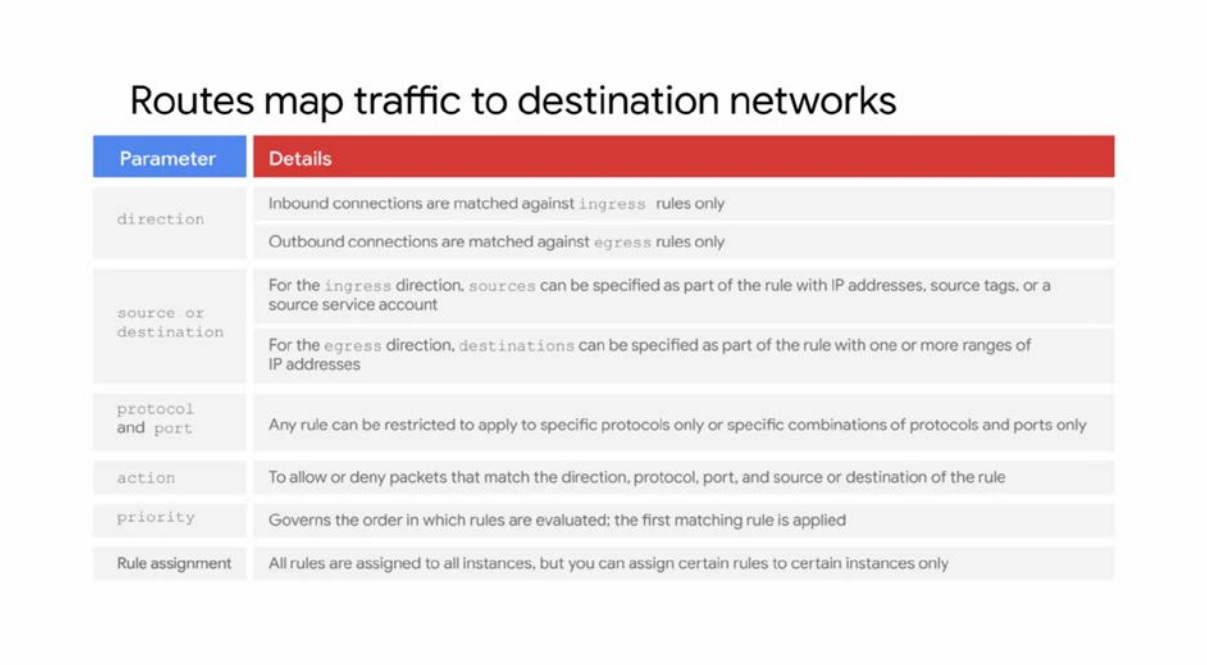
- Essentially, every VPC network functions as a distributed firewall. Although firewall rules are applied to the network as a whole, connections are allowed or denied at the instance level.

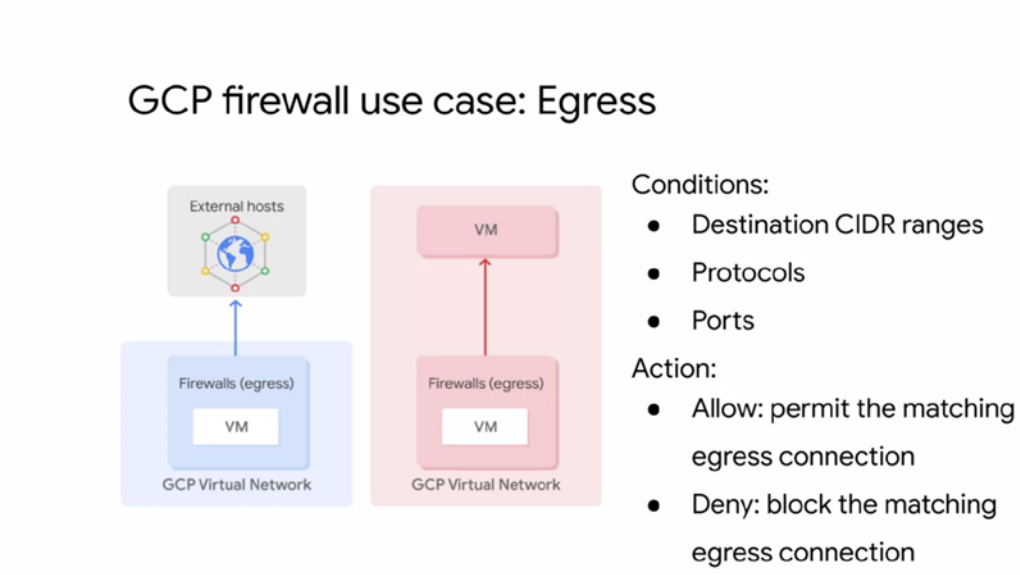
- You can think of the firewall as existing not only between your instances and other networks, but between individual instances within the same network.

- GCP firewall rules are stateful. This means that if a connection is allowed between a source and a target or a target at a destination, all subsequent traffic in either direction will be allowed. In other words, firewall rules allow bidirectional communication once a session is established.

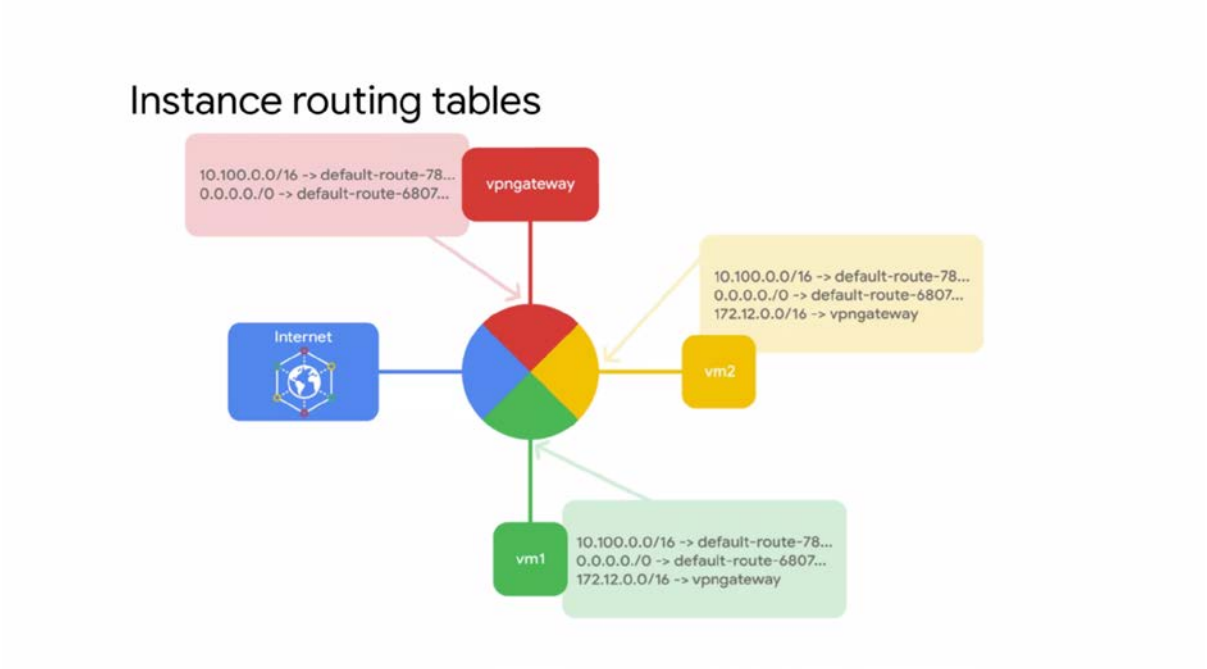
- Also if for some reason all firewall rules in a network are deleted, there is still an implied deny all ingress rule and an implied allow all egress rule for the network.

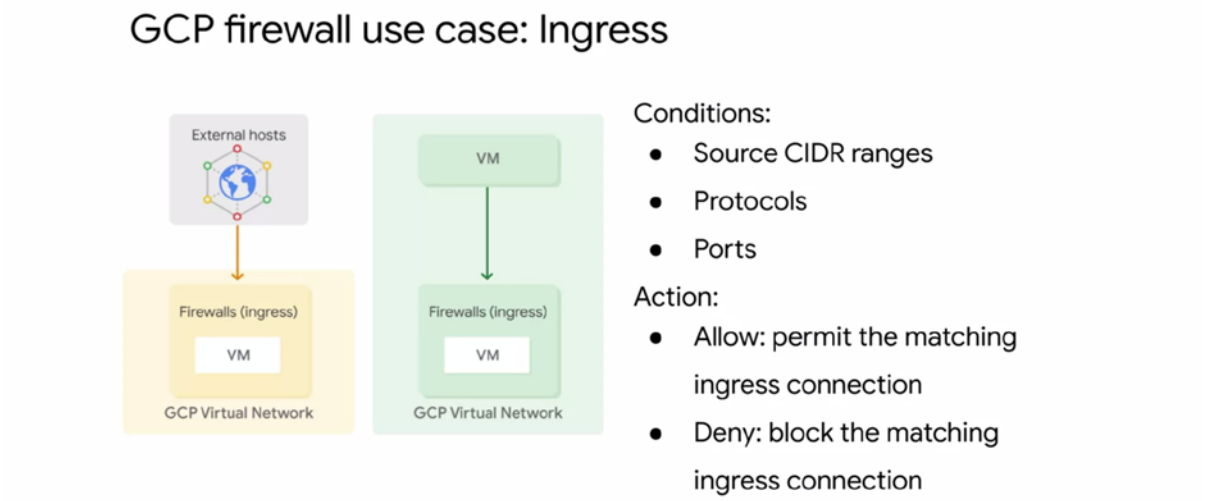
- You can express your desired firewall configuration as a set of firewall rules. Conceptually, a firewall rule is composed of the following parameters: the direction of the rule.





- Egress firewall rules control outgoing connections originated inside your GCP network.
- Egress allow rules allow outbound connections that match
- specific protocol ports and IP addresses. Egress deny rules prevent instances from initiating
- connections that match non permitted port protocol and IP range combinations.
- For egress firewall rules, destinations to which a rule applies may be specified using IP CIDR ranges. Specifically, you can use the destination ranges to protect from undesired connections initiated by a VM instance towards an external host
- You can also use destination ranges to prevent undesired connections from internal VM instances to specific GCP CIDR ranges.





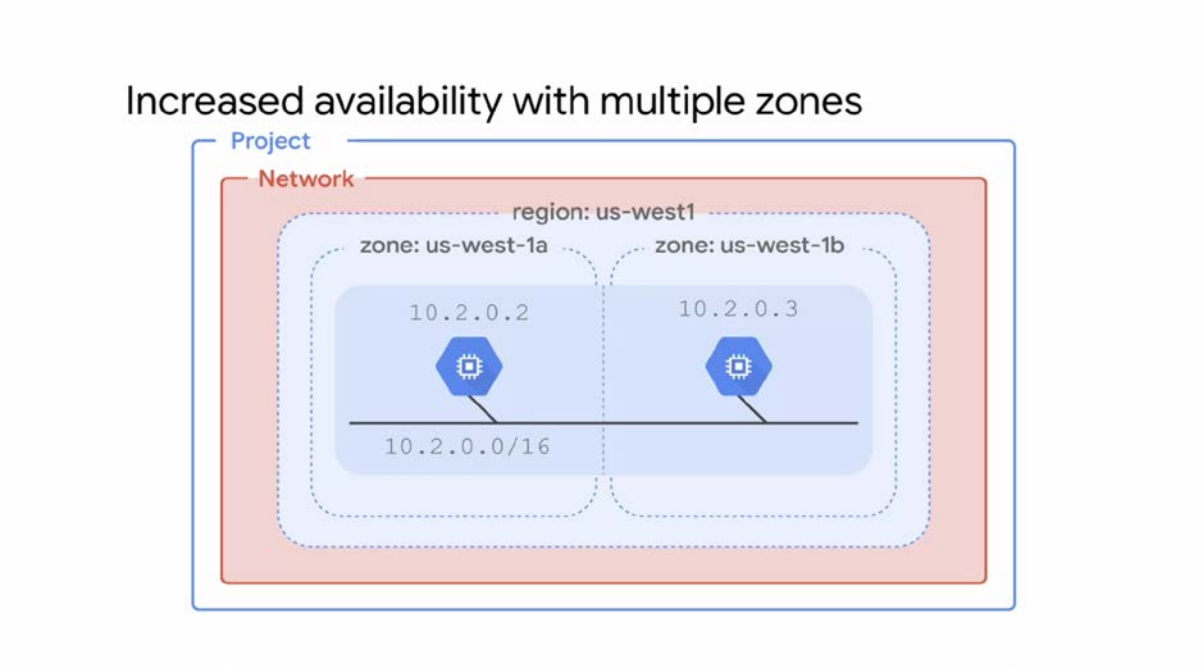
- Ingress firewall rules protect against incoming connections to the instance from any source.
- Ingress allow rules allow specific protocol ports and IP ranges to connecting.
- The firewall prevents instances from receiving connections on non-permitted ports and protocols.
- Rules can be restricted to only affect particular sources.
- Source CIDR ranges can be used to protect an instance from undesired connections coming either from external networks or from GCP IP ranges.

- You can control ingress connections from a VM instance by constructing inbound connection conditions using source CIDR ranges, protocols, or ports.

NETWORKING PRICING

- The pricing calculator is a web-based tool, that you use to specify the expected consumption of certain services and resources, and it then provides you with an estimated cost.
- You can adjust the currency and time frame to meet your needs, and when you finish, you can e-mail the estimate or save it to a specific URL for future reference.

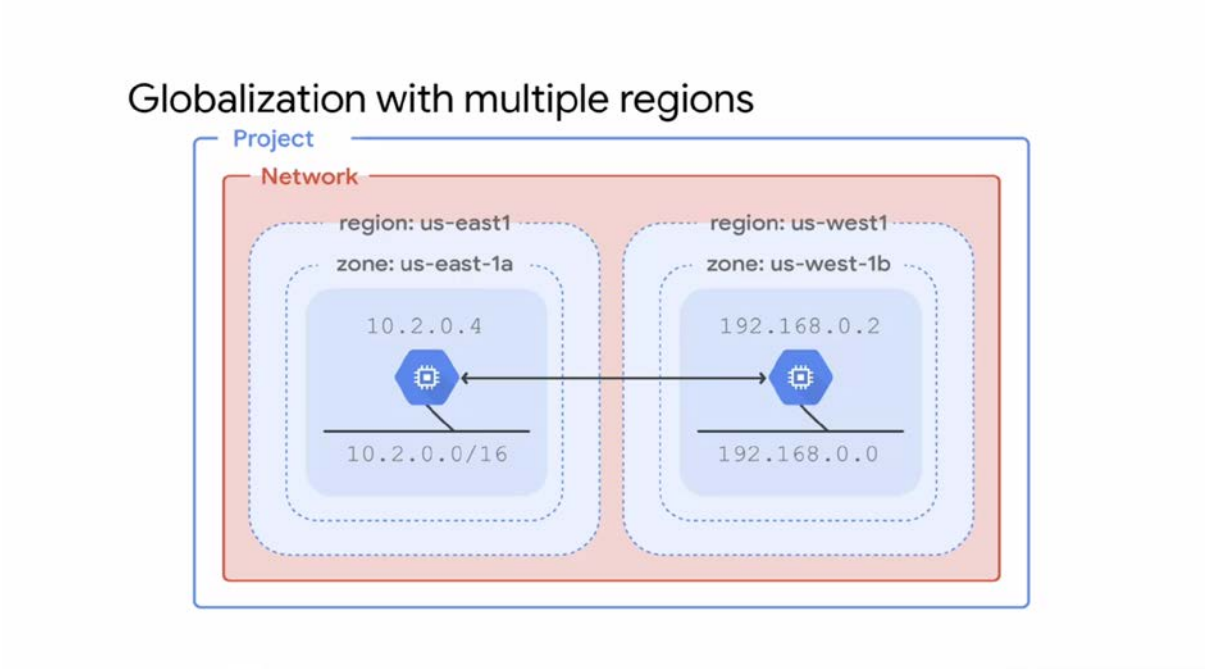
COMMON NETWORK DESIGNS



AVAILABILITY :

- If your application needs increased availability, you can place two virtual machines into multiple zones, but within the same subnet work as shown on this slide.
- Using a single sub-network allows you to to create a file a rule against the sub-network, in this case, 10.2.0.0/16. Therefore, by allocating VMs on a single subnet to separate zones,
- you get improved availability without additional security complexity.
- A regional managed instance group contains instances from multiple zones

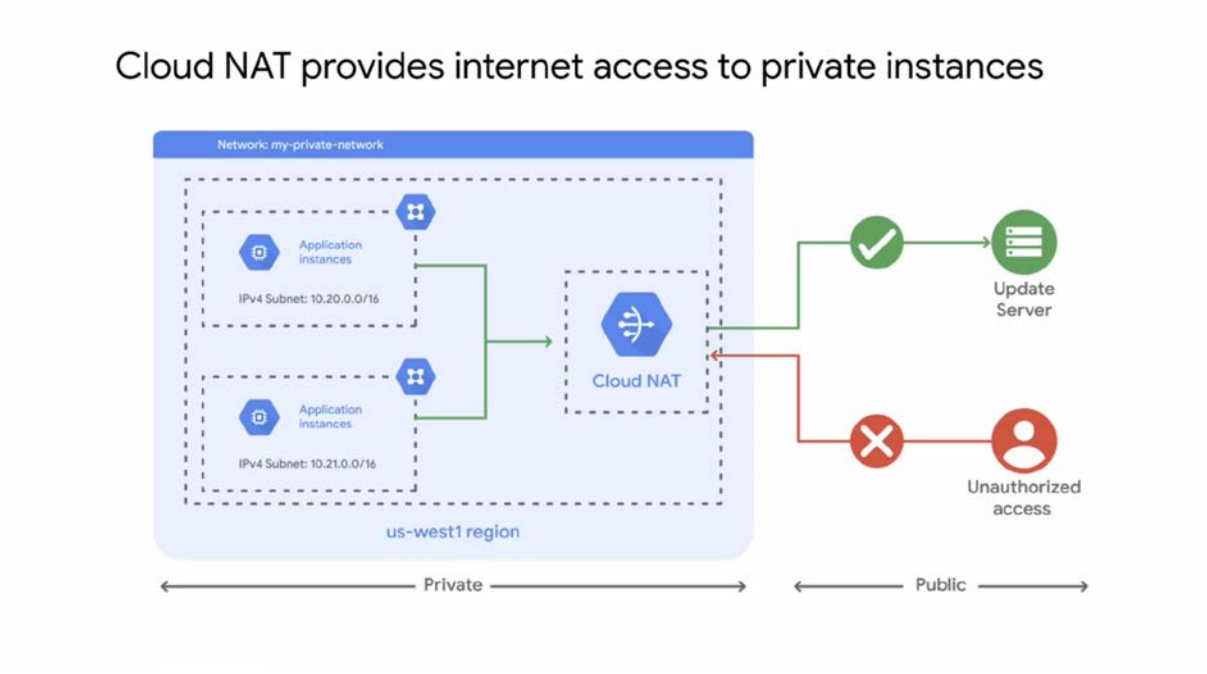
- across the same region, which provides increased availability



GLOBALIZATION :

- Putting resources in different regions provides an even higher degree of failure independence. This allows you to design robust systems with resources spread across different failure domains.

- When using a global load balancer like the HTTP load balancer, you can route traffic to the region that is closest to the user. This can result in better latency for users and lower network traffic costs for your project.



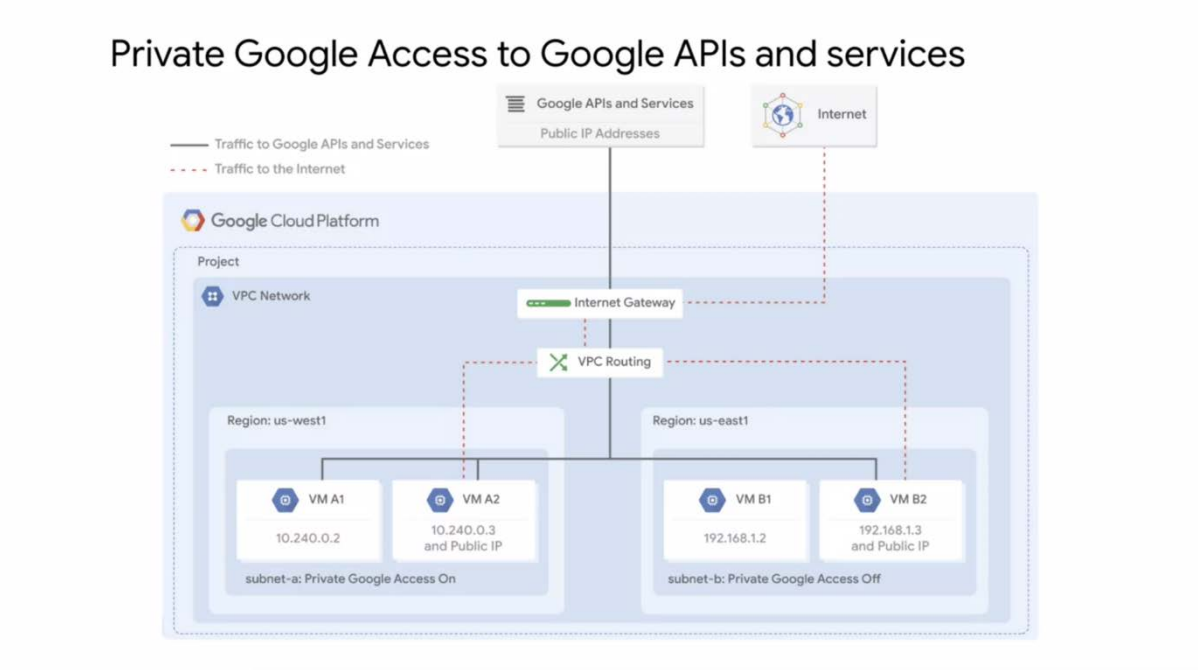
- Now, as a general security best practice, I recommend only assigning internal IP addresses to your VM instances whenever possible.

- **Cloud NAT** is Google's managed **network address translation** service. It lets you provision your application instances without public IP addresses, while also allowing them to access the internet in a controlled and efficient manner.

- This means your private instances can access the internet for updates, patching, configuration management, and more. In this diagram Cloud NAT enables two private instances to

- access an update server on the Internet, which is referred to as outbound NAT.

- However, Cloud NAT does not Implement inbound NAT. In other words, hosts outside your VPC network cannot directly access any of the private instances behind the cloud NAT gateway. This helps you keep your VPC networks isolated and secure.



- Similarly, you should enable private Google access to allow VM instances that only have internal IP addresses to reach the external IP addresses of Google APIs and services.

- For example, if your private VM instance needs to access a cloud storage bucket, you need to enable private Google access.

- You enable private Google access on a subnet by subnet basis. As you can see in this diagram, subnet A has private Google access enabled and subnet B has it disabled.

- This allows VMA one to access Google APIs and services, even though it has no external IP address.

- Private Google access has no effect on instances that have external IP addresses, that's why VMs A2 and B2 can access Google APIs and services. The only VM that can't access those APIs and services is VM B1. This VM has no public IP address and it is in a subnet where Google private access is disabled.

- **NOTE** : The Internet Control Message Protocol ICMP) is a supporting protocol in the Internet protocol suite. It is used by network devices, including

- **NOTE :** Classless Inter-Domain Routing (**CIDR**) is a method for allocating IP addresses and for IP routing.

20. COMPUTE ENGINE (IN-DEPTH)

GCP compute and processing options

	Compute Engine	Kubernetes Engine	App Engine Standard	App Engine Flexible	Cloud Functions
Language support	Any	Any	Python Node.js Go Java PHP	Python Node.js Go Java PHP Ruby .NET Custom Runtimes	Python Node.js Go
Usage model	IaaS	IaaS PaaS	PaaS	PaaS	Microservices Architecture
Scaling	Server Autoscaling	Cluster	Autoscaling managed servers		Serverless
Primary use case	General Workloads	Container Workloads	Scalable web applications Mobile backend applications		Lightweight Event Actions

- Compute Engine gives you the utmost inflexibility. Run whatever language you want, it's your virtual machine. This is purely an Infrastructure as a Service or IaaS model.
- You have a VM and an operating system and you can choose how to manage it and how to handle aspects such as autoscaling, where you'll configure the rules about adding more virtual machines in specific situations.
- The primary work case of Compute Engine is any general workload, especially an enterprise application that was designed to run on a server infrastructure.
- This makes Compute Engine very portable and easy to run in the Cloud.
- Other services like Google Kubernetes Engine, which consist of containers workloads may not be as easily transferable as what you're used to find On-premises.

Compute Engine

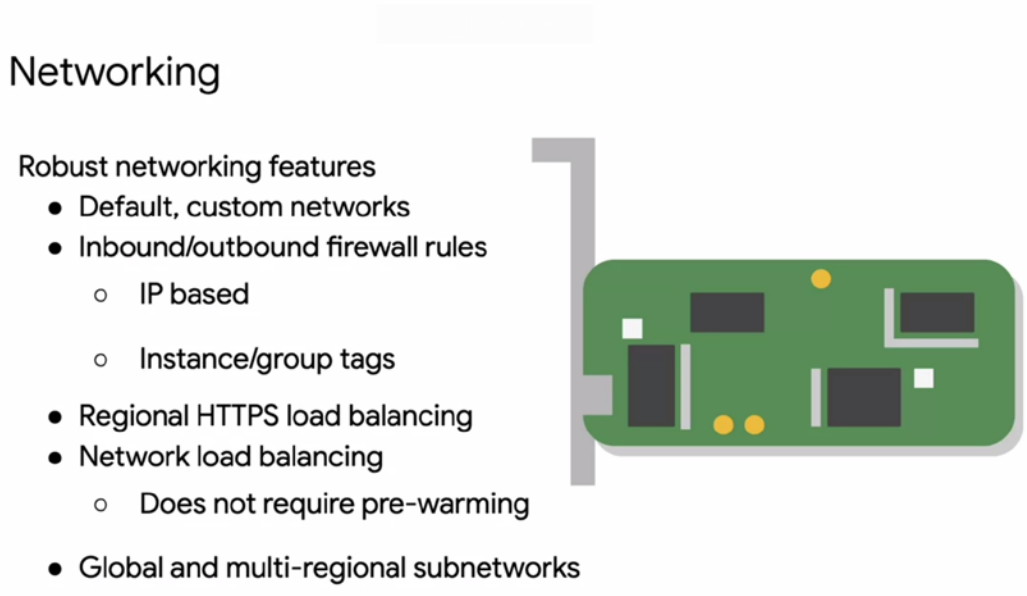
Infrastructure as a Service (IaaS)

Predefined or custom machine types:

- vCPUs (cores) and Memory (RAM)
- Persistent disks: HDD, SSD, and Local SSD
- Networking
- Linux or Windows

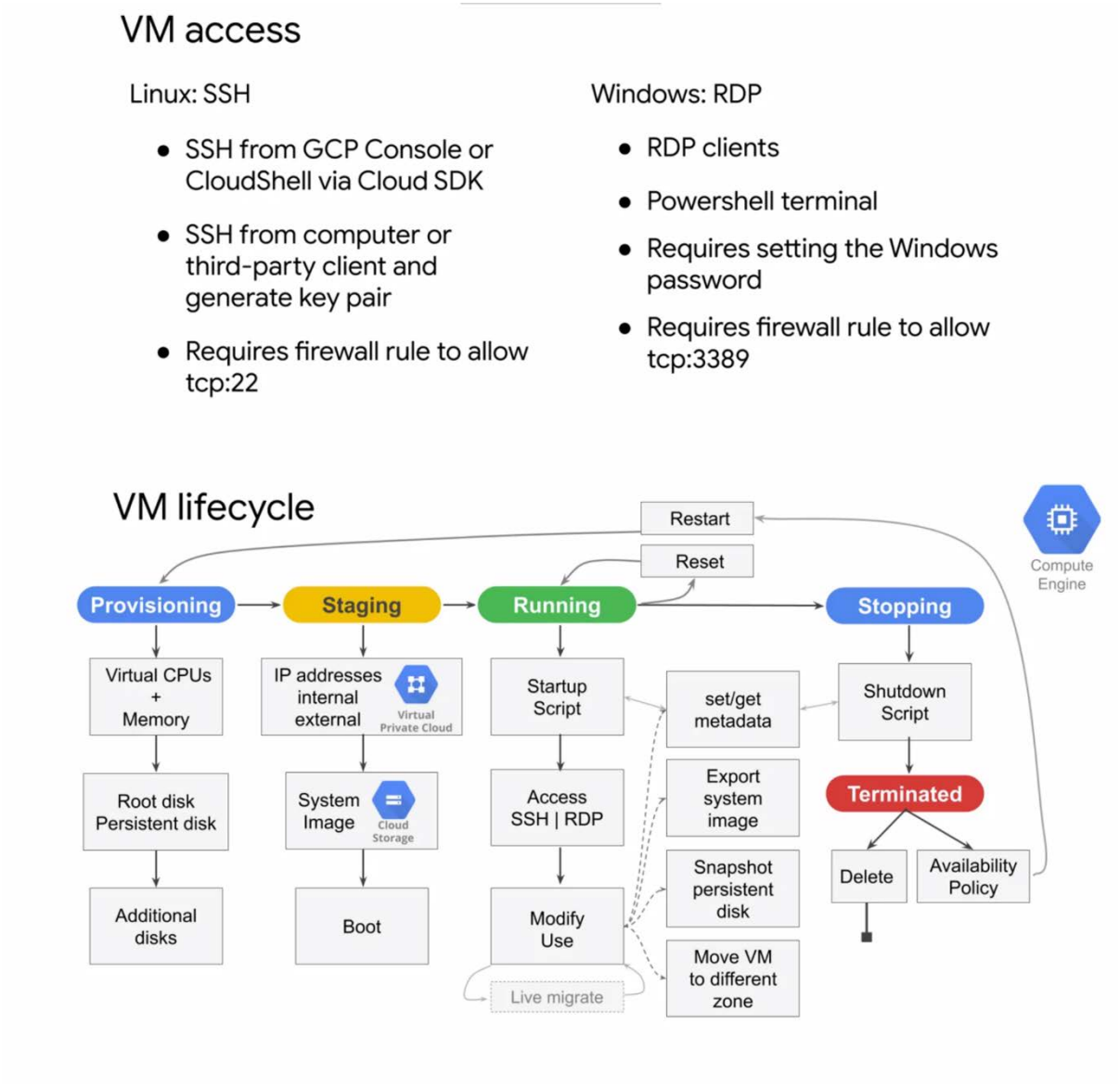
Compute Engine

- **Compute Engine** : it's physical servers that you're used to running inside the GCP environment with a number of different configurations. Both predefined and custom machine types allow you to choose how much memory and how much CPU you want.
- You chose the type of disk you want, what do you want to just use standard hard drives, SSDs, local SSDs, or a mix. You can even configure the networking interfaces and run a combination of Linux and Windows machines.
- Compute Engine provides several different machine types that we'll discuss later in this module.
- If those machines don't meet your needs, you can also customize your own machine. Your choice of CPU will affect your network throughput. Specifically, your network will scale at two gigabits per second for each CPU core, except for instances with two and four virtual CPUs which receive up to 10 gigabits per second of bandwidth. There's a theoretical maximum throughput of 32 gigabits per second for an instance with 16 or more CPUs, and 100 gigabits per second maximum throughput for specific instances that have T4 of V100 GPUs attached.
- When you're migrating from an on-premises setup, you're used to physical cores which have hyper-threading. On Compute Engine, each virtual CPU or vCPU is implemented as a single hardware hyper-thread on one of the available CPU platforms.
- After you pick your compute options, you want to choose your disk. You have three options, standard, SSD, or local SSD. So basically, do you want the standard spinning hard disk drives or HDDs, or flash memory solid state drives SSDs.
- Both of these options provide the same amount of capacity in terms of disk size when choosing a persistent disk. Therefore, the question really is about performance versus cost because there is a different pricing structure.
- Basically, SSDs are designed to give you a higher number of IOPS per dollar versus standard disks, which will give you a higher amount of capacity for your dollar. Local SSDs have even higher throughput and lower latency than SSD persistent disks because they're attached to the physical hardware.
- However, the data that you store on local SSDs persists only until you stop or delete the instance. Typically, a local SSD is used as a swap disk just like you would do if you want to create a RAM disc. But if you need more capacity, you can store those on a local SSD. You can create instances with up to eight separate 375 gigabytes local SSD partitions for total of
- three terabytes of local SSD space for each instance.
- Standard and non-local SSD disks can be sized up to 64 terabytes for each instance. The performance of these disks scales with each gigabyte of space allocated.



- There are different types of networks and created firewall rules using IP addresses and network tags. You'll also notice that you can do regional HTTPS load balancing and network load balancing.
- This doesn't require any pre-warming because a load balancer isn't a hardware device that needs to analyze your traffic. A load balancer is essentially a set of traffic engineering rules that are coming into the Google network. VPC is applying the rules destined for your IP address subnet range.

VM ACCESS AND LIFECYCLE



- The life cycle of a VM is represented by different statuses.
- When you define all the properties of an instance, and click "Create" the instance
- enters the **provisioning** state. Here the resources such as CPU, memory, and disk are being reserved for the instance but the instance itself isn't running yet.
- Next, the instant moves to the **staging** state where resources have been acquired and the instance is prepared for launch. Specifically in this state Compute Engine is adding IP addresses, booting up the system image, and booting up the system.

- After the instance starts **running**, it will go through pre-configured startup scripts and enable SSH or RDP access. Now, you can do several things while your instance is running.

- For example, you can live migrate your virtual machine to another host in the same zone

- instead of requiring your instance to be rebooted. This allows GCP to perform maintenance that is integral to keeping the infrastructure protected and reliable without interrupting any of your VMs. While you're instance is running, you can also

    - move your VM to a different zone.

    - take a snapshot of the VMs persistent disk,

    - export the system image or reconfigure metadata.

- Some actions require you to **stop** your virtual machine.

- For example, if you want to upgrade our machine by adding more CPU. When the instance enters this state, it will go through pre-configured shutdown scripts and end in the **terminated** state.

- From this state, you can choose to either restart instance which would bring it back to

- its provision state or delete it.

- You also have the option to reset a VM which is similar to pressing the reset button on your computer. This action wipes the memory content of the machine and resets the virtual machine to its initial state. The instance remains in the running state throughout the reset.

- There are different ways you can change a VM state from running. Some methods involve the GCP Console and the GCloud command while others are performed from the OS such as for a reboot and shut down.

- It's important to know that if you're restarting, rebooting, stopping, or even deleting an instance, the shutdown process will take about 90 seconds.

- For a preemptible VM, if the instance is not stopped after 30 seconds, Compute Engine sends an ACPI G3 mechanical off signal to the operating system.

- As I mentioned previously, Compute Engine can live migrate your virtual machine to another host due to a maintenance event to prevent your applications from experiencing disruptions. A VMs availability policy determines how they instance behaves in such an event.

- The default maintenance behavior for instances is to live migrate, but you can change the behavior to terminate your instance during maintenance events instead.

- If your VM is terminated due to a crash or other maintenance event, your instance automatically restarts by default but this can also be changed.

- These availability policies can be configured both during the instance creation and while an instance is running by configuring the automatic restart and on host maintenance options.

- When a VM is terminated, you do not pay for memory and CPU resources. However, you are charged for any attached disks and reserved IP addresses.

- In the terminated state, you can perform any of the actions listed here such as changing the machine type, but you cannot change the image of a stopped VM.

- Also, not all of the actions require you to stop a virtual machine. For example, VM availability policies can be changed while the VM is running as discussed previously.

Changing VM state from running

	methods	Shutdown Script time	state
`reset`	console, `gcloud`, API, OS	no	remains running
`restart`	console, `gcloud`, API, OS	no	terminated➡️running
`reboot`	OS: `sudo reboot`	~90 sec	running➡️running
`stop`	console, `gcloud`, API	~90 sec	running➡️terminated
`shutdown`	OS: `sudo shutdown`	~90 sec	running➡️terminated
`delete`	console, `gcloud`, API	~90 sec	running➡️N/A
preemption	*automatic*	~30 sec	N/A

"ACPI Power Off"

Availability policy: Automatic changes

Called "scheduling options" in SDK/API

Automatic restart

- Automatic VM restart due to crash or maintenance event
  - Not preemption or a user-initiated terminate

On host maintenance

- Determines whether host is live-migrated or terminated due to a maintenance event. Live migration is the default.

Live migration

- During maintenance event, VM is migrated to different hardware without interruption.
- Metadata indicates occurrence of live migration.

Stopped (Terminated) VM

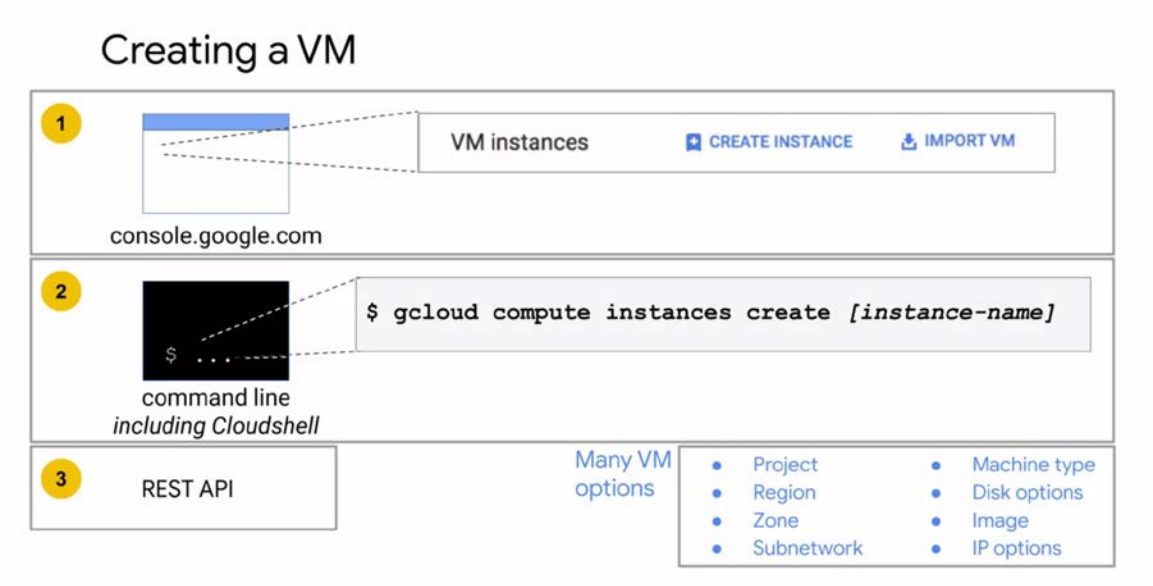
No charge for stopped VM

- Charged for attached disks and IPs

Actions

- Change the machine type.
- Add or removed attached disks; change auto-delete settings.
- Modify instance tags.
- Modify custom VM or project-wide metadata.
- Remove or set a new static IP.
- Modify VM availability policy.
- Can't change the image of a stopped VM.

VM COMPUTE MODELS



- You have three options for creating and configuring a VM.
    - GCP console
    - the Cloud Shell command-line,
    - or the RESTful API. If you'd like to automate andprocess very complex configurations, you might want to programmatically configure these through the RESTful API by defining all the different options for your environment.
- If you plan on using the command line or RESTful API, you should first configure the instance through the GCP Console, and then ask Compute Engine for the equivalent REST request or command-line. This way you avoid any typos and get drop-down lists of all the available CPU and memory options.
- A machine type specifies a particular collection of virtual hardware resources available to a VM instance, including the system memory size, vCPU count, and maximum persistent disk capability. GCP offers several machine types that can be grouped into two categories.

Machine types

Predefined machine types: Ratio of GB of memory per vCPU

- Standard
- High-memory
- High-CPU
- Memory-optimized
- Compute-optimized
- Shared-core

Custom machine types:

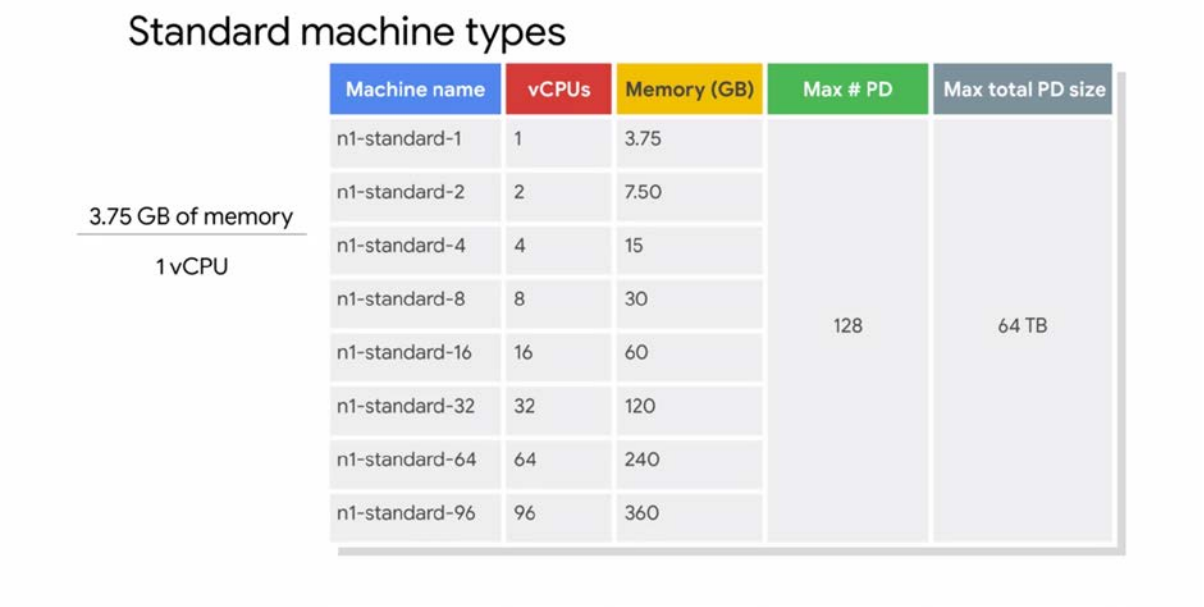
- You specify the amount of memory and number of vCPUs.

- **Predefined machine types -** These have fixed collection of resources, are managed by Compute Engine, and are available in multiple different classes. Each class has a predefined ratio of gigabytes of memory per Virtual CPU. These are the standard machine types, high memory, high CPU, memory optimized, Compute optimize, and shared-core machine types.

- There also the custom machine types. These lets you specify the number of virtual CPUs, and the amount of memory for your instance.

- **Standard machine types** are suitable for tasks that have a balance of CPU and memory needs. Standard machine types have 3.75 gigabytes of memory per virtual CPU.

- The virtual CPU configurations come in different intervals from 1vCPU all the way to 96 vCPUs as shown on this table. Each of these machines supports a maximum of 128 persistent disks with a total persistent disk size of 64 terabytes, which is also the case for the high memory, high CPU, memory optimized, and compute optimized machine types.



- **High memory machine types** are ideal for tasks that require more memory relative to vCPUs. High memory machine types have 6.5 gigabytes of system memory per vCPU.

- Similar to the std machine types, the vCPU configurations come in different intervals from 2vCPUs all the way to 96vCPUs,

High-memory machine types

6.5 GB of memory

1 vCPU

Machine name	vCPUs	Memory (GB)	Max # PD	Max total PD size
n1-highmem-2	2	13	128	64 TB
n1-highmem-4	4	26		
n1-highmem-8	8	52		
n1-highmem-16	16	104		
n1-highmem-32	32	208		
n1-highmem-64	64	416		
n1-highmem-96	96	624		

- **High CPU machine types** are ideal for tasks that require more vCPUs relative to memory. High CPU machine types have 0.9 gigabytes of memory per vCPU.

High-CPU machine types

0.9 GB of memory

1 vCPU

Machine name	vCPUs	Memory (GB)	Max # PD	Max total PD size
n1-highcpu-2	2	1.80	128	64 TB
n1-highcpu-4	4	3.60		
n1-highcpu-8	8	7.20		
n1-highcpu-16	16	14.4		
n1-highcpu-32	32	28.8		
n1-highcpu-64	64	57.6		
n1-highcpu-96	96	86.4		

- **Memory optimized machine types** are ideal for tasks that require intensive use of memory with higher memory to vCPU ratios than high memory machine types. These machine types are perfectly suited for in-memory databases and in-memory analytics such as SAP HANA (High Performance Analytics Application) and business warehouse workloads, genomic analysis, and SQL Analysis Services. Memory optimized machine types have more than 14 gigabytes of memory per vCPU. These Machines come in four configurations

Memory-optimized machine types

>14 GB of memory

1 vCPU

Machine name	vCPUs	Memory (GB)	Max # PD	Max total PD size
n1-ultramem-40	40	961	128	64 TB
n1-ultramem-80	80	1922		
n1-megamem-96	96	1433.6		
n1-ultramem-160	160	3844		

- **Compute optimized machine types** are ideal for compute intensive workloads. These machine types are for the highest performance per core on Compute Engine. Built on the latest generation Intel scalable processors, the casket lake, C2 machine types offer up to 3.8 gigahertz sustained all-core turbo, and provide full transparency into the architecture of the underlying server platforms, enabling advanced performance tuning.

- C2 machine types offer much more computing power, run on a newer platform, and are generally more robust for compute intensive workloads than the n1 high CPU machine types.

Compute-optimized machine types

Highest performance per vCPU (3.8Ghz sustained all-core turbo)

Machine name	vCPUs	Memory (GB)	Max # PD	Max total PD size
c2-standard-4	4	16	128	64 TB
c2-standard-8	8	32		
c2-standard-16	16	64		
c2-standard-30	30	120		
c2-standard-60	60	240		

- **Shared-core machine types** provide one virtual CPU that is allowed to run for a portion of the time on a single hardware hyper-thread on the host CPU running your instance.

Shared-core machine types

Machine name	vCPUs	Memory (GB)	Max # PD	Max total PD size
f1-micro	0.2	0.60	16	3 TB
g1-small	0.5	1.70		

- Shared-core instances can be more cost effective for running small non resource intensive applications than other machine types.
- There are only two shared-core machine types to choose from, they're the f1-micro and the g1-small. The f1-micro machine types offer bursting capabilities that allow instances to use additional physical CPU for short periods of time.
- Bursting happens automatically when your instance requires more physical CPU than you originally allocated. During these spikes, your instance will opportunistically take advantage of available physical CPU in bursts. Note that bursts are not permanent and are only possible periodically.
- If none of the predefined machine types match your needs, you can independently specify the number of vCPUs and the amount of memory for your instance.
- **Custom machine types** are ideal for the following scenarios;
    - when you have workloads that are not a good fit for the predefined machine types that are available to you,
    - when you have workloads that require more processing power or more memory but you don't need all of the upgrades that are provided by the next larger predefined machine type.

Creating custom machine types

When to select custom:

- Requirements fit between the predefined types
- Need more memory or more CPU

Customize the amount of memory and vCPU for your machine:

- Either 1 vCPU or even number of vCPU
- 0.9 GB per vCPU, up to 6.5 GB per vCPU (default)
- Total memory must be multiple of 256 MB

- It cost slightly more to use a custom machine type than equivalent predefined machine type.

- There are still some limitations in the amount of memory and vCPUs you can select. Only machine types with one virtual CPU or an even number of virtual CPUs can be created. Memory must be between 0.9 gigabytes and 6.5 gigabytes per virtual CPU by default. The total memory of the instance must be a multiple of 256 megabytes.

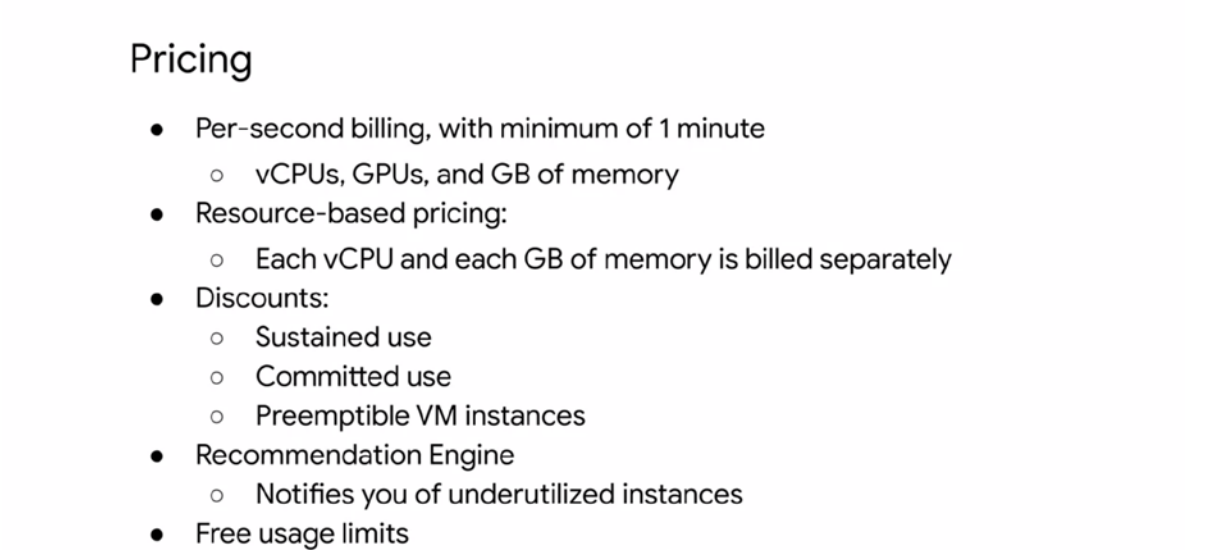
- By default, a custom machine can have up to 6.5 gigabytes of memory per virtual CPU. However, this might not be enough memory for your workload. So at an additional cost, you can get more memory per virtual CPU beyond the 6.5 gigabytes limit. This is referred to as **extended memory**,

- The first thing you want to consider when choosing a region and zone is the geographical location in which you want to run your resources.

- This map shows the current and planned GCP regions and the number of zones. Each zone supports a combination of different microprocessors - Ivy Bridge, Sandy Bridge, Haswell, Broadwell, and Skylake platforms. When you create an instance in the zone, your instance will use the default processes supported in that zone. For example, if you create an instance in the US-central1 a zone, your instance will use the Sandy Bridge processor.

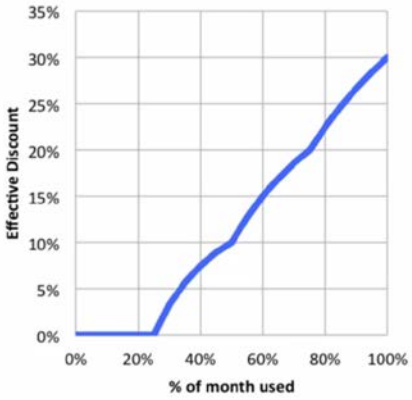
VM PRICING

Compute Engine uses a resource-based pricing model where each virtual CPU and each gigabyte of memory on Compute Engine is built separately rather than as part of a single machine type. You still create instances using predefined machine types, but your bill reports them as individual vCPUs and memory used.



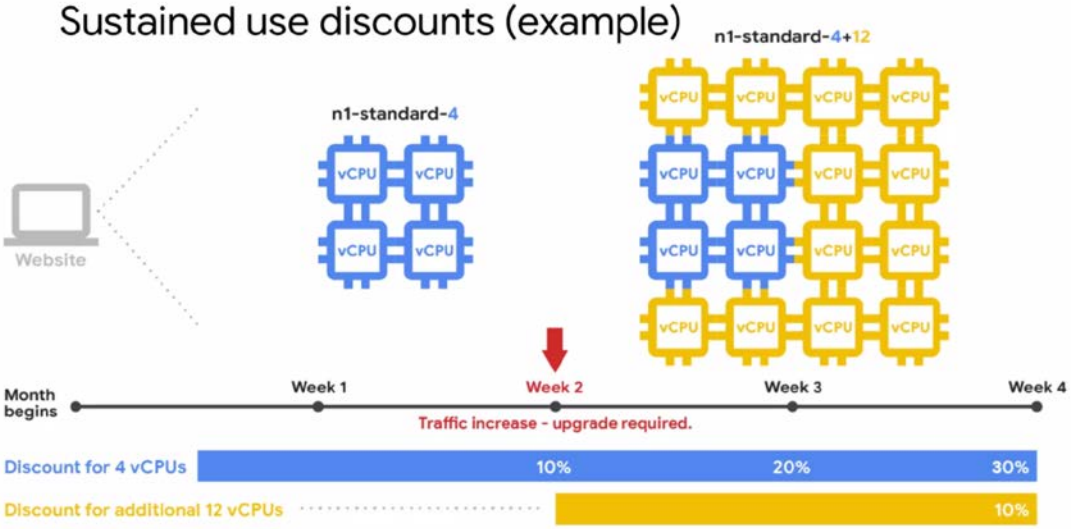
Sustained use discounts

Usage Level (% of month)	% at which incremental is charged
0% - 25%	100% of base rate
25% - 50%	80% of base rate
50% - 75%	60% of base rate
75% - 100%	40% of base rate



Up to 30% net discount for instances that run the entire month

Sustained use discounts (example)

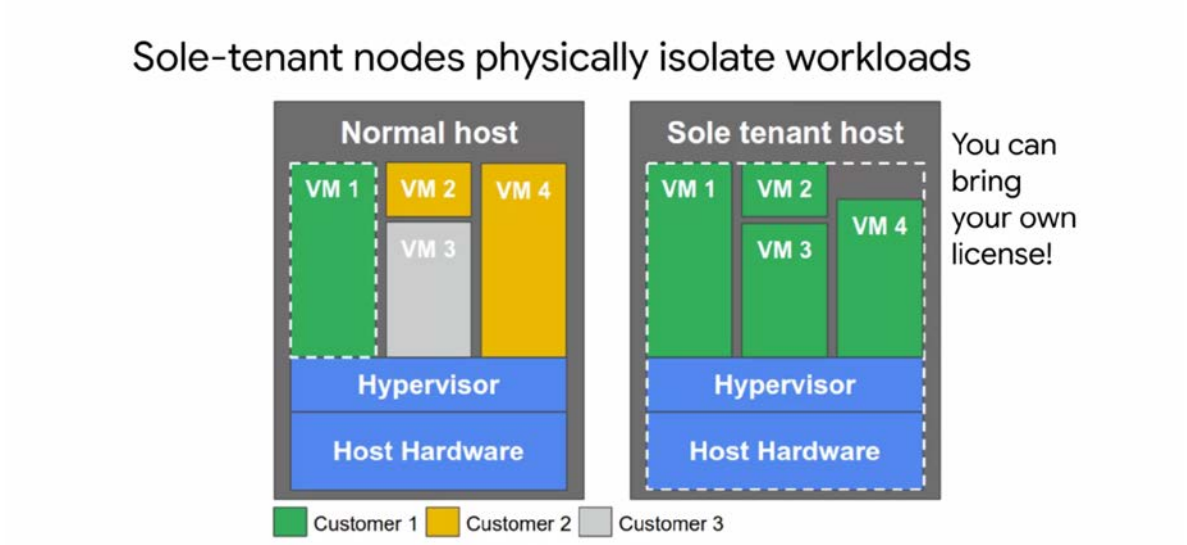


OTHER VM CONFIGURATIONS

Preemptible

- Lower price for interruptible service (up to 80%)
- VM might be terminated at any time
  - No charge if terminated in the first minute
  - 24 hours max
  - 30-second terminate warning, but not guaranteed
    - *Time for a shutdown script*
- No live migrate; no auto restart
- You *can* request that CPU quota for a region be split between regular and preemption
  - Default: preemptible VMs count against region CPU quota

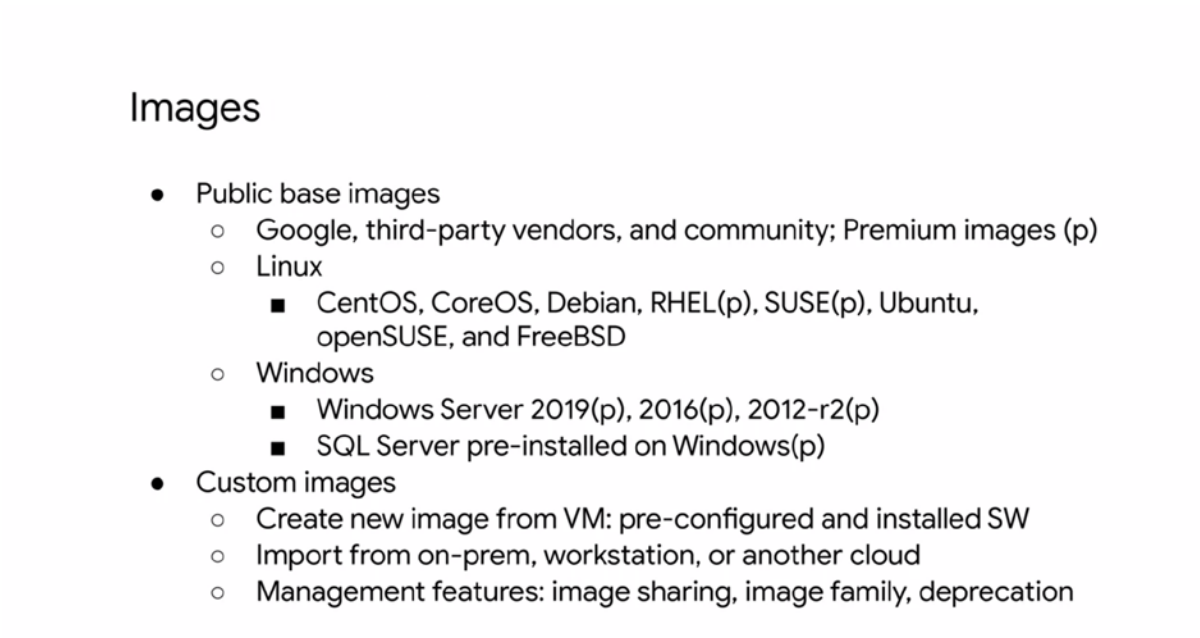
One major use case of preemptible VMs is running a batch processing job. If some of those instances terminate during processing, the job slows down but does not completely stop. Therefore preemptible instances complete your batch processing tasks without placing additional workload on your existing instances and without requiring you to pay full price for additional normal instances.



If you have workloads that require physical isolation from other workloads, or virtual machines in order to meet compliance requirements, you want to consider sole-tenant nodes.
A sole-tenant node is a physical Compute Engine server that is dedicated to hosting VM instances only for your specific project. Use sole-tenant nodes to keep your instances physically separated from instances in other projects, or to group your instances together in the same host hardware.
For example if you have a payment processing workload that needs to be isolated to meet the compliance requirements. The diagram on the left shows a normal host with multiple VM instances from multiple customers. A sole-tenant node as shown on the right, also has multiple VM instances, but they all belong to the same project.
The only available node type can accommodate VM instances up to 96 V CPUs and 624 gigabytes of memory. You can also fill the node with multiple smaller VM instances of various sizes including custom machine types and instances with extended memory.

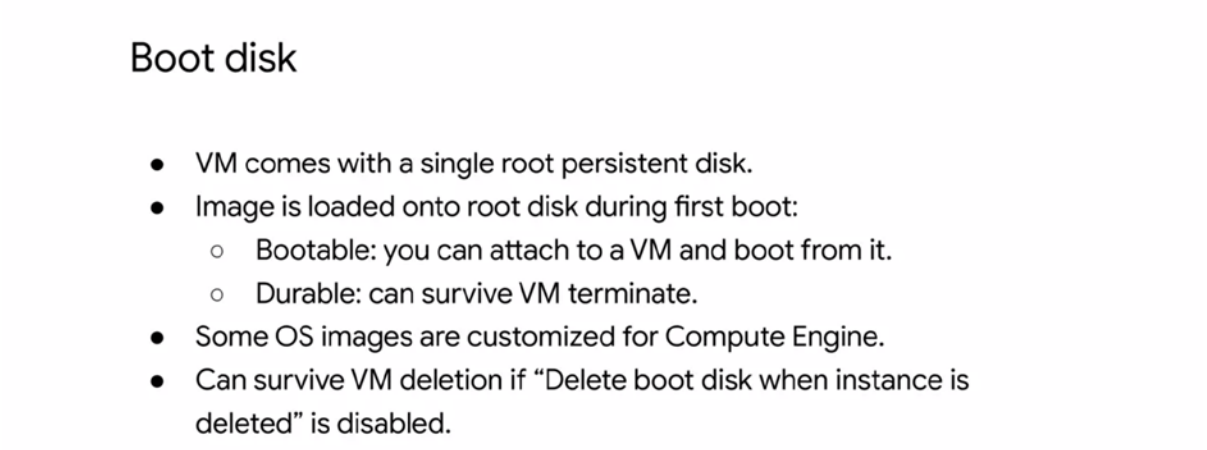
Also if you have existing operating system licenses, you can bring them to Compute Engine using sole-tenant nodes while minimizing Physical Core usage with the in-place restart feature.
Another Compute option is to create shielded VMs**. Shielded VMs** offer verifiable integrity of your VM instances.
So you can be confident that you're instances haven't been compromised by boot or kernel level of malware or rootkits.
Shielded VMs verifiable integrity is achieved through the use of secure boot, Virtual Trusted Platform Module or VTPM enabled measured boot and integrity monitoring. Shield VMs is the first offering in the **shielded Cloud initiative.**
The shielded Cloud initiative is meant to provide an even more secure foundation for all of GCP by providing verifiable integrity, and offering features like VTPM shielding or ceiling that help prevent data exfiltration.
In order to use the shielded VM features, you need to select a shielded image.

IMAGES



- This image includes the
  - boot loader,
  - the operating system,
  - the file system structure,
  - any pre-configured software,
  - any other customizations
- you can choose from both Linux and Windows images. Some of these images are premium images as indicated in parentheses with a p. These images will have per second charges after a one-minute minimum, with the exception of SQL Server images, which are charged per minute after a 10-minute minimum.
- Premium image prices vary with the machine type. However, these prices are global and do not vary by region or zone. You can also use custom images.
- For example, you can create and use a custom image by pre installing software that's been authorized for your particular organization. You also have the option of importing images from your on-premises or workstation, or from another cloud provider.
- This is a no cost service that is as simple as installing an agent. You can also share custom images with anybody in your project or among other projects too.

DISKS

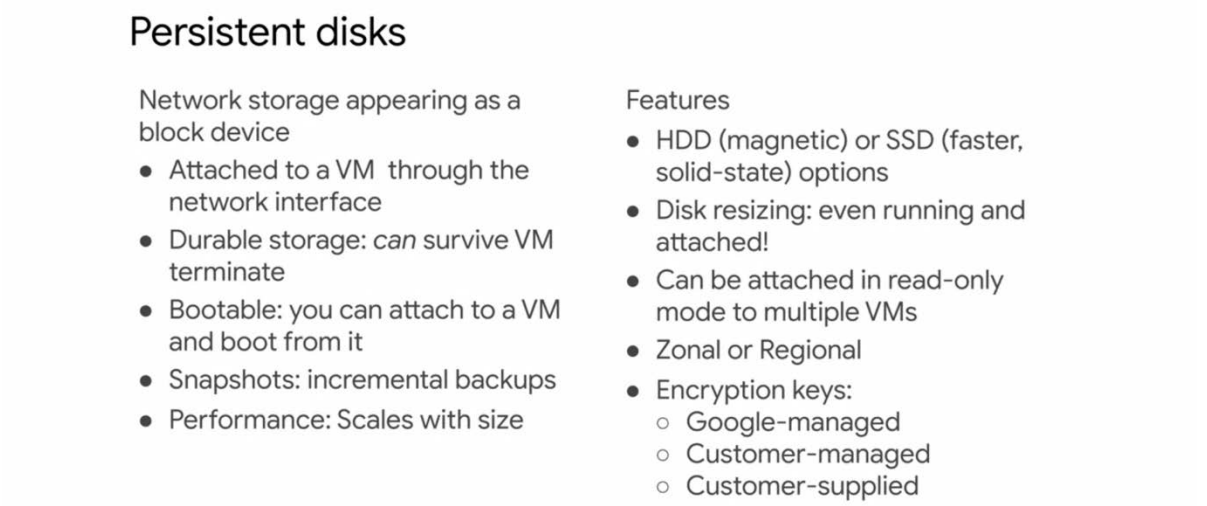


- The operating system is going to be included as part of some kind of disk. Every single VM comes with a single root persistent disk because you're choosing a base image to have that loaded on.

- This image is bootable and that you can attach it to VM and boot from it. It's durable and that it can survive, if the VM terminates. To have a boot disks survive a VM deletion, you need to disable the delete boot disk when instance is deleted option in the instances properties.

- There are different types of disks.

- The first is that we create, is what we call a **persistent disk**. That means it's going to be attached to the VM through the network interface. Even though it's persistent, it's not physically attached to the machine.



- The separation of disk and compute, allows a disk to survive if the VM terminates. You can also perform snapshots of these disks which are incremental backups that we'll discuss later.

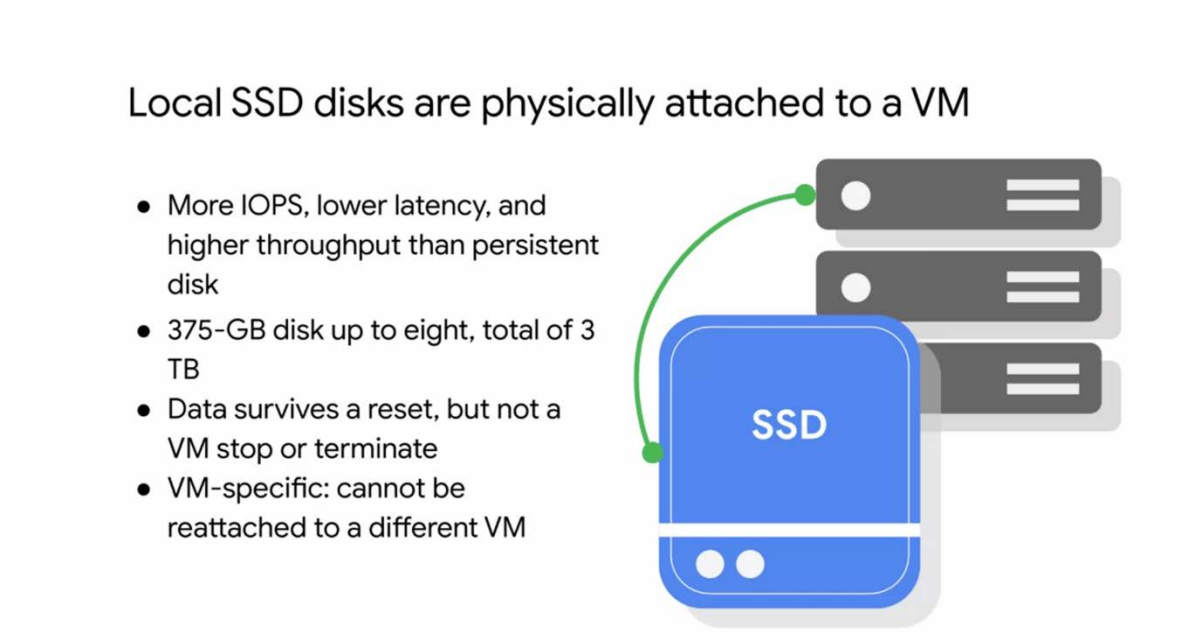
- The choice between HDD and SSD disk comes down to cost and performance. To learn more about the disk performance and how it scales with disk size, see the links section of this video.

- Another cool feature of persistent disks, is that you can dynamically resize them, even while they are running and attached to a VM.

- You can also attach a disk in read only mode to multiple VMs. This allows you to share static data between multiple instances, which is cheaper than replicating your data to unique disks for individual instances.

- By default, computer engine encrypts all data at rest. GCP handles and manages this encryption for you, without any additional actions on your part. However, if you wanted to control and manage this encryption yourself, you can either use Cloud key management service, to create and manage key encryption keys, which is known as customer managed encryption keys. Or you can create and manage your own key encryption keys known as customer supplied encryption keys.

- Now, **local SSDs** are different from persistent disks and that they're physically attached to the virtual machine. Therefore, these disk are ephemeral, but provide very high IOPS.



- Currently, you can attach up to eight local SSD disks with 375 gigabytes each, resulting in a total of three terabytes. Data on these disks will survive a reset, but not a VM stop or terminate, because these disks can't be reattached to a different VM.

- There is also the option of using a **RAM disk**. You can simply use ‘tmpfs’ if you want to store data in memory. This will be the fastest type of performance available if you need small data structures. A high memory virtual machine is recommended to take advantage of such features, along with a persistent disk to backup the RAM disk data. This limit depends on the machine type.

RAM disk

- `tmpfs`
- Faster than local disk, slower than memory
  - Use when your application expects a file system structure and cannot directly store its data in memory
  - Fast scratch disk, or fast cache
- Very volatile; erase on stop or restart
- May need a larger machine type if RAM was sized for the application
- Consider using a persistent disk to back up RAM disk data

- For the shared core machine type, you can attach up to 16 disks. For the standard high CPU memory optimized and compute-optimized machine types, you can attach up to 128 disks.

Maximum persistent disks

Machine Type	Disk number limit
Shared-core	16
Standard	128
High-memory	
High-CPU	
Memory-optimized	
Compute-optimized	

- Throughput is limited by the number of cores that you have. That throughput also shares the same bandwidth with disk IO. So if you plan on having a large amount of disk IO throughput, you will also compete with any network egress or ingress throughput.

- So remember that, especially if you'll be increasing the number of drives attached to a virtual machine.

Summary of disk options

	Persistent disk HDD	Persistent disk SSD	Local SSD disk	RAM disk
Data redundancy	Yes	Yes	No	No
Encryption at rest	Yes	Yes	Yes	N/A
Snapshotting	Yes	Yes	No	No
Bootable	Yes	Yes	No	Not
Use case	General, bulk file storage	Very random IOPS	High IOPS and low latency	low latency and risk of data loss

Persistent disk management differences

Cloud Persistent Disk

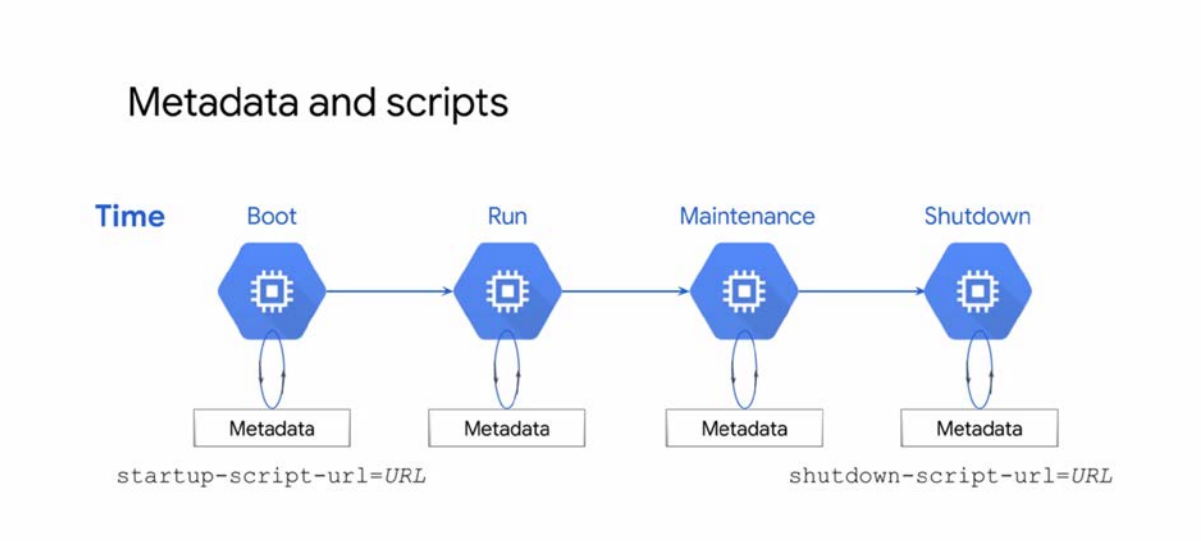
- Single file system is best
- Resize (grow) disks
- Resize file system
- Built-in snapshot service
- Automatic encryption

Computer Hardware Disk

- Partitioning
- Repartition disk
- Reformat
- Redundant disk arrays
- Subvolume management and snapshots
- Encrypt files before write to disk

COMMON COMPUTE ENGINE ACTIONS (METADATA, MOVING,SNAPSHOTTIN, RESIZING)

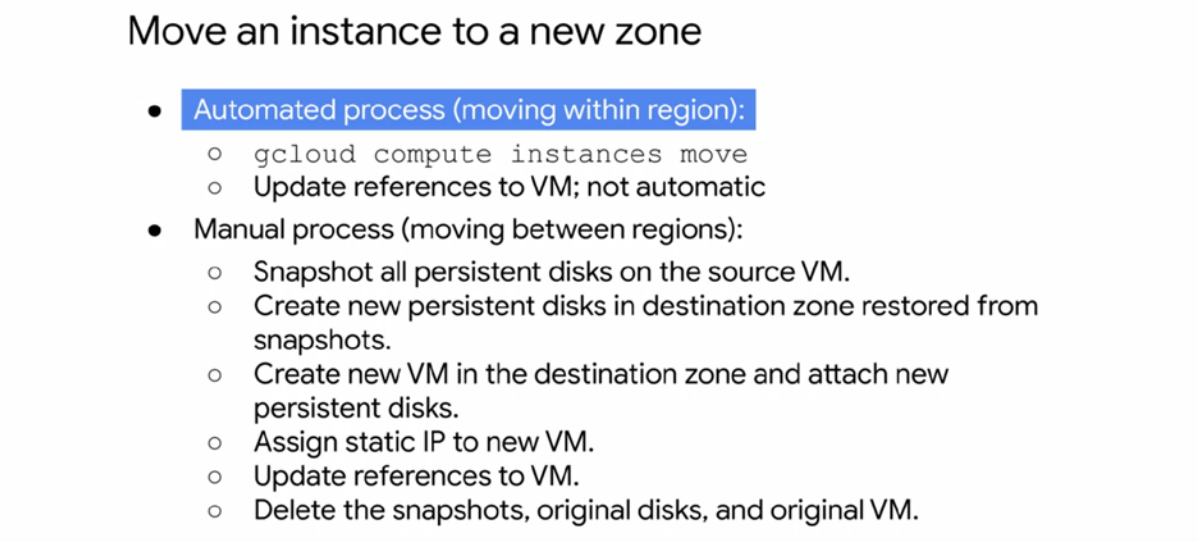
- Every VM instance **stores its metadata** on a metadata server. The metadata server is particularly useful in combination with startup and shutdown scripts because you can use the metadata server to programmatically get unique information about an instance without additional authorization.



- For example, you can write a startup script that gets the metadata key value pair for an instance's external IP address and use that IP address new script to setup a database. Because the default metadata keys are the same on every instance, you can reuse your script without having to update it for each instance, this helps you create less brittle code for your applications.

- Storing and retrieving instance metadata is a very common Compute Engine action.

- **Different Zone (same region)** - Another common action is to move an instance to a new zone. For example, you might do so for geographical reasons or because a zone is being deprecated. If you move your instance within the same region, you can **automate** the move by using the ***gcloud compute instances move*** command.



- **Different Region** - If we move your instance to a different region, you need to manually do so

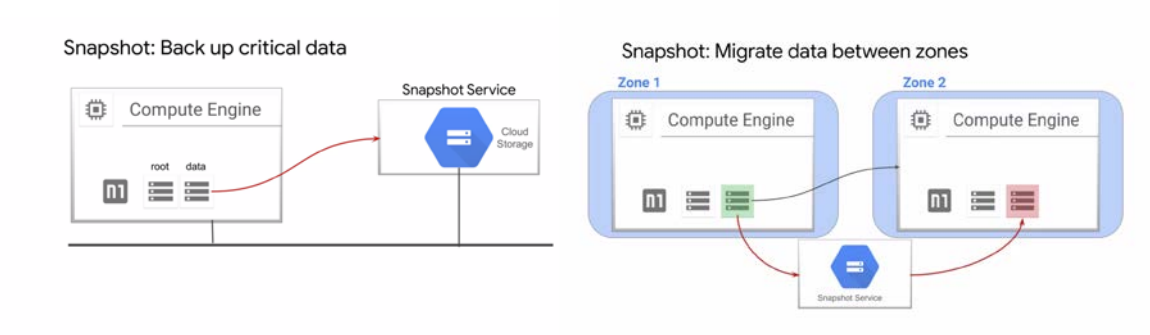
- **Snapshots** have many use cases.

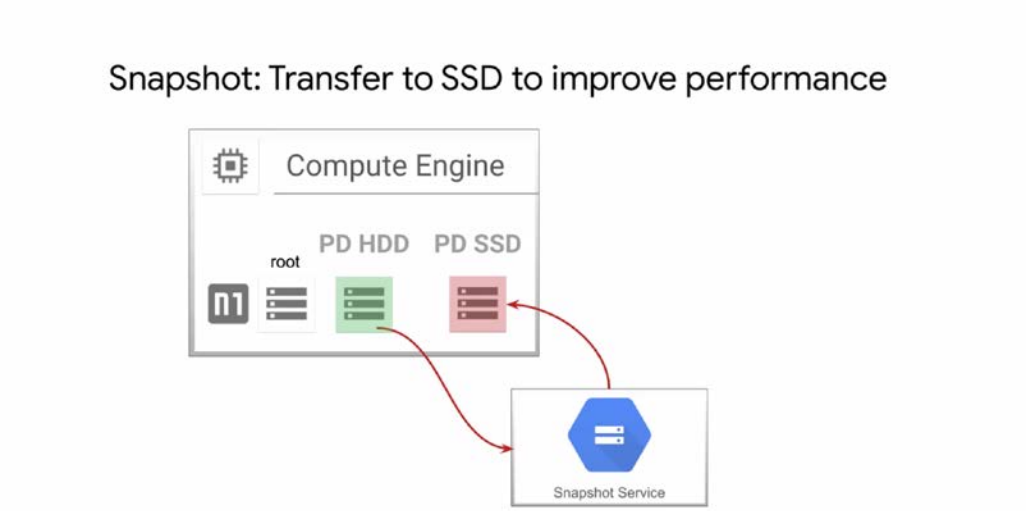
- For example, they can be used to backup critical data into a durable storage solution to meet application, availability, and recovery requirements. These snapshots are stored in Cloud Storage.

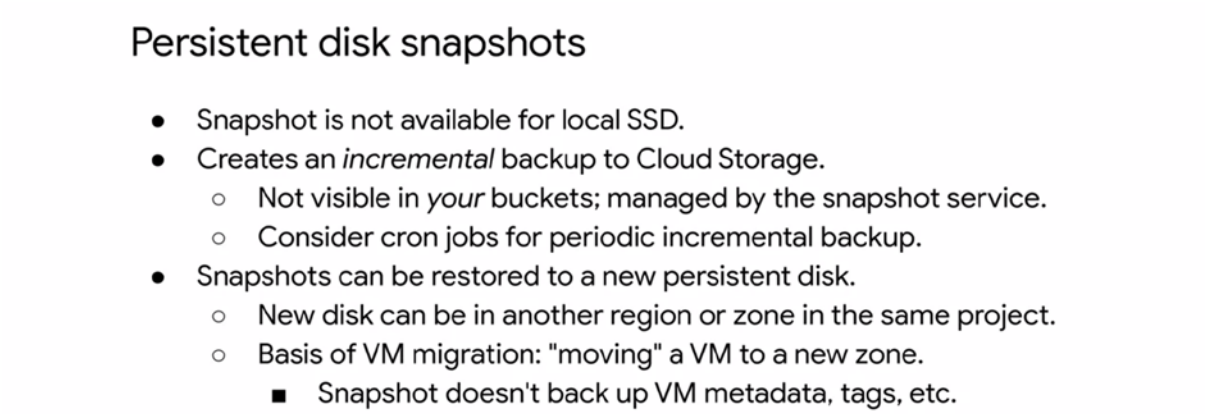
- Snapshots can also be used to micro data between zones. the manual process of moving an instance between two regions, but this can also be used to simply transfer data from one zone to another. For example, you might want to minimize latency by migrating data to a drive that can be locally attached in the zone where it is used.

- Which brings me to another snapshot use case of transferring data to a different disk type. For example, if you want to improve disk performance, you could use a snapshot to transfer data from a standard ECD persistent disk to a SSD persistent disk. Now that I've covered some of these snapshot use cases, let's explore the concept of a disk snapshot.

- **Snapshots** are available only to persistent disks and not to local SSDs. Snapshots are different from public images and custom images which are used primarily to create instances or configure instance templates, in that snapshots are useful for periodic backup of the data on your persistent disks.







- Snapshots are incremental and automatically compressed, so you can create regular snapshots on a persistent disk faster and at a much lower cost than if you regularly created a full image of the disk.

- As we saw with the previous examples, snapshots can be restored to a new persistent disk, allowing for a move to a new zone.

- Another common Compute Engine action is to **resize your persistent disk.** The added benefit of increasing storage capacity is to improve I/O performance. This can be achieved while the disk is attached to a running VM without having to create a snapshot. Now, while you can grow disk and size, you can never shrink them.

21. CLOUD IAM

- It is a way of identifying who can do what, on which resource.

- The who can be a person, group, or application. The what refers to specific privileges or actions,

- and the resource could be any GCP service. For example I could give you the privilege or role of compute viewer. This provides you with read-only access to get enlist Compute Engine resources without being able to read the data stored on them.

- Cloud IAM is composed of different objects

- Google Cloud Platform resources are organized hierarchically as shown in this tree structure. The organization node is the root node in this hierarchy. Folders are the children of the organization. Projects are the children of the folders, and the individual resources are the children of projects. Each resource has exactly one parent.

- Cloud IAM allows you to set policies at all of these levels. Where a policy contains a set of roles, and role members.

- Resources inherit policies from their parent.

- The organization resource represents your company. Cloud IAM roles granted at this level are inherited by all resources under the organization. The folder resource could represent your department. Cloud IAM roles granted at this level are inherited by all resources that the folder contains. Projects represent a trust boundary within your company. Services within the same project have a default level of trust. The Cloud IAM

policy hierarchy always follows the same path as the GCP resource hierarchy, which means that if you change the resource hierarchy the policy hierarchy also changes.

- For example, moving a project into a different organization will update the project's Cloud IAM policy to inherit from the new organizations Cloud IAM policy.
- Also, child policies cannot restrict access granted at the parent level.
- For example, if I grant you that editor role for department X, and I grant you the viewer role at the Bookshelf project level, you still have the editor role for that project.
- Principle of least privilege : The principle applies to identities, roles, and resources. Always select the smallest scope that's necessary for the task in order to reduce your exposure to risk.

ORGANIZATION

- The organization resource is the root node in the GCP resource hierarchy.
- This node has many roles like the organization admin. The organization admin provides a user like Bob with access to administer all resources belonging to his organization, which is useful for auditing.
- There is also a project creator role, which allows a user like Alice to create projects within her organization.
- Project creator role here because it can also be applied at the organization level which would then be inherited by all the projects within the organization.
- The organization resource is closely associated with a G Suite or Cloud Identity account.
- When a user with a G Suite or Cloud Identity account creates a GCP project, an organization resource is automatically provisioned for them.
- Then Google Cloud communicates its availability to the G Suite or Cloud Identity super admins.
- These super admin accounts should be used very carefully because they have a lot of control over your organization and all the resources underneath it.
- The G Suite or Cloud Identity super administrators and the GCP organization admin are
- key roles during the setup process and for lifecycle control for the organization resource.
- The two roles are generally assigned to different users or groups, although this depends on the
- organization's structure and needs. In the context of GCP organization setup,
- the G Suite or Cloud Identity super administrator responsibilities are;
    - assign the organization admin role to some users,
    - be a point of contact in case of recovery issues,

    - control the lifecycle of the G Suite or Cloud Identity account and organization resource.

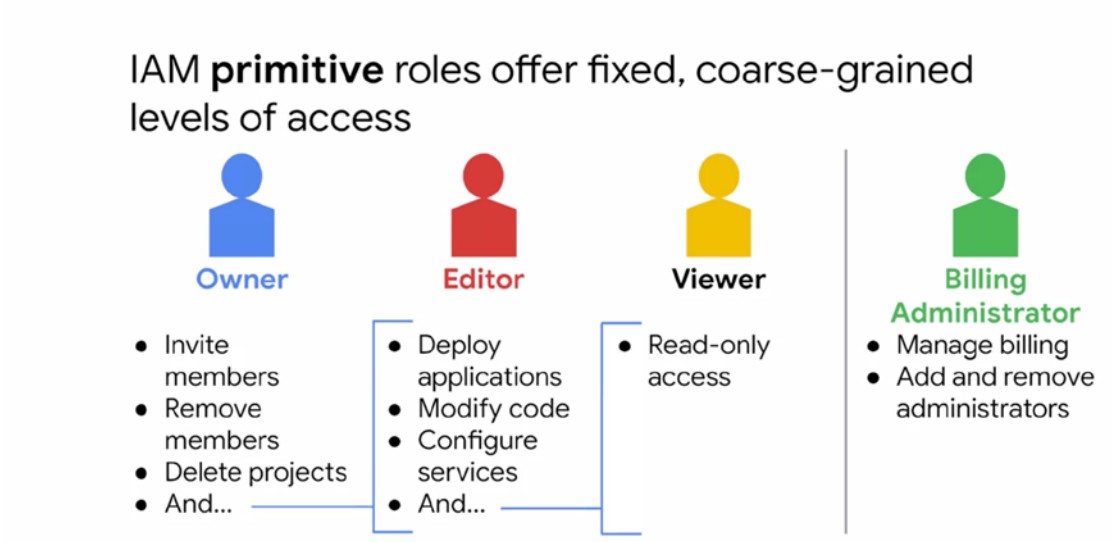
- The responsibilities of the organization admin role
    - are; define IAM policies,
    - determined the structure of the resource hierarchy,
    - delegate responsibility over critical components such as networking,
    - billing, and resource hierarchy through IAM roles.
- Following the principle of least privilege, this role does not include the permission to perform
- other actions such as creating folders. To get these permissions, an organization admin must assign additional roles to their account for.
- Let's talk more about folders because they can be viewed as sub organizations within the organization. Folders provide an additional grouping mechanism and isolation boundary between projects.
- Folders can be used to model different legal entities, departments, and teams within a company.
- Folders allow delegation of administration rights. So for example, each head of a department can be granted full ownership of all GCP resources that belong to their department.
- Similarly, access to resources can be limited by folder. So users in one department can only access and create GCP resources within that folder.

- The folder node has multiple roles that mimic the organizational roles but are applied to resources within a folder.

- **Admin** role that provides full control over folders, a creator role to browse the hierarchy and create folders, and a viewer role to view folders and projects below a resource.
- **Creator** role that allows a user to create new projects making that user automatically the owner.
- **Deleter** role that grants deletion privileges for projects.

ROLES

There are three types of roles in Cloud IAM;
Primitive roles, predefined roles, and custom roles.
Primitive roles are the original roles that were available in the GCP console, but they are broad. You apply them to a GCP project and they affect all resources in that project.
In other words, IAM primitive roles are for fixed, coarse-grained levels of access.
The permanent roles are the owner, editor, and viewer roles. There is also a billing administrator role to manage billing an add or remove administrators without the right to change the resources in the project.



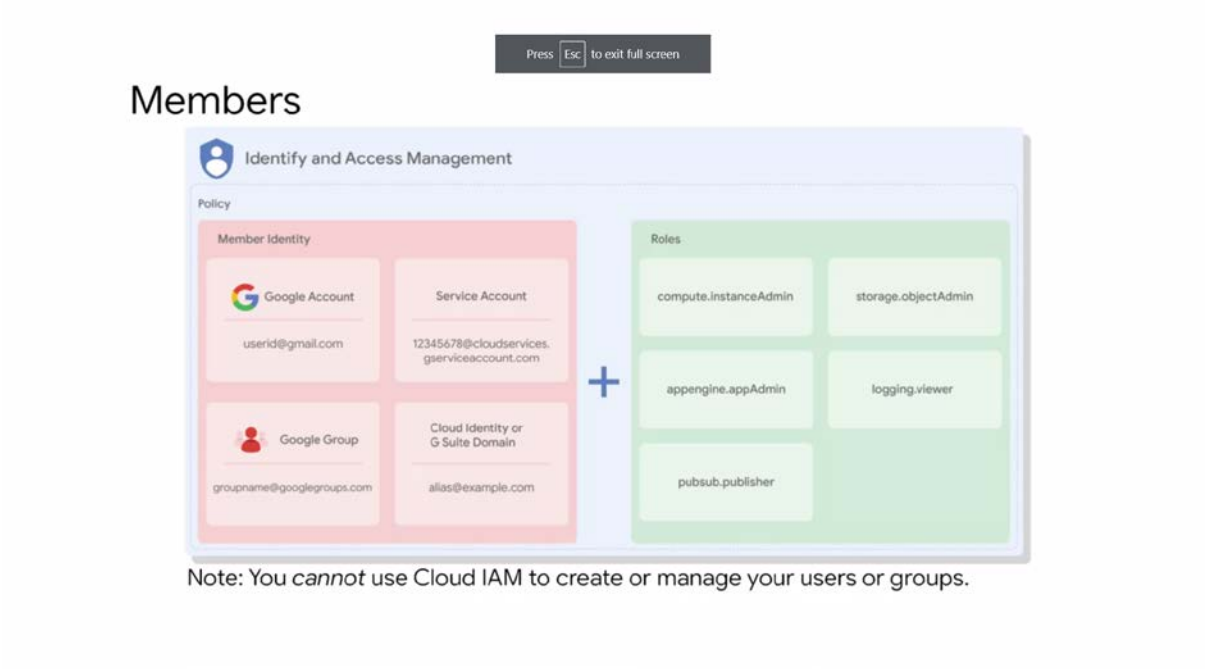
- Each project can have multiple owners, editors, viewers, and billing administrators. GCP services offer their own set of predefined roles and they define where those roles can be applied.
- This provides members with granular access to specific GCP resources and prevents unwanted access to other resources.
- These roles are collections of permissions because to do any meaningful operations, you usually need more than one permission.
- For example, as shown here a group of users is granted the instance admin role on Project A. This provides the users of that group with all the Compute Engine permissions Grouping these permissions into a role makes them easier to manage. The permissions themselves are
- classes and methods in the APIs. For example, compute.instances.start can be broken down into the service, resource, and verb. That means that this permission is used to start and stop Compute Engine instance. These permissions usually align with the actions corresponding REST API.
- Compute Engine has several **predefined IAM roles**. Let's look at three of those.

Compute Engine IAM roles

Role Title	Description
Compute Admin	Full control of all Compute Engine resources (compute.*)
Network Admin	Permissions to create, modify, and delete networking resources, except for firewall rules and SSL certificates
Storage Admin	Permissions to create, modify, and delete disks, images, and snapshots

- The Compute Admin role provides full control of all Compute Engine resources. This includes all permissions that start with Compute which means that every action for any type of Compute Engine resource is permitted.
- The Network Admin role contains permissions to create, modify, and delete networking esources
- except for firewall rules and SSL certificates. In other words, the network admin role allows read-only access to firewall rules, SSL certificates, and instances to view their ephemeral IP addresses.
- The Storage Admin role contains permissions to create, modify, and delete disks, images and snapshots. Or example, if your company has someone who manages project images and you don't want them to have the editor role on the project, grant their account the Storage Admin role on the project.
- But what if one of those roles does not have enough permissions or you need something even finer-grained? That's what **Custom Roles** permit.
- A lot of companies use the **least privileged model** in which each person in your organization is given the minimal amount of privilege needed to do their job.

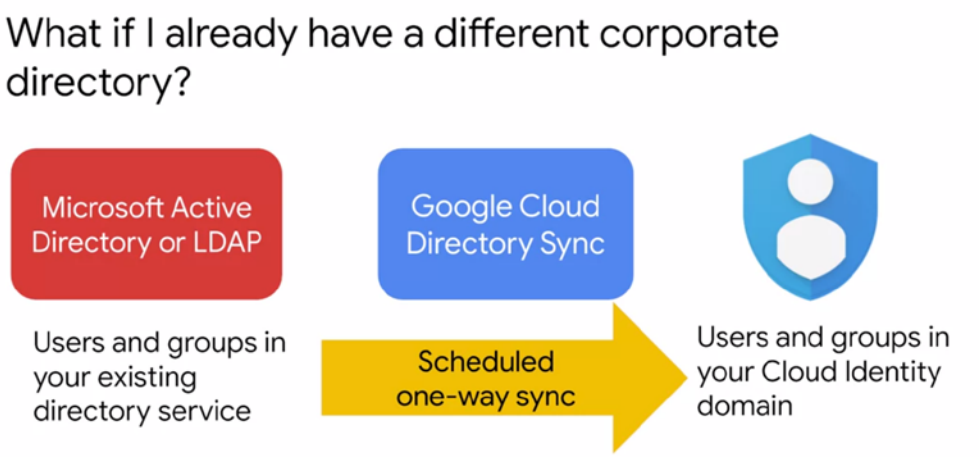
MEMBERS



- Members : define the who part, of who can do what, on which resource.
- There are five different types of members.
    - Google Accounts, Service Accounts,
    - Google Groups,
    - G Suite Domains,
    - and Cloud Identity Domains.
- A **Google Account** represents a developer, an administrator or any other person
- who interacts with GCP. Any email address that is associated with a Google Account can be an identity, including gmail.com or other domains. New users can sign up for a Google Account, by going to the Google Account sign-up page, without receiving mail through Gmail.
- A **Service Account** is an account that belongs to your application, instead of to an individual end user. When you run code that is hosted on GCP, you specify the account that the code should run as. You can create as many Service Accounts as needed to represent the different logical components of your application.
- A **Google Group** is a named collection of Google Accounts and Service Accounts. Every group has a unique email address that is associated with the group. Google Groups are a convenient way to apply an access policy to a collection of users.
- You can grant and change access controls for a whole group at once, instead of granting or changing access controls one at a time, for individual users or Service Accounts.
- G Suite Domains, represent your organization's Internet domain name, such as example.com.
- When you add a user to your G Suite Domain, a new Google Account is created for the user inside this virtual group such as, username@example.com.
- GCP customers who are not G Suite customers can get the same capabilities through Cloud Identity. **Cloud Identity**, lets you manage users in groups using the Google Admin Console.
- But you do not pay for or receive G Suite's collaboration products such as Gmail, Docs, Drive, and Calendar. Cloud Identity is available and free in Premium Editions. The Premium Edition adds capabilities for mobile device management.
- You cannot use Cloud IAM to create or manage your users or groups.
- Instead, you can use Cloud Identity or G Suite to create and manage users.
- Migrating from existing directories to GCP : Using Google Cloud Directory Sync, your administrators can log in and manage GCP resources, using the same usernames and passwords they already use.
- This tool synchronizes users and groups, from your existing Active Directory or LDAP system, with the users and groups in your Cloud Identity Domain. The synchronization is one

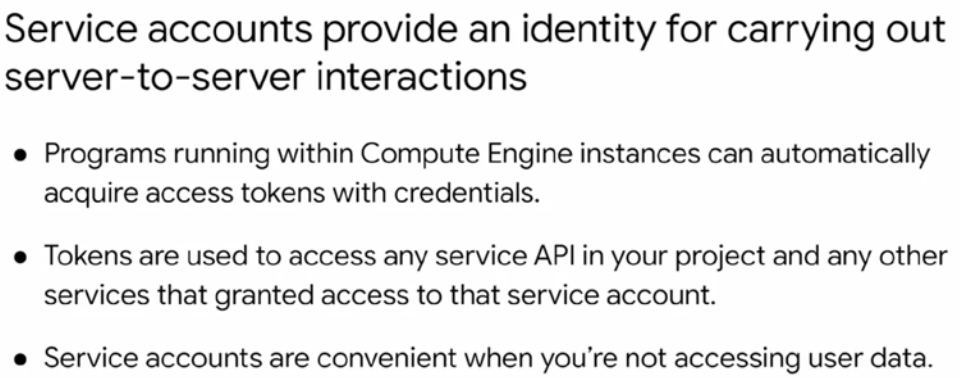
way only, which means that no information in your Active Directory or LDAP map is modified.

- **Google Cloud Directory Sync**, is designed to run scheduled synchronizations without supervision, after its synchronization, rules are set up.

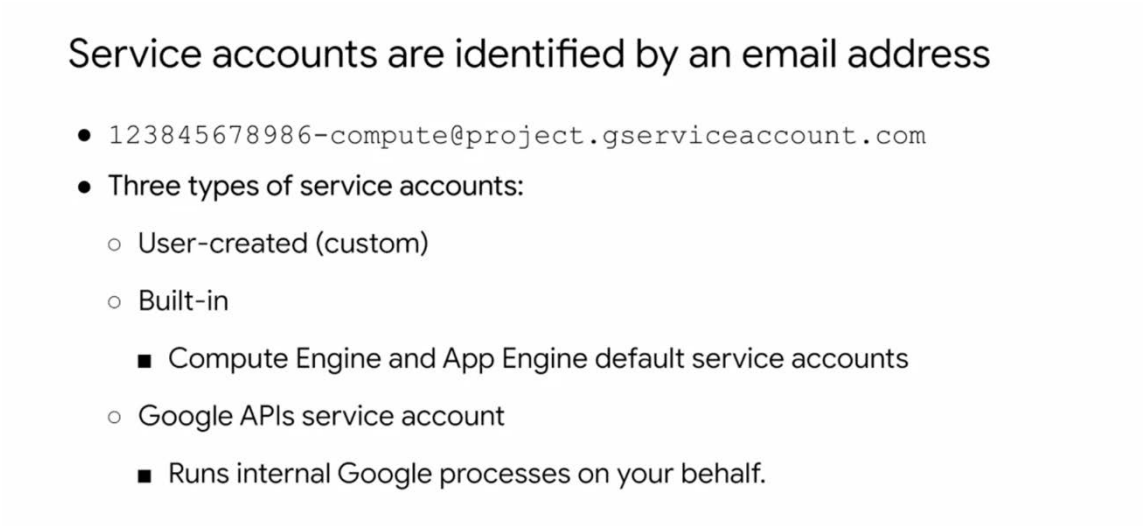


- GCP also provides **Single Sign-On Authentication**. If you have your identity system, you can continue using your own system, and processes with SSL configured. When user authentication is required, google will redirect to your system. If the user is authenticated in your system, access to Google Cloud Platform is given. Otherwise, the user is prompted to sign in. This allows you to also revoke access to GCP. If your existing authentication system support SAML2, SSO configuration, is as simple as three links (sign in page URL, sign out page URL, change password URL) and a certificate(validation certificate). Otherwise, you can use a third-party solution like ADFS, Ping or Okta.
- Also, if you want to use a Google account, but are not interested in receiving mail through Gmail, you can still create an account without Gmail.

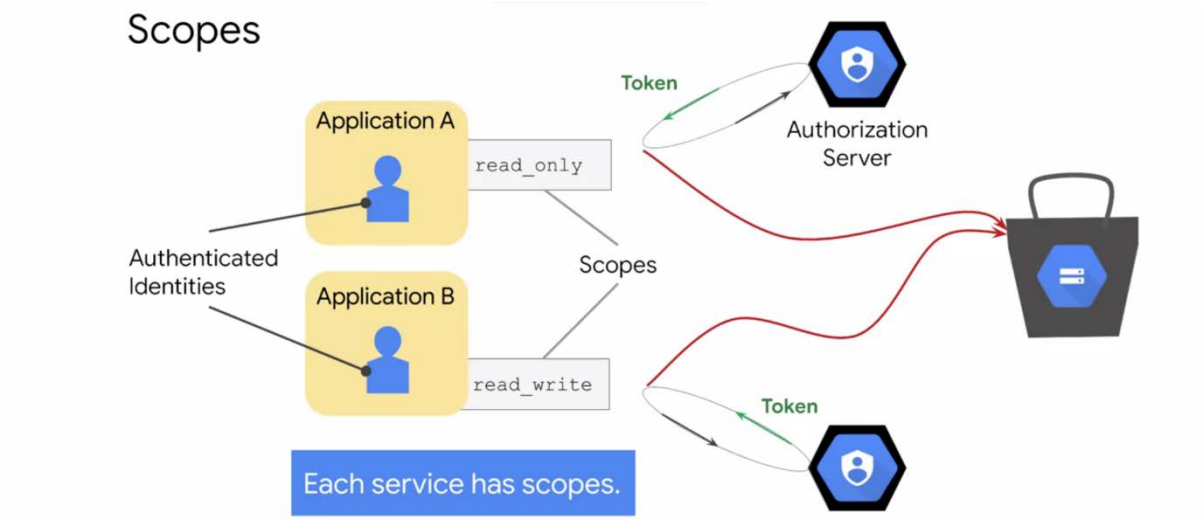
SERVICE ACCOUNTS



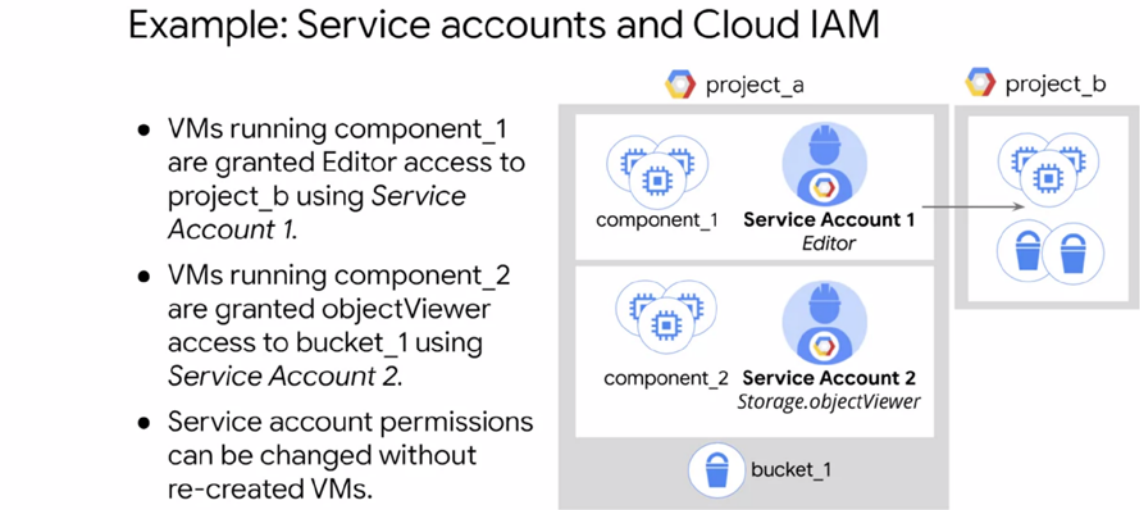
- A service account is an account that belongs to your application instead of to an individual end user. This provides an identity for carrying out server-to-server interactions in a project without supplying user credentials.
- For example, if you write an application that interacts with Google Cloud Storage, it must first authenticate to either the Google Cloud Storage XML API or JSON API.
- You can enable service accounts and grant read write access to the account on the instance where you plan to run your application, then program the application to obtain credentials from the service account. Your application authenticate seamlessly to the API without embedding any secret keys or credentials in your instance, image, or application code.
- Service accounts are identified by an email address,



  - There are three types of service accounts,
  - user-created or custom,
  - built-in,
  - and Google APIs service accounts.
- By default, all projects come with the **built-in** Compute Engine default service account.
- Apart from the default service account, all projects come with the **Google Cloud Platform APIs service account**, identifiable by the email, *project-number@cloudservices.gserviceaccount.com.*
- This is the service account designed specifically to run internal Google processes on your behalf, and it is automatically granted the editor role on your project.
- Alternatively, you can also start an instance with a **custom service account**. Custom service accounts provide more flexibility than the default service account, but they require more management from you. You can create as many custom service accounts as you need, assign any arbitrary access scopes, or Cloud IAM roles to them, and assign the service accounts to any Virtual Machine instance.
- Eg : Default Compute Engine service account : this account is automatically created per project. This account is identifiable by the email, project-number-compute@developer.gserviceaccount.com, and it is automatically granted the editor role on a project. When you start a new instance using GCloud, the default service account is enabled on that instance.
- You can override this behavior by specifying another service account or by disabling service accounts for the instance.
- Now, **authorization** is the process of determining what permissions an authenticated identity has on a set of specified resources. Scopes are used to determine whether an authenticated identity is authorized.



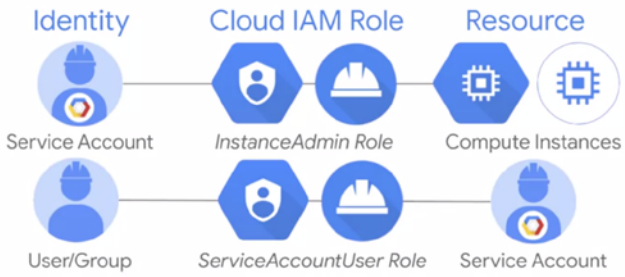
- Scopes can be customized when you create an instance using the default service account as shown in this screenshot. These scopes can be changed after an instance is created by stopping in.
- Access scopes are actually the legacy method of specifying permissions for your VM. Before the existence of IAM roles, access scopes were the only mechanism for granting permissions to service accounts.
- For user-created service accounts, use Cloud IAM roles instead to specify permissions. Another distinction between service accounts is that default service account support both primitive and predefined roles.



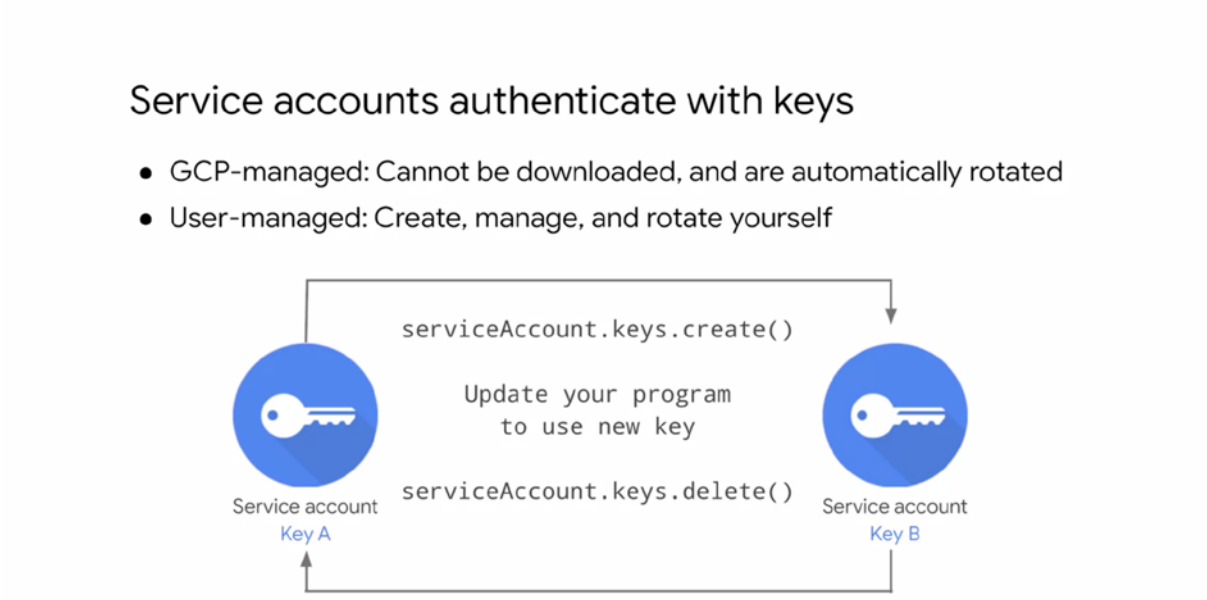
- But user-created service accounts only use predefined IAM roles. Now, roles for service accounts can also be assigned to groups or users.

Service account permissions

- Default service accounts: primitive and predefined roles
- User-created service accounts: predefined roles
- Roles for service accounts can be assigned to groups or users

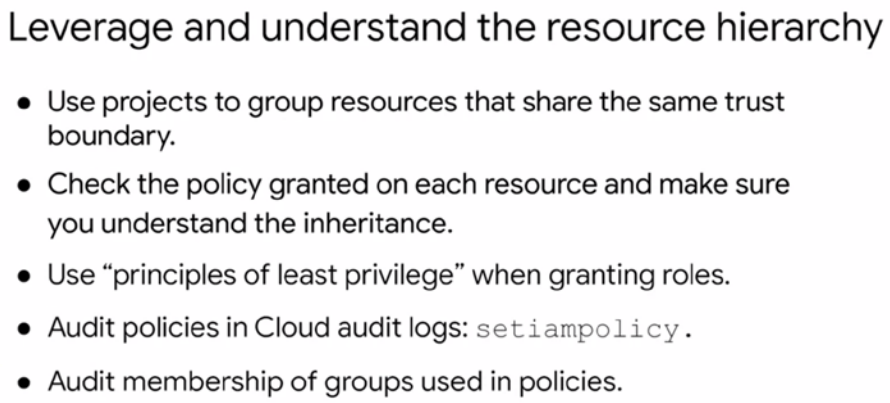


    - First, you created a service account that has the instance admin role, which has permissions to create, modify, and delete Virtual Machine instances and disks.
    - Then, you treat this service account as the resource, and decide who can use it by providing users or a group with the service account user role. This allows those users to act as that service account to create, modify, and delete Virtual Machine instances and disks.
    - Users who are service account users for a service account can access all the resources of the service account has access to. Therefore, be cautious when granting the service account user role to a user or group.
- Essentially, Cloud IAM lets you slice a project into different microservices, each with access to different resources by creating service accounts to represent each one.
- You assign the service accounts to the VMs when they are created, and you don't have to ensure the credentials are being managed correctly because GCP manages security for you.
- Although users require a username and password to authenticate, service accounts use keys.



- There are two types of service account keys:
    - GCP-managed keys and user-managed keys. GCP-managed keys are used by GCP services, such as App Engine and Compute Engine. These keys cannot be downloaded, and are automatically rotated and used for a maximum of two weeks.
    - User-managed keys are created, downloadable, and managed by users. When you create a new key pair, you download the private key, which is not retained by Google. With user-managed keys, you are responsible for security of the private key and other management operations such as key rotation.

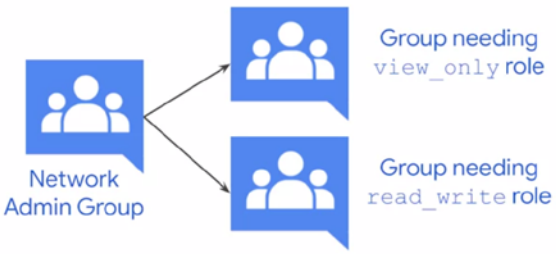
CLOUD IAM : BEST PRACTICES



Managing Roles : grant roles to groups instead of individuals. This allows you to update group membership instead of changing a Cloud IAM policy.

Grant roles to Google groups instead of individuals

- Update group membership instead of changing Cloud IAM policy.
- Audit membership of groups used in policies.
- Control the ownership of the Google group used in Cloud IAM policies.



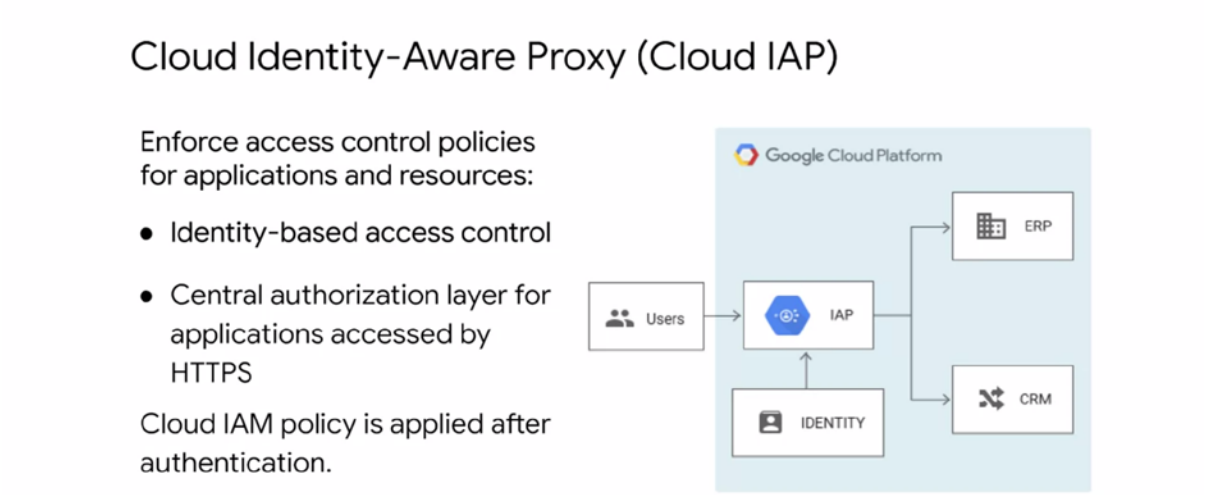
If you do this, make sure to audit membership of groups used in policies and control the ownership of the Google group used in Cloud IAM policies. You can also use multiple groups to get better control.
Using service accounts :

Service accounts

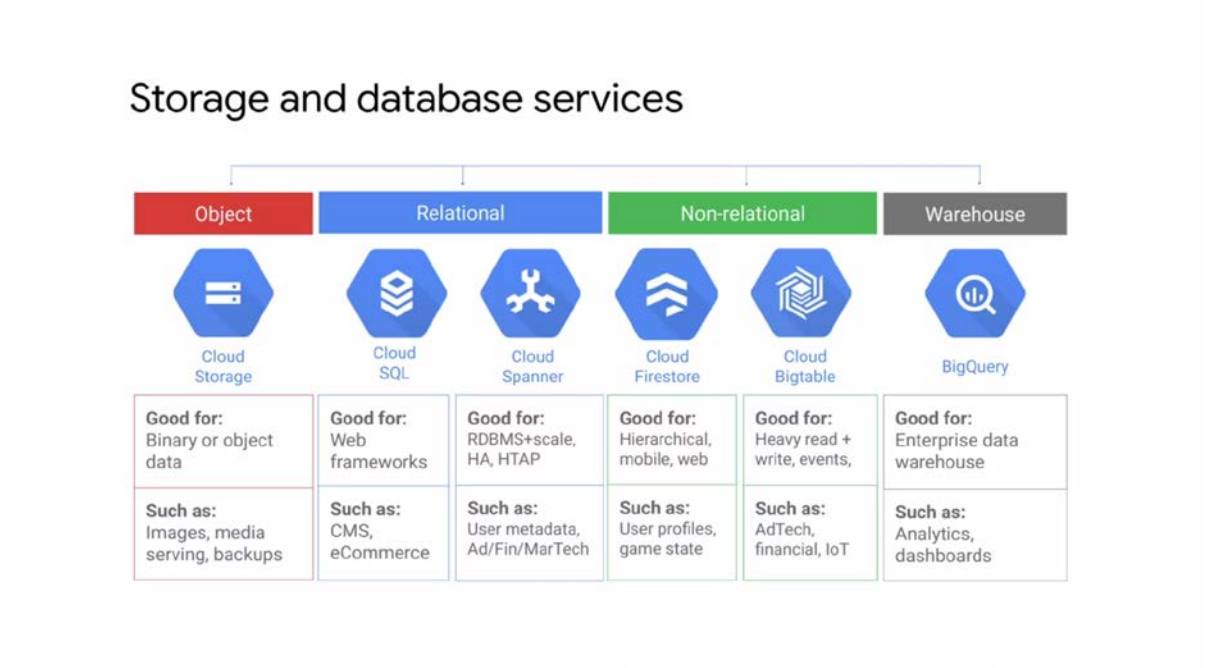
- Be very careful granting `serviceAccountUser` role.
- When you create a service account, give it a display name that clearly identifies its purpose.
- Establish a naming convention for service accounts.
- Establish key rotation policies and methods.
- Audit with `serviceAccount.keys.list()` method.

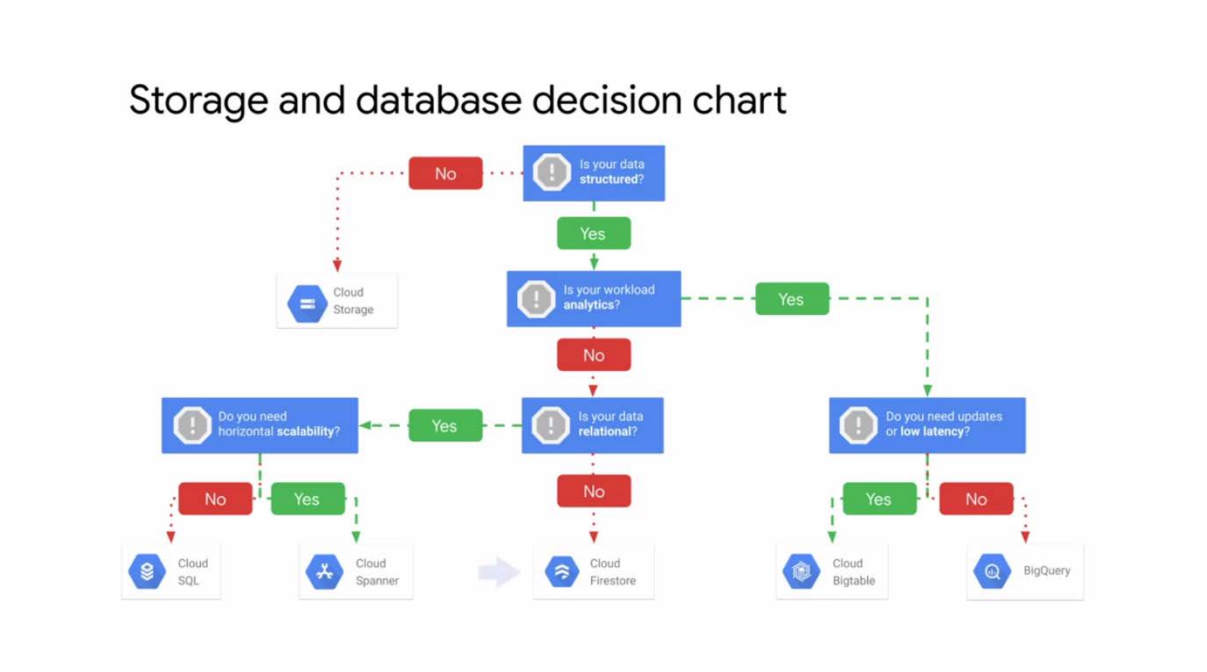
Cloud Identity Aware Proxy or Cloud IAP. : Cloud IAP lets you establish a central authorization layer for applications accessed by HTTPS. So you can use an application level access control model instead of relying on network level firewalls.
Applications and resources protected by Cloud IAP can only be accessed through the proxy by users and groups with the correct Cloud IAM role.
When you grant a user access to an application or resource by Cloud IAP, they're subject to the fine-grained access controls implemented by the product in use without requiring a VPN.

Cloud IAP performs authentication and authorization checks when a user tries to access a Cloud IAP secure resource



STORAGE





- Cloud Storage is Google Cloud's object storage service, and it allows worldwide storage and retrieval of any amount of data at anytime.
- You can use Cloud Storage for a range of scenarios including serving website content, storing data for archival and disaster recovery, or distributing large data objects to user via direct download.

Cloud Storage has a couple of key features.

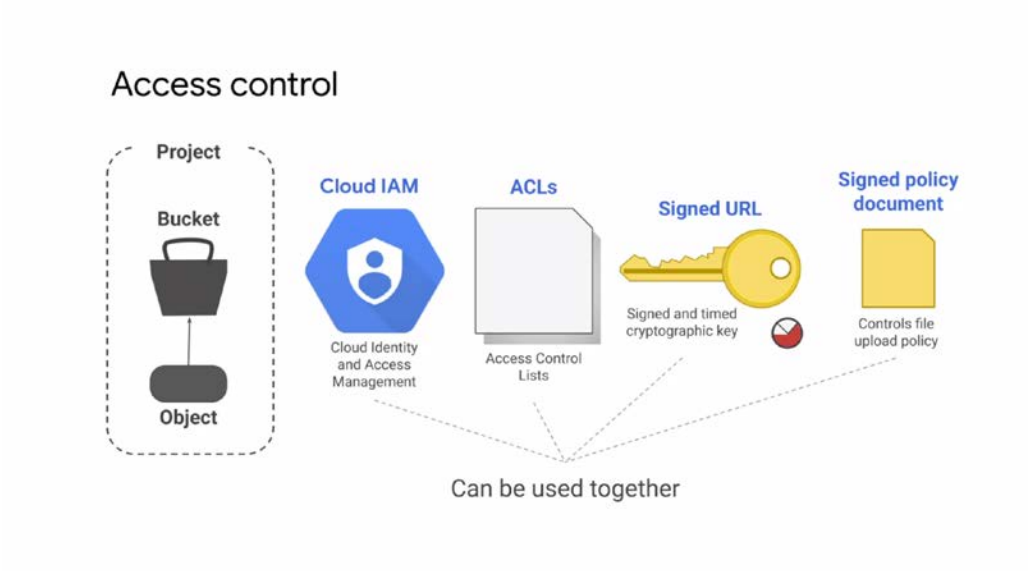
1. It's scalable to exabytes of data.
2. The time to first byte is in milliseconds.
3. It has very high availability across all storage classes and
4. it has a single API across those storage classes.

- Some like to think of Cloud Storage as files in a file system, but it's not really a file system. Instead, Cloud Storage is a collection of buckets that you place objects into. You can create directory, so to speak, but really directory is just another object that points to different objects in the bucket.
- You're not going to easily be able to index all of these files like you would in a file system. You just have a specific URL to access objects.
- Cloud Storage has four storage classes.

Overview of storage classes

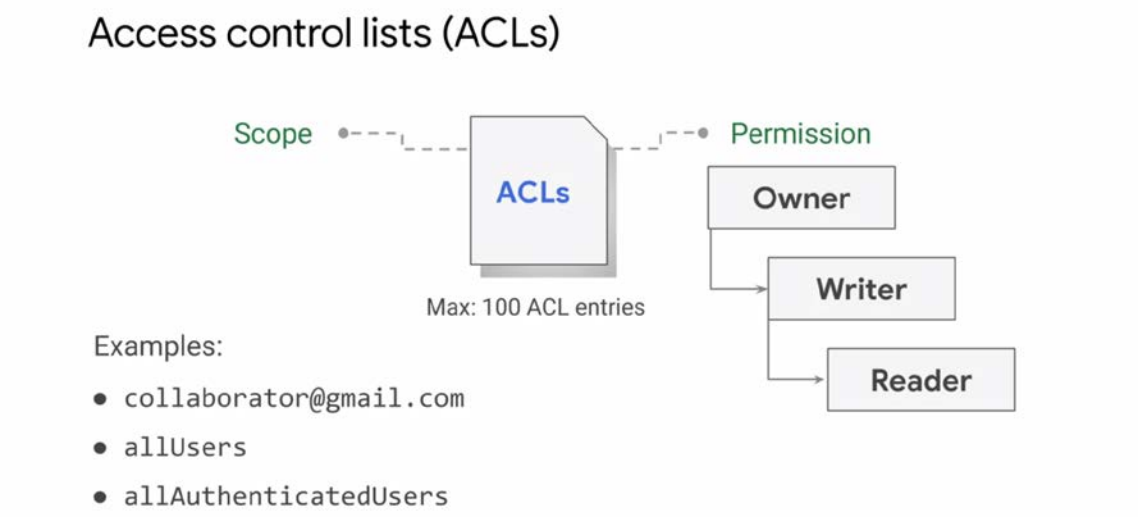
	Standard	Nearline	Coldline	Archive
Use case	"Hot" data and/or stored for only brief periods of time like data-intensive computations	Infrequently accessed data like data backup, long-tail multimedia content, and data archiving	Infrequently accessed data that you read or modify at most once a quarter	Data archiving, online backup, and disaster recovery
Minimum storage duration	None	30 days	90 days	365 days
Retrieval cost	None	$0.01 per GB	$0.02 per GB	$0.05 per GB
Availability SLA	99.95% (multi/dual) 99.90% (region)	99.90% (multi/dual) 99.00% (region)		None
Durability	99.999999999%			

- Standard, nearline, coldline, and archive in each of those storage classes provide three location types.
    1. **Multi region** which is a large geographic areas such as the United States that contains two or more geographic places. Object stored in a multi region or dual region are geo redundant.
    2. **Dual region** is a specific pair of regions such as Finland and the Netherlands.
    3. A **region** is a specific geographic place such as London.
- **Standard** storage is fest for data that is frequently accessed. And are stored for only brief periods of time. This is the most expensive storage class, but it has no minimum storage duration and no retrieval cost.
- When used in a region, standard storage is appropriate for storing data in the same location as Google, Kubernetes engine clusters or compute engine instances that use the data. Co locating your resources maximizes the performance for data intensive computations and can reduce network charges.
- When used in a dual region, you still get optimized performance when accessing Google Cloud products that are located in one of the associated regions. But you also get improved availability that comes from storing data in geographically separate locations. When used in multi region, standard search is appropriate for storing data that is accessed around the world. Such as serving website content, stream videos, executing interactive workloads, or serving data supporting mobile and gaming applications.
- **Nearline** storage is a low cost-, highly durable storage service for storing infrequently accessed data like data backup, long tailed multimedia content, and data archiving. Nearline storage is a better choice than standard storage in scenarios were slightly lower availability, authority day, minimum storage duration, and costs for data access are acceptable tradeoffs for lowered at less storage costs.
- **Coldline** storage is a very lowcost-, highly durable storage service for storing infrequently accessed data. Coldline storage is a better choice than standard storage or nearline storage.
- In scenarios where slightly lower availability, a 90 day minimum storage duration and higher costs for data access are acceptable tradeoffs for lowered address storage costs.
- **Archive** storage is the lowest cost, highly durable storage service for data archiving, online backup and disaster recovery. Unlike the source, big coldest storage service offered by other cloud providers, your data is available within milliseconds, not hours or days. Unlike others, cloud storage classes, archive storage has no availability SLA. Though the typical availability is comparible to nearline and coldline storage.
- Archive storage also has higher costs for data access and operations as well as a 365 day minimum storage duration.
- Archive storage is the best choice for data that you plan to access less than once a year.
- Cloud storage is broken down into a couple of different items here.
- **Buckets,** which are required to have a globally unique name and cannot be nested.
- The data that you put into those buckets are **objects** that inherit the storage class of the bucket, and those objects could be text files, Doc files, video files, etc. There's no minimum size to those objects, and you can't scale this as much as you want, as long as your quota allows it.
- To access the data you can use the gsutil command or either JSON or XML APIs.
- When you upload an object to a bucket, the object is assigned til bucket storage class unless you specify a storage class for the object.
- We can use IAM for the project to control which individual users or service account can see the bucket, list the objects in the bucket, view the names of the objects in the bucket or create new buckets.



1. For most purposes **Cloud IAM** is sufficient and roles are inherited from project to bucket to object.
2. **Access control lists** or ACLs offer finer control for even more detailed control.
3. **Signed URL** provide a cryptographic key that gives tying limited access to a bucket or object.
4. Finally, a **signed policy document** further refines the control by determining what file can be uploaded by someone with a signed URL.

- Let's take a closer look at ACLs and signed URLs.



- An ACL is a mechanisms used to define who has access to your buckets and objects, as well as what the level of access to have. The maximum number of ACL entries you can create for a bucket or object is 100.
- Each ACL consists of one or more entries, and these entries consists of two pieces of information.
    1. A **scope** which defines who can perform the specified actions.
    2. For example, a specific user or group of users.
    3. And the **permission** which defines what actions can be performed.
    4. For example, read or write.
- The allUsers identifier listed under slide represents anyone who is on the Internet, with or without a Google account.
- The allAuthenticatedUsers identifier, in contrast, represents anyone who is authenticated with a Google account.

- For some applications it is easier and more efficient to grant limited time access tokens that can be used by any user instead of using account based authentication for controlling resource access. For example, when you don't want to require users to have a Google account.
- Signed URLs allow you to do this for cloud storage.
- You create a URL, the grants read or write access to a specific cloud storage resource, and specifies when this access expires. That URL is signed using a private key associated with their service account.
- When the request is received, Cloud Storage can verify that the axis granting URL was issued on behalf of a trusted security principle. In this case, the service account and delegates its trust of that account to the holder of the URL. After you give out these signed URL, it is out of your control.
- So you want these signed URL to expire after some reasonable amount of time.

Object Versioning supports the retrieval of objects that are deleted or overwritten

- Objects are immutable.
- Object Versioning:
  - Maintain a history of modifications of objects.
  - List archived versions of an object, restore an object to an older state, or delete a version.

Object Lifecycle Management policies specify actions to be performed on objects that meet certain rules

- Examples:
  - Downgrade storage class on objects older than a year.
  - Delete objects created before a specific date.
  - Keep only the 3 most recent versions of an object.
- Object inspection occurs in asynchronous batches.
- Changes can take 24 hours to apply.

The Object Life Cycle Management policies are set of rules that apply to all objects in the buckets. When an object meets the criteria of one of the rules, Cloud Storage automatically performs a specified action on the object

Data import services

- **Transfer Appliance:** Rack, capture and then ship your data to Google Cloud.
- **Storage Transfer Service:** Import online data (another bucket, an S3 bucket, or web source).
- **Offline Media Import:** Third-party provider uploads the data from physical media.

Cloud Storage provides strong global consistency

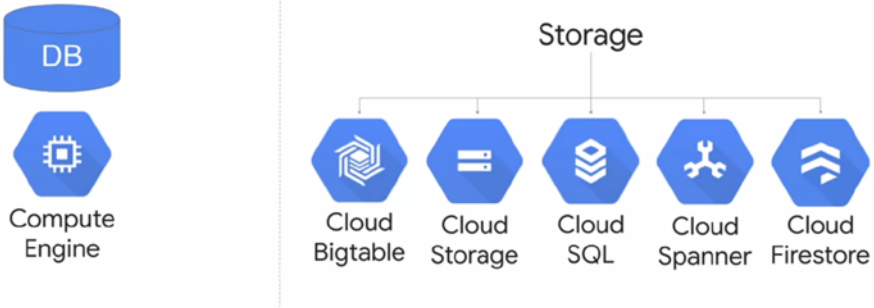
- Read-after-write
- Read-after-metadata-update
- Read-after-delete
- Bucket listing
- Object listing

Cloud Storage features

- Customer-supplied encryption key (CSEK)
  - Use your own key instead of Google-managed keys
- Object Lifecycle Management
  - Automatically delete or archive objects
- Object Versioning
  - Maintain multiple versions of objects
- Directory synchronization
  - Synchronizes a VM directory with a bucket
- Object change notification
- Data import
- Strong consistency

CLOUD SQL : Cloud SQL is a fully managed service of either MySQL, PostgreSQL or Microsoft SQL server databases.

Build your own database solution or use a managed service



Cloud SQL is a fully managed database service (MySQL, PostgreSQL, or Microsoft SQL Server)

- Patches and updates automatically applied
- You administer MySQL users
- Cloud SQL supports many clients
  - `gcloud sql`
  - App Engine, G Suite scripts
  - Applications and tools
    - SQL Workbench, Toad
    - External applications using standard MySQL drivers

Cloud SQL instance

Performance:

- 30 TB of storage
- 40,000 IOPS
- 416 GB of RAM
- Scale out with read replicas

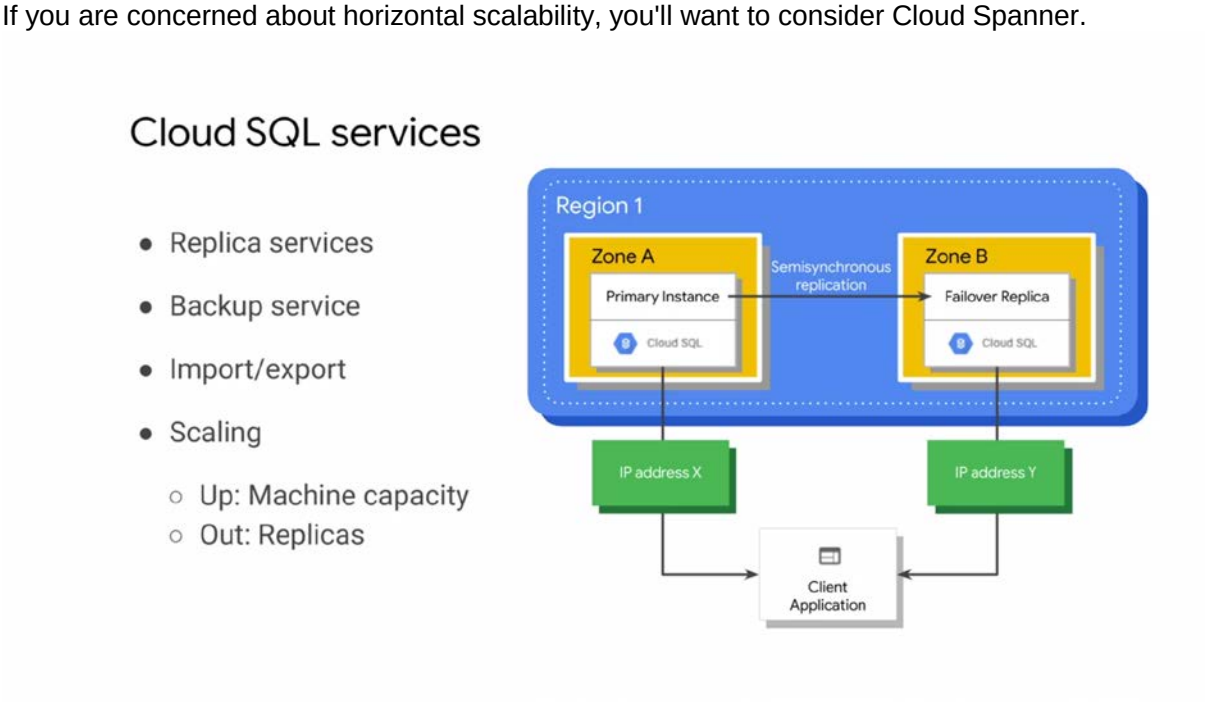
Choice:

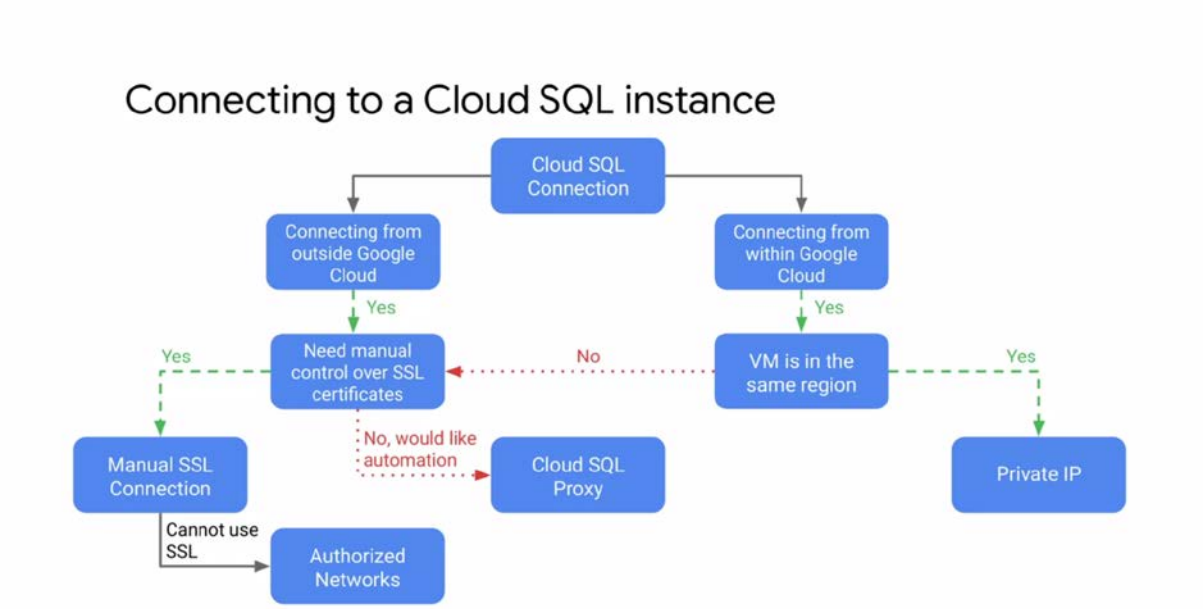
- MySQL 5.6, 5.7 (default), or 8.0
- PostgreSQL 9.6, 10, 11 or 12 (default)
- Microsoft SQL Server 2017

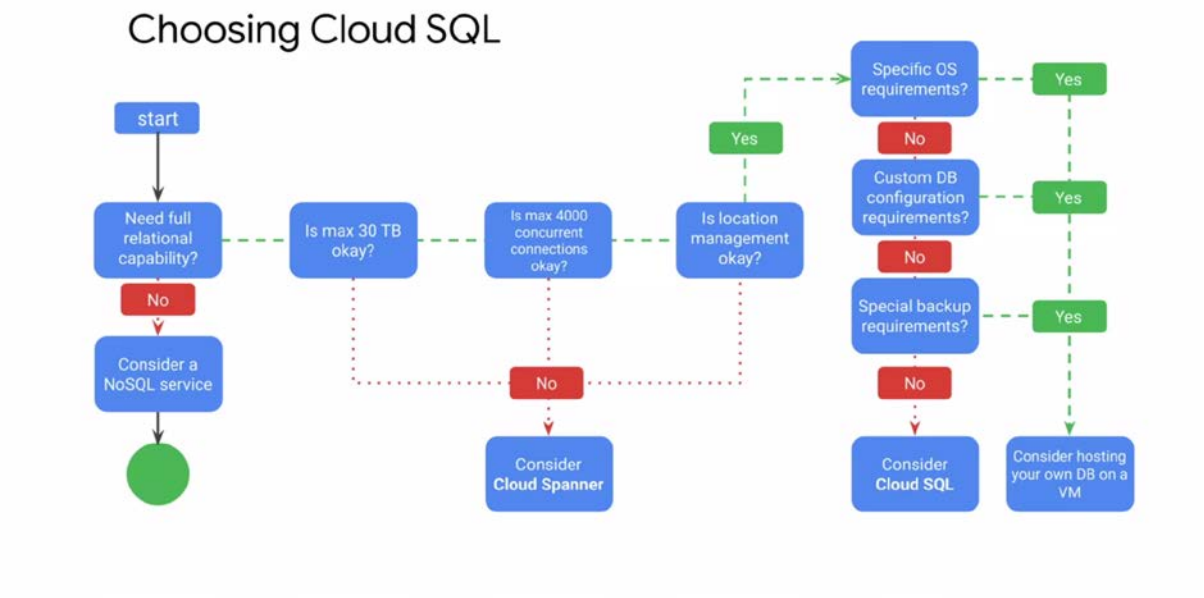
- There is a replica service that can replicate data between multiple zones as shown on the right. This is useful for automatic failover if an outage occurs.

- Cloud SQL also provides automated an on-demand backups with point in time recovery. You can import and export databases using my SQL dump or import and export CSV files.

- Cloud SQL can also scale up, which does require a machine restart or scale out using read replicas
- If you are concerned about horizontal scalability, you'll want to consider Cloud Spanner.







CLOUD SPANNER

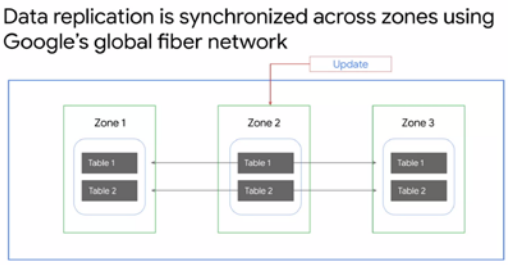
- If Cloud SQL does not fit your requirements because you need horizontal scalability, consider using Cloud Spanner.
- Cloud Spanner is a service built for the Cloud, specifically to combine the benefits of relational database structure with non-relational horizontal scale.
- This service can provide petabytes of capacity and offers transactional consistency at global scale, schemas, SQL, and automatic synchronous replication for high availability.
- Use cases include financial applications and inventory applications, traditionally served by relational database technology.

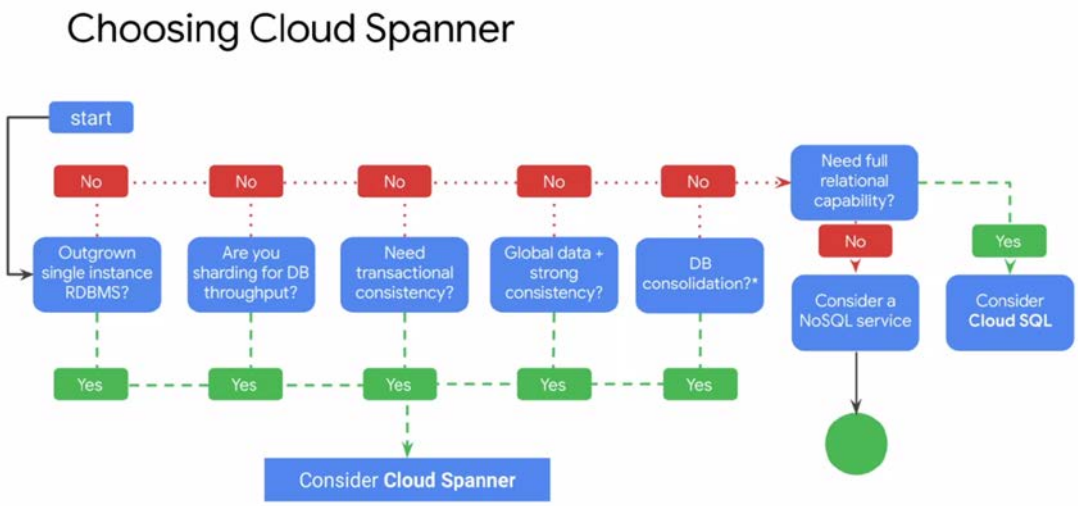
- Depending on whether you create a multi-regional or regional instance, you'll have different monthly up-time SLAs

- Cloud Spanner offers the best of the relational and non-relational worlds. These features allow for mission-critical use cases such as building consistent systems for transactions and inventory management in the financial services in retail industries.

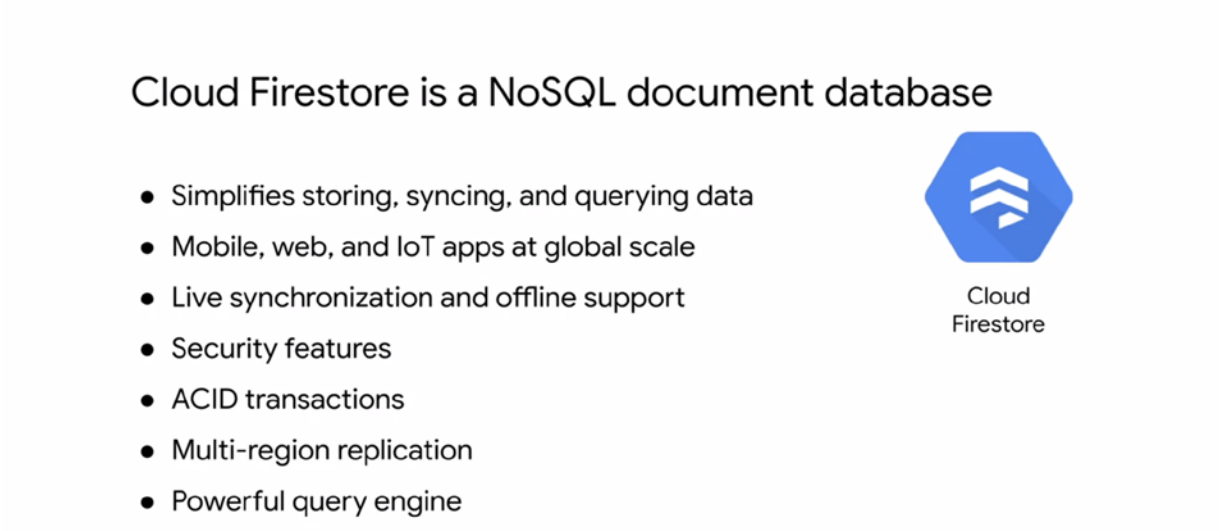
Characteristics

	Cloud Spanner	Relational DB	Non-Relational DB
Schema	✓ Yes	✓ Yes	✗ No
SQL	✓ Yes	✓ Yes	✗ No
Consistency	✓ Strong	✓ Strong	✗ Eventual
Availability	✓ High	✗ Failover	✓ High
Scalability	✓ Horizontal	✗ Vertical	✓ Horizontal
Replication	✓ Automatic	↻ Configurable	↻ Configurable

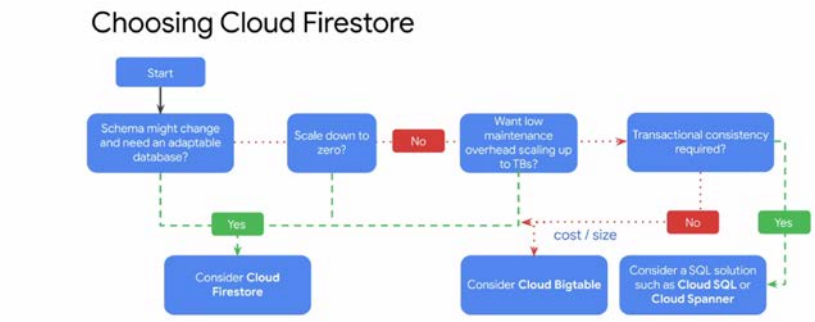




CLOUD FIRESTORE



- Cloud Firestore can operate in Datastore mode, making it backwards compatible with Cloud Datastore. By creating a Cloud Firestore database in Datastore mode, you can access Cloud Firestore's improved storage layer while keeping Cloud Datastore system behaviour.



- Cloud Firestore is backward compatible with Cloud Datastore, but the new data model real-time updates in mobile and web client library features are not. A general guideline is to use Cloud Firestore in data store mode for new server projects and native mode for new mobile and web apps. As the next generation of Cloud Datastore, Cloud Firestore is

compatible with all cloud Datastore APIs and client libraries. Existing Cloud Datastore users will be live upgraded to Cloud Firestore automatically at a future date.

CLOUD BIGTABLE

If you don't require transactional consistency, you might want to consider Cloud Bigtable.

- Cloud Bigtable is a fully managed NoSQL database with petabyte-scale and very low latency. It seamlessly scales for throughput, and it learns to adjust to specific access patterns.
- Cloud Bigtable is actually the same database that powers many of Google's core services including Search, Analytics, Maps, and Gmail.
- Cloud Bigtable is a great choice for both operational and analytical applications including IoT, user analytics, and financial data analysis because it supports high read and write throughput at low latency.
- It's also a great storage engine for machine learning applications.
- Cloud Bigtable integrates easily with popular big data tools like Hadoop, Cloud Dataflow, and Cloud Dataproc, plus Cloud Bigtable supports the open-source industry standard HBase API, which makes it easy for your development teams to get started.

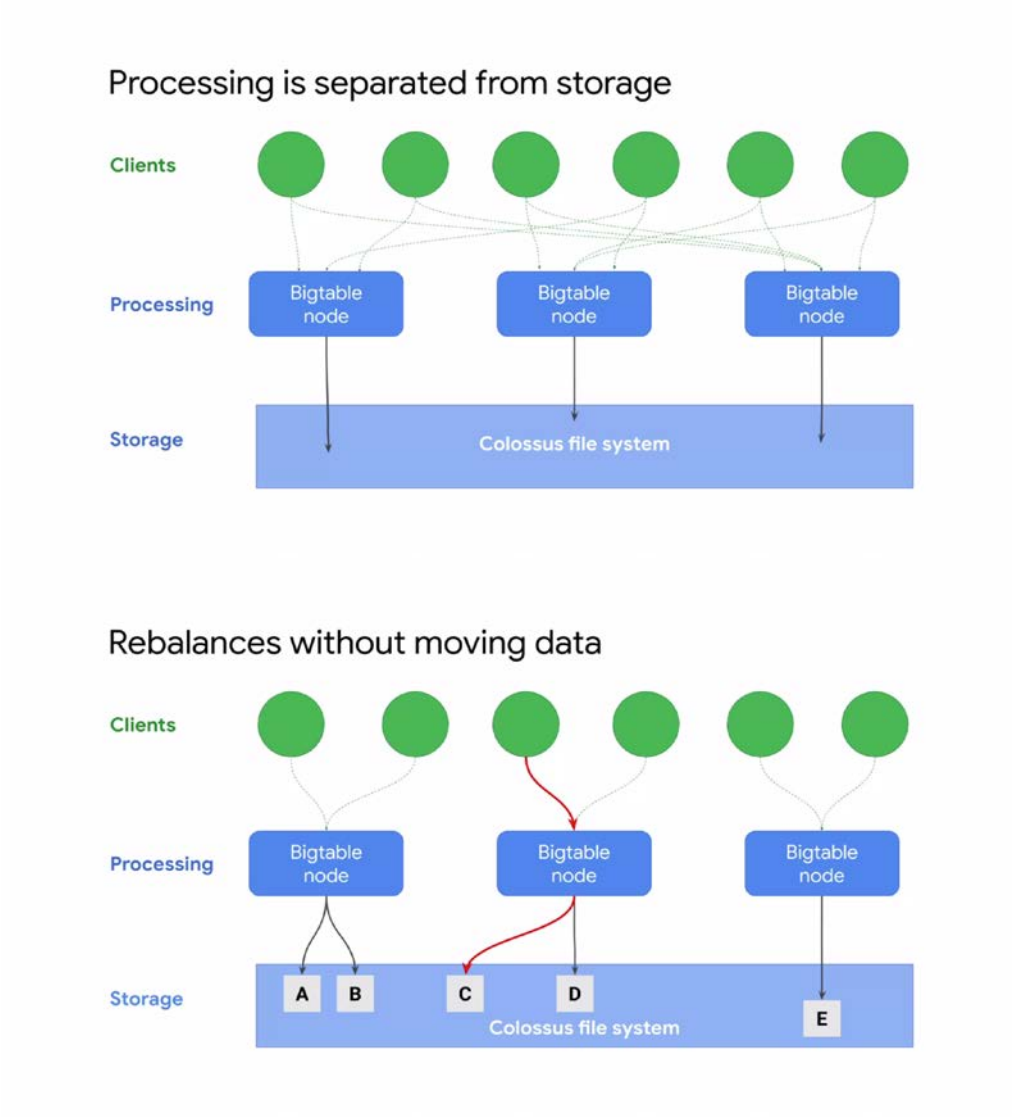
- Cloud Bigtable stores data in massively scalable tables, each of which is a sorted key value map. The table is composed of rows, each of which typically describes a single entity, and columns which contain individual values for each row.
- Each row is indexed by a single row key. Columns that are related to one another are typically grouped together into a column family. Each column is identified by a combination of the column family and a column qualifier which is a unique name within the column family.
- Each row column intersection can contain multiple cells or versions at different timestamps, providing a record of how the stored data has been altered over time.
- Cloud Bigtable tables are sparse. If a cell does not contain any data, it does not take up any space.
- The example shown here is for a hypothetical social network for United States presidents, where each president can follow posts from other presidents.

- The table contains one column family, the follows family. This family contains multiple column qualifiers. Column qualifiers are used as data.

- This design choice takes advantage of the sparseness of Cloud Bigtable tables and the fact that new column qualifiers can be added as your data changes. The username is used as the row key.

- Assuming usernames are evenly spread across the alphabet, data access will be reasonably uniform across the entire table.

- This diagram shows a simplified version of Cloud Bigtables overall architecture. It illustrates that processing which is done through a front end server pool and nodes is handled separately from the storage.

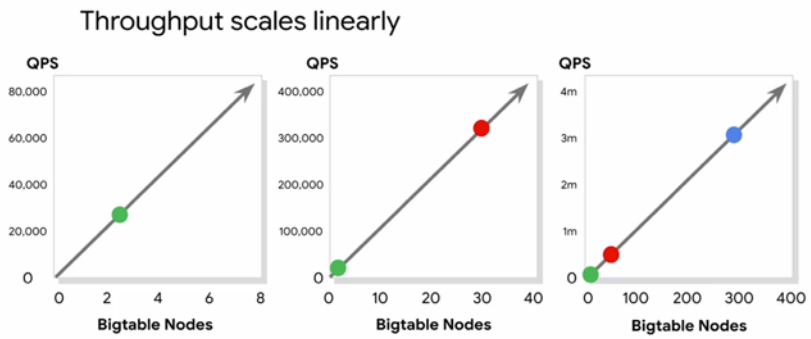


- A Cloud Bigtable table is sharded into blocks of contiguous rows called tablets, to help balance the workload of queries. Tablets are similar to HBase regions, Tablets are stored on Colossus which is Google's file system in SSTable format.

- An SSTable provides a persistent, ordered, immutable map from keys to values where both keys and values are arbitrary byte strings. Cloud Bigtable learns to adjust to specific access patterns.

- If a certain Bigtable node is frequently accessing a certain subset of data, Cloud Bigtable will update the indexes so that other nodes can distribute that workload evenly as shown here.

- That throughput scales linearly, so for every single node that you do add, you're going to see a linear scale of throughput performance up to hundreds of nodes.

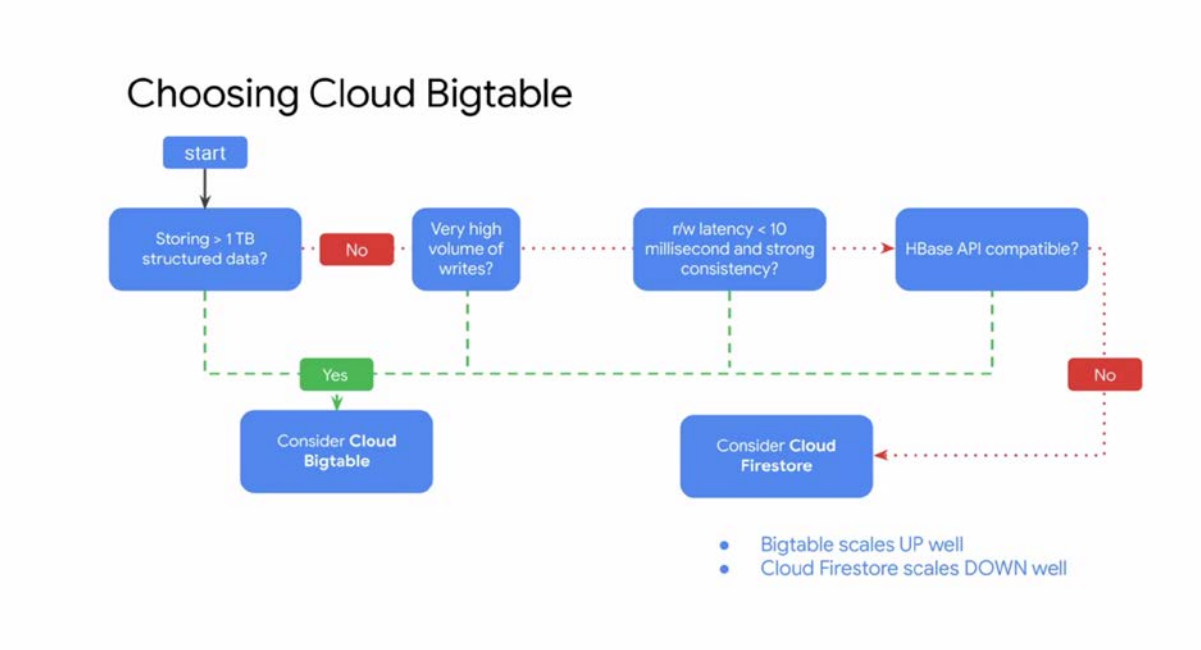


- In summary, if you need to store more than one terabyte of structured data, have very high volumes of writes. Need read write latency of less than 10 milliseconds along with strong consistency or need a storage service that is compatible with the HBase API. Consider using Cloud Bigtable.

- If you don't need any of these and are looking for a storage service that scales down well, consider using Cloud Firestore.

- The smallest Cloud Bigtable cluster you can create has three nodes and can handle 30,000 operations per second.

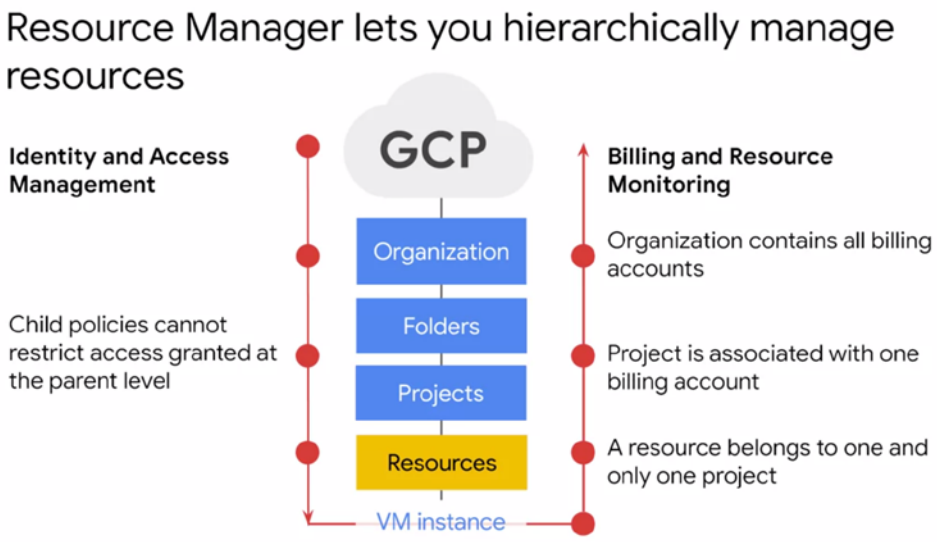
- You pay for those nodes while they are operational, whether your application is using them or not.



CLOUD MEMORYSTORE

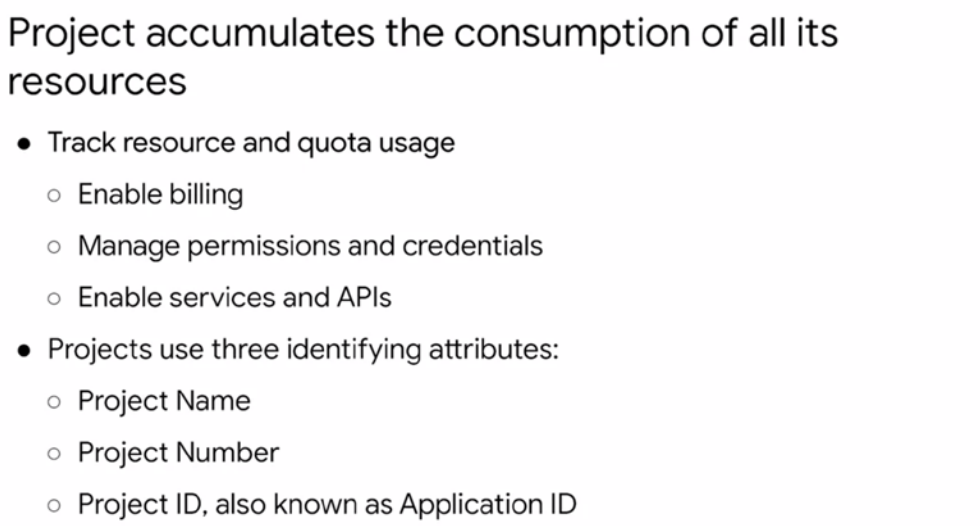
- Cloud Memorystore for Redis provides a fully managed in-memory data store service built on scalable, secure, and highly available infrastructure managed by Google.
- Applications running on GCP can achieve extreme performance by leveraging the highly scalable available secure Redis service without the burden of managing complex reddest deployments.
- This allows spending more time writing code, so that focus can be on building great apps.
- Cloud Memorystore also automates complex tasks like enabling high availability, failover, patching, and monitoring.
- High availability instances are replicated across two zones and provide a 99.9 percent availability SLA. You can easily achieve this sub-millisecond latency and throughput your applications need. Start with the lowest tier and smallest size and then grow your instance effortlessly with minimal impact to application availability.
- Cloud Memorystore can support instances of up to 300 gigabytes and network throughput of 12 gigabytes per second.
- Because Cloud Memorystore for Redis is fully compatible with the Redis protocol, you can lift-and-shift your applications from open-source Redis to Cloud Memorystore without any code changes by using the import export feature.
- There is no need to learn new tools because all existing tools in Client Libraries just work.

CLOUD RESOURCE MANAGER



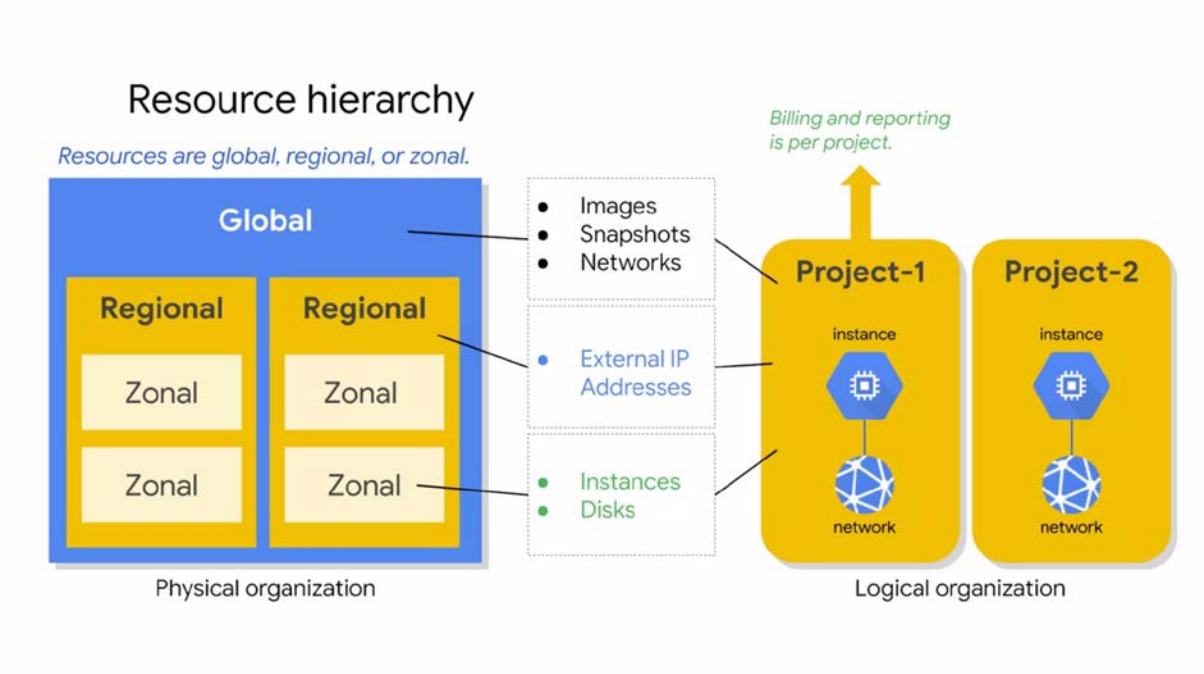
- The resource manager lets you hierarchically manage resources by project, folder, and organization.
- Although IAM policies are inherited top to bottom, billing is accumulated from the bottom up.

- Resource consumption is measured in quantities like rate of use or time, number of items, or feature use.
- Because a resource belongs to only one project, a project accumulates the consumption of all its resources.
- Each project is associated with one billing account, which means that an organization contains all billing accounts.



- A project can be identified by
    - the project name, which is a human-readable way to identify your projects, but it isn't used by any Google APIs.
    - There's also the project number, which is automatically generated by the server and assigned to your project,
    - and there is the Project ID, which is a unique ID that is generated from your project name.
- Finally, let's talk about the resource hierarchy. From a physical organization standpoint, resources are categorized as global, regional, or zonal.
    - Images, snapshots, and networks, are global resources.
    - External IP addresses are regional resources,
    - and instances and disks are zonal resources.
- However, regardless of the type, each resource is organized into a project. This enables each project to have its own billing and reporting.

QUQUQ



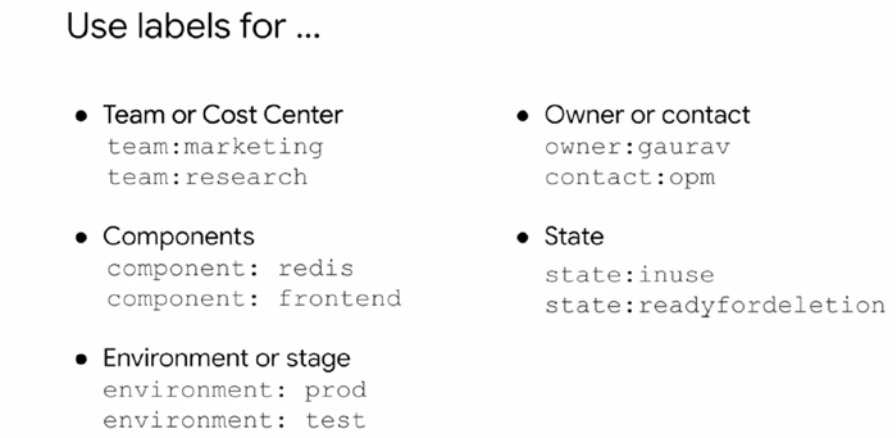
QUOTAS

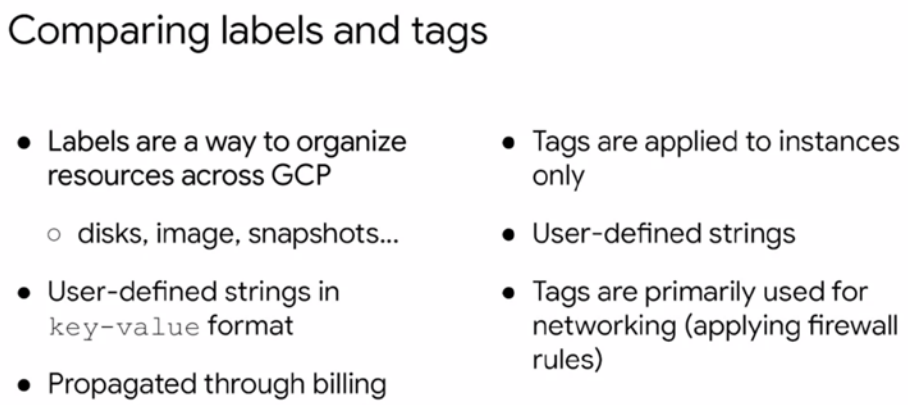
- All resources and GCP are subject to project quotas or limits. These typically fall into one of the three categories shown here.
    - How many resources you can create per project? For example, you can only have five VPC networks for project.
    - How quickly you can make API requests in a project or rate limits. For example, by default,
    - you can only make five administrative actions per second per project when using the Cloud Spanner API.
    - There are also regional quotas. For example, by default, you can only have 24 CPUs per region.
- As your use of GCP expands over time, your quotas may increase accordingly. If you expect a notable upcoming increasing usage, you can proactively request quota adjustments from the quotas page in the GCP Console. This page will also display your current quotas.
- If quotas can be changed, why do they exist?
    - Project quotas prevent runaway consumption in case of an error or malicious attack. For example, imagine you accidentally create a 100 instances instead of 10 Compute Engine instances using the gcloud command-line.
    - Quotas also prevent billing spikes or surprises. Quotas are related to billing, but we will go through how to set up budgets and alerts later, which will really help you manage billing.
    - Finally, quotas forces consideration and periodic review. For example, do you really need a 96 Core instance? Or can you go with a smaller and cheaper alternative? It is also important to mention the quotas are the maximum amount of resources you can create for that resource type as long as those resources are available.
- Quotas do not guarantee that resources will be available at all times. For example, if a region is out of local SSDs, you cannot create local SSDs in that region even if you still had quota for local SSDs.

LABELS

- Labels are a utility for organizing GCP resources. Labels are key value pairs that you can attach to your resources like VMs, disks, snapshots, and images.

- You can create and manage labels using the GCP Console, gcloud or the Resource Manager API, and each resource can have up to 64 labels. For example, you could create a label to define the environment of your virtual machines. Then you define the label for each of your instances as either production or test. Using this label, you could search and list all of your production resources for inventory purposes.
- Labels can also be used in scripts to help
    - analyze costs
    - run bulk operations on multiple resources.





BILLING

- Because the consumption of all resources under a project accumulates into one billing account, let's talk billing. To help with project planning and controlling costs. You can set a budget.
- Setting a budget lets you track how your spend is growing towards that amount.
    - First, you set a budget name and specify which project this budget applies to.
    - Then you can set the budget at a specific amount or match it to the previous month spend.
    - After you determine your budget amount, you can set the budget alerts.
    - These alerts send emails to billing admins after spend exceeds a percentage of the budget or a specified amount. In our case, it would send an email when spending reaches 50 percent, 90 percent and a 100 percent of the budget amount.

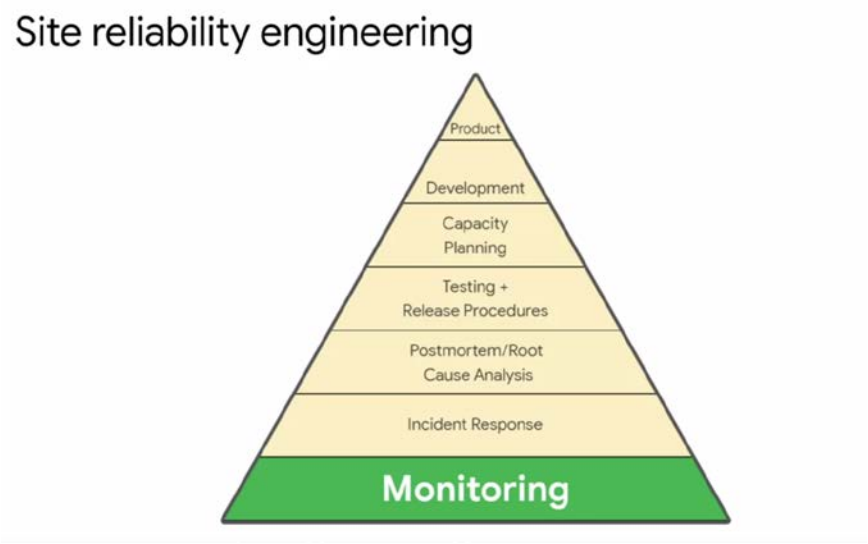
    - You can even choose to send an alert when the spend is forecasted to exceed the percent of the budget amount by the end of the budget period.
    - In addition to receiving an email, you can use Cloud Pub Sub notifications to programmatically receive spend updates about this budget. You could even create a Cloud Function that listens to the Pub Sub topic to automate cost management.
- Another way to help optimize your GCP spend is to used labels. For example, you could label VM instances that are spread across different regions. Maybe these instances are
- sending most of their traffic to a different continent which could incur higher cost.
- In that case, you might consider relocating some of those instances or using a caching service like Cloud CDN to cach content closer to your users, which reduces your networking spend.
- I recommend labeling all your resources and exporting your billing data to BigQuery to analyze your spend. BigQuery is Google's scalable, bully managed enterprise data warehouse
- with SQL and fast response times. Creating a query is damn simple
- You can even visualize spend over time with Data studio. Data Studio turns your data into an formative dashboards and reports that are easy to read, easy to share, and fully customizable. For example, you can slice and dice your billing reports using your labels.

STACKDRIVER

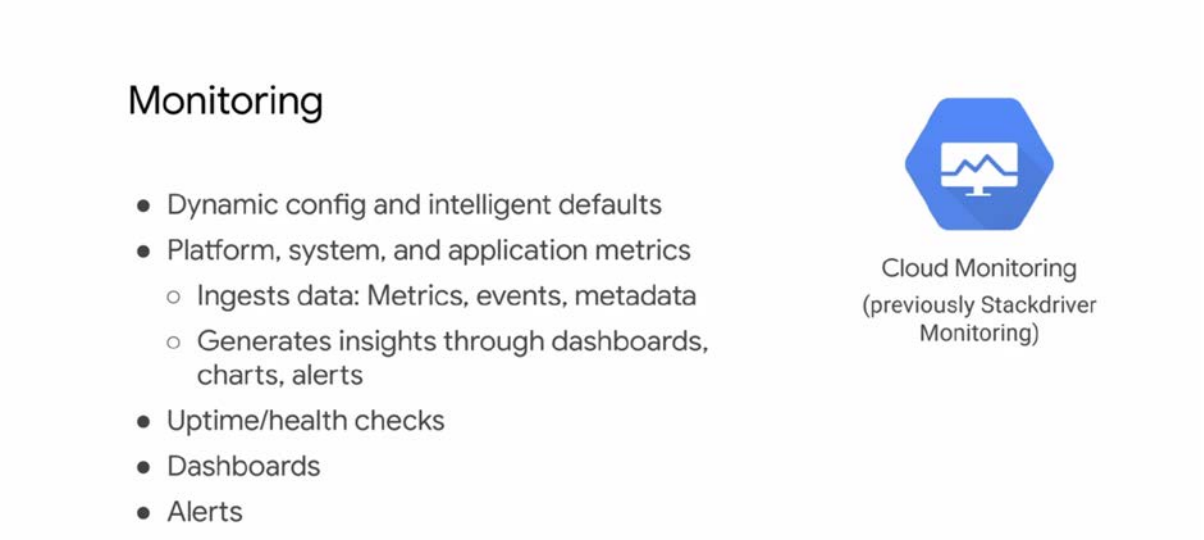
- Stackdriver dynamically discovers cloud resources and application services, based on deep integration with Google Cloud Platform and Amazon Web Service.
- Because of its smart defaults,
- you can have core visibility into your cloud platform in minutes. This provides you with access to powerful data and analytics tools.
- Plus collaboration with many different third-party software providers. Stackdriver has services for monitoring, logging, error reporting, fault tracing and debugging.
- You only pay for what you use. And there are free usage allotments, so that you can get started with no upfront fees or commitments.
- Now, in most other environments these services are handled by completely different packages, or by a loosely integrated collection of software. When you see these functions working together in a single, comprehensive, and integrated service, you'll realize how important that is. Stackdriver integration streamlines and unifies these traditionally independent services, making it much easier to establish procedures around them and to use them in continuous ways.
- To creating reliable, stable, and maintainable applications. Stackdriver also supports a rich and growing ecosystem of technology partners,

MONITORING

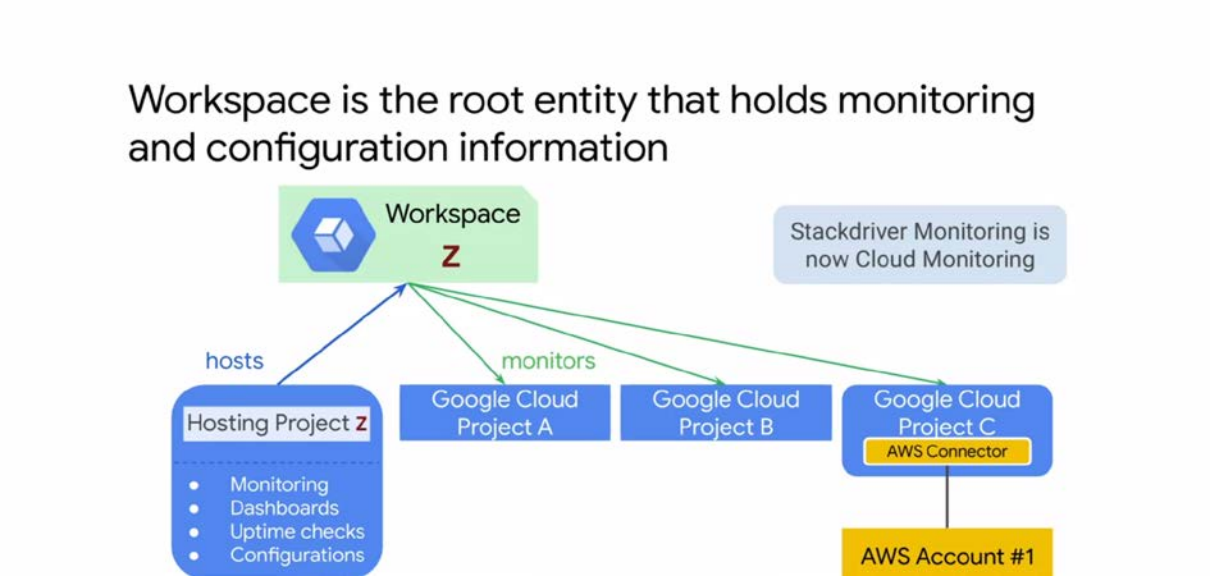
- Monitoring is important to Google because it is at the base of site reliability engineering, or SRE. SRE is a discipline that applies aspects of software engineering to operations whose goals are to create ultra scalable and highly reliable software systems. This discipline has enabled Google to build, deploy, monitor, and maintain some of the largest software systems in the world.



- Stackdriver dynamically configures monitoring after resources are deployed and has intelligent defaults that allow you to easily create charts for basic monitoring activities. This allows you to monitor your platform, system and application metrics by ingesting data, such as metrics, events, and metadata. You can then generate insights from this data through dashboards, charts, and alerts.

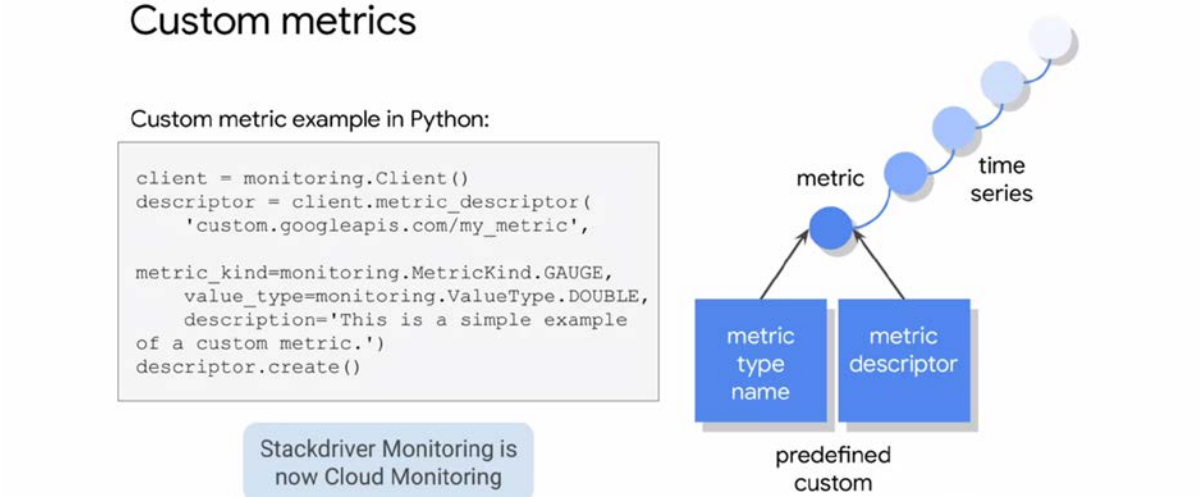


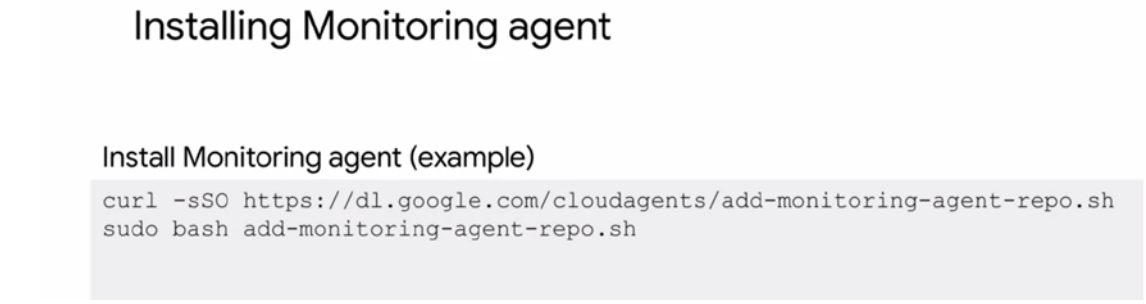
- For example, you can configure and measure uptime and health checks that send alerts via e-mail.
- A **workspace** is the root entity that holds monitoring and configuration information in Stackdriver monitoring. Each workspace can have between one and 100 monitored projects, including one or more GCP projects, and any number of AWS accounts. You can have as many workspaces as you want, but GCP projects and AWS accounts can be monitored by more than one workspace.



- A workspace contains the custom dashboards, alerting policies, uptime checks, notification channels, and group definitions that you use with your monitored projects.
- A workspace can access metric data from its monitored projects, but the measured data in log entries remain in the individual projects.
- The first monitor GCP project in a workspace is called the hosting project and it must be specified when you create the workspace. The name of that project becomes the name of your workspace. To access an AWS account, you must configure a project in GCP to hold the AWS connector.
- Because workspaces can monitor all of your GCP projects in a single place, a workspace is a single pane of glass through which you can view resources from multiple GCP projects and AWS accounts.
- All Stackdriver users who have access to that workspace have access to all data by default. This means that a Stackdriver role assigned to one person on one project applies equally to all projects monitored by that workspace.
- In order to give people different roles per project and to control visibility to data, consider placing the monitoring of those projects in separate workspaces.
- Stackdriver monitoring allows you to create custom dashboards that contain charts of the metrics that you want to monitor.
- For example, you can create charts that display your instances CPU utilization. The packets are bytes sent and received by those instances, and the packets are bytes dropped by the firewall of those instances. In other words, charts provide visibility into the utilization and network traffic of your VM instances, as shown on this slide.
- These charts can be customized with filters to remove noise, groups to reduce the number of time series and aggregates to group multiple time series together,
- Now, although charts are extremely useful, they can only provide insight when someone is looking at them.
- If not, you want to create **alerting policies** than notify you when specific conditions are met. When this condition is met, you or someone else can be automatically notified through e-mail, SMS, or other channels in order to troubleshoot this issue.
- You can also create an alerting policy that monitors your Stackdriver usage and alerts you when you approach the threshold for billing.
- I also recommend customizing your alerts to the audiences need by describing what actions need to be taken or what resources need to be examined.
- Finally, avoid noise because this will cause alerts to be dismissed overtime. Specifically adjust monitoring alerts so that they are actionable and don't just set up alerts on everything possible.
- **Uptime checks** can be configured to test the availability of your public services from locations around the world. The type of uptime check can be set to HTTP, HTTPS, or TCP.

- The resource to be checked can be an App Engine application, a Compute Engine instance, a URL of a host or an AWS instance, or load balancer.
- For each uptime check, you can create an alerting policy and view the latency of each global location.
- Uptime checks that do not get a response within this time out period are considered failures. So far there is 100 percent uptime with no outages.
- Stackdriver monitoring can access some metrics without the monitoring agent, including CPU utilization, some disk traffic metrics, network traffic and uptime information.
- However, to access additional system resources and application services, you should install the **monitoring agent.**
- The monitoring agent is supported for Compute Engine and EC2 instances. The monitoring agent can be installed with these two simple commands which you could include in your startup script.
- If the standard metrics provided by Stackdriver monitoring do not fit your needs, you can create custom metrics.
- For example, imagine a game server that has a capacity of 50 users. What metric indicator might you use to trigger Scaling events? From an infrastructure perspective, you might consider using CPU load or perhaps network traffic load as values that are somewhat correlated with the number of users. But with a custom metric, you could actually pass the current number of users directly from your application into Stackdriver.

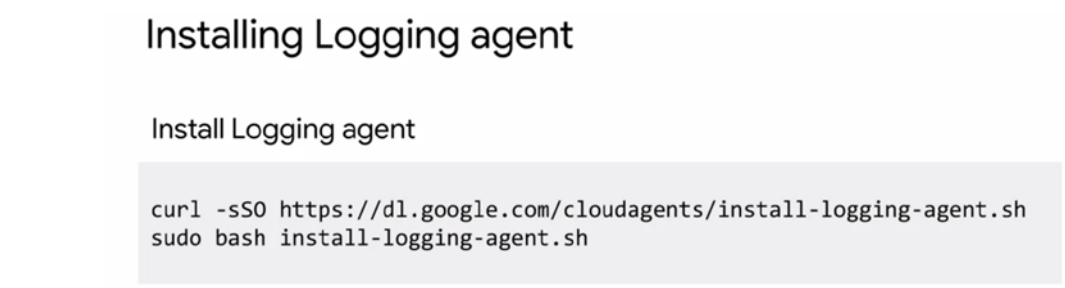




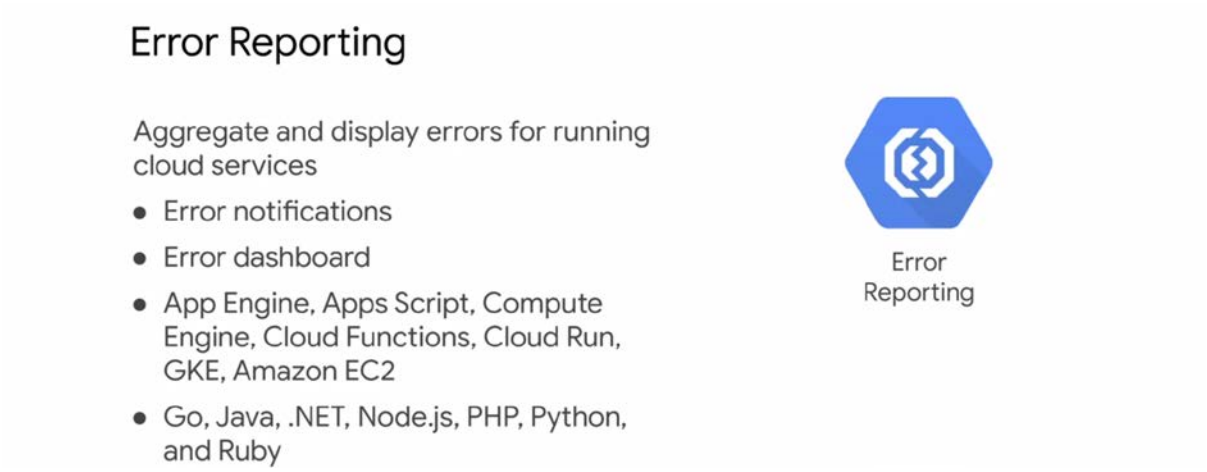
LOGGING

- Monitoring is the basis of Stackdriver, but the service also provides logging error reporting, tracing and debugging.
- **Stackdriver Logging** allows you to store search, analyze, monitor, and alert on log data and events from GCP an AWS. It is a fully managed service that performs at scale and can ingest application and system log data from thousands of EMS.
- Logging include storage for logs, a user interface called the log viewer, and an API to manage logs programmatically.

- The service let's you read and write log entries, search and filter your logs and create log based metrics.
- Logs are only retained for 30 days, but you can export your logs to **cloud storage buckets, BigQuery datasets** and **cloud Pub/Sub topics.**
- Exporting logs to **BigQuery** allows you to analyze logs and even visualize them in **data studio**. BigQuery runs extremely fast SQL queries on gigabytes to petabytes of data. This allows you to analyze logs such as your network traffic so that you can better understand traffic growth to forecast capacity.
- Network usage to optimize network traffic expenses, or network forensics to analyze incidents.
- I recommend connecting your BigQuery tables to data studio. Data Studio transforms your raw data into the metrics and dimensions that you can use to create easy to understand reports and dashboards.
- Exporting logs to cloud **Pub/Sub** - this enables you to stream logs to applications or end points. Similar to Stackdriver monitoring agent, it's a best practice to install the logging agent on all of your VM instances. The logging agent can be installed with these two simple commands which you could include in your startup script. This agent is supported for compute engine and EC2 instances.



ERROR REPORTING



TRACING

- Tracing is another Stackdriver feature integrated into GCP. Stackdriver Trace is a distributed tracing system that collects latency data from your applications and displays it in the GCP console.
- You can track how request propagate through your application and receive detailed near real time performance insights.
- Stackdriver Trace automatically analyzes all of your applications traces to generate in depth latency reports that surface performance degradations, and can capture traces from App Engine, HTTPS load balancers, and applications instrumented with the Stackdriver Trace API.

- Managing the amount of time it takes for your application to handle incoming requests and perform operations is an important part of managing overall application performance. Stackdriver Trace is actually based on the tools used at Google to keep our services running at extreme scale.

DEBUGGING

S

- Stackdriver debugger is a feature of GCP, that lets you inspect the state of a running application in real time without stopping or slowing it.
- Specifically, the debugger adds less than 10 milliseconds to the request latency when the application state is captured.
- In most cases, this is not noticeable by users. These features allow you to understand the behavior of your code in production and analyze its state to locate those hard to find bugs.
- With just a few mouse clicks, you can take a snapshot (capture call stack and local variables) of your running application state or inject a new logging statement.
- Stackdriver debugger supports multiple languages, including Java, Python, Go, Node.js, and Ruby.

CLOUD OPERATIONS

- Faster problem resolution
- Multi-cloud monitoring
- Reduces monitoring overhead

NOTE : Which service requires a logging agent installed to collect and send logs to Cloud Operations? Ans : GCE

VPN

- Cloud VPN securely connects your on-premise network to your GCP VPC network through an IPSec VPN tunnel.

- NOTE : Internet Protocol Security (IPSec) is a secure network protocol suite that authenticates and encrypts the packets of data to provide secure encrypted communication between two computers over an Internet Protocol network.

- Traffic traveling between the two networks is encrypted by one VPN gateway. Then decrypted by the other VPN gateway. This protects your data as it travels over the public internet. That's why Cloud VPN is useful for low volume data connections.

- As a managed service Cloud VPN provides an SLA of 99.9 percent service availability and supports

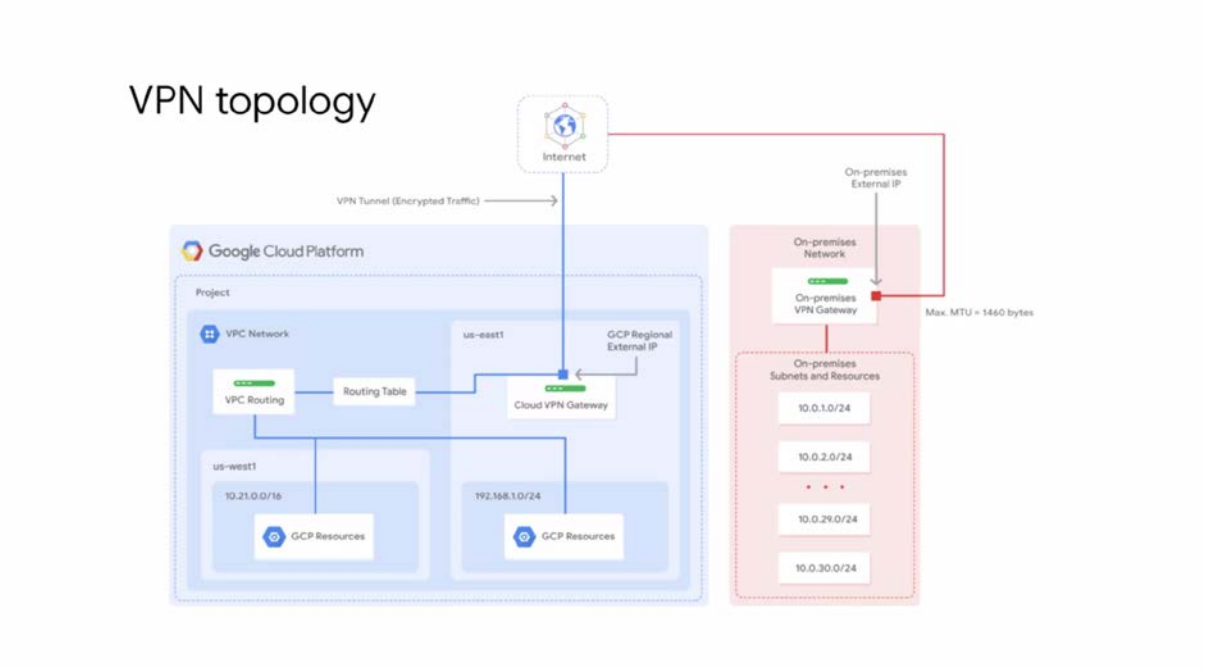
    - site to site VPN static and

    - dynamic routes,

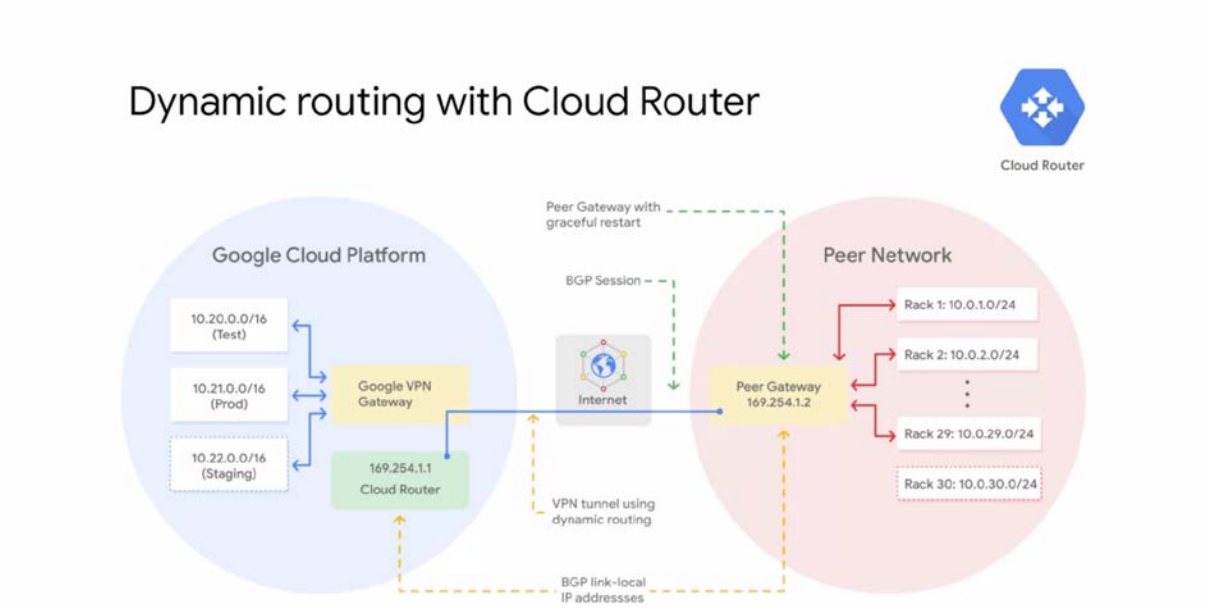
    - and IKEv1 and IKEv2 ciphers.

- Cloud VPN doesn't support new cases where a client computers need to dial in to a VPN using client VPN software.

- Also, dynamic routes are configured with Cloud Router

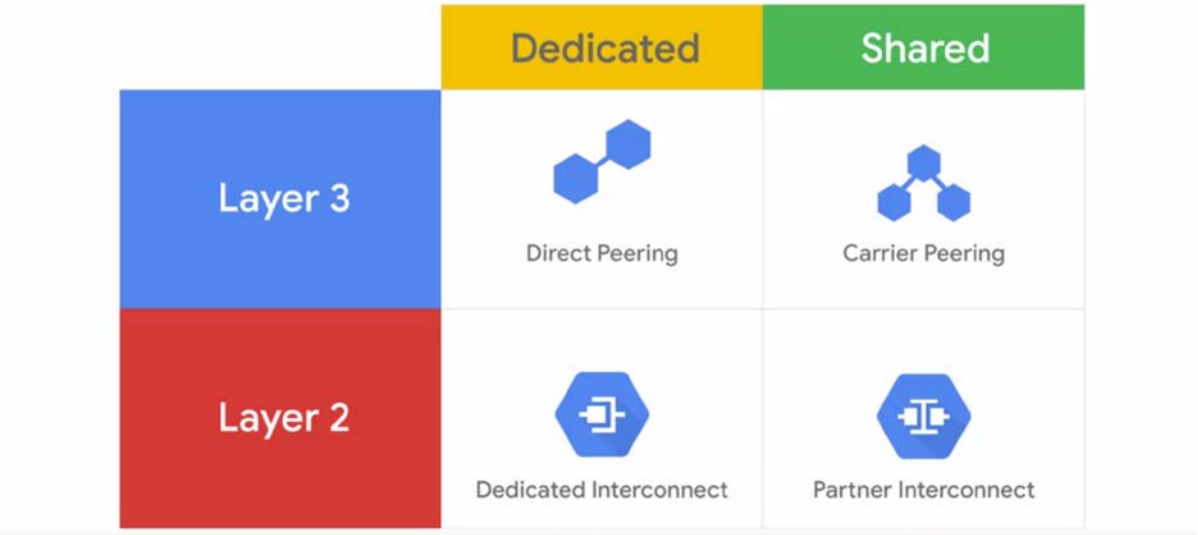


- This diagram shows a simple VPN connection between your VPC and on-premise network.
- Your VPC network has subnets in US-east one and US-west one. With GCP resources in each of those regions. These resources are able to communicate using their internal IP addresses because routing within a network is automatically configured, assuming that firewall rules allow the communication.
- Now, in order to connect to your on-premise network and its resources you need to configure your Cloud VPN gateway on-premise VPN gateway and to VPN tunnels.
- The Cloud VPN gateway is a regional resource that uses a regional external IP address. Your on-premise VPN gateway can be a physical device in your data center or a physical or software based VPN offering in another Cloud providers network.
- This VPN gateway also has an external IP address. A VPN tunnel then connects your VPN gateways and serves as the virtual medium through which encrypted traffic is passed.
- In order to create a connection between two VP gateways you must establish two VPN tunnels.
- Each tunnel defines the connection from the perspective of its gateway and traffic can only pass when the pair of tunnels established.
- When using Cloud VPN is that the maximum transmission unit or MTU for your on-premises VPN gateway cannot be greater than 1,460 bytes. This is because of the encryption and encapsulation of packets.



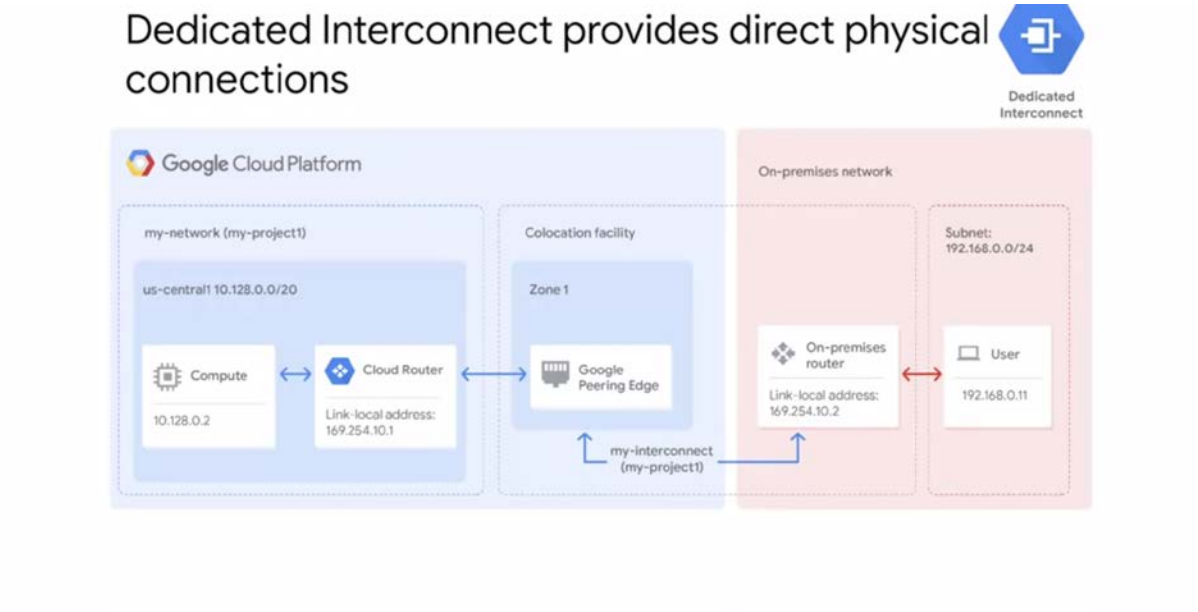
- Cloud VPN supports both static and dynamic routes. In order to use dynamic routes you need to configure Cloud Router. Cloud Router can manage routes from Cloud VPN tunnel using **border gateway protocol** or BGP.
- This routing method allows for routes to be updated and exchanged without changing the tunnel configuration.
- For example, this diagram shows two different regional subnets in a VPC network namely tests and prod.
- The on-premise network has 29 subnets and the two networks are connected through Cloud VPN tunnels.
- Now, how would you handle adding new subnets? For example, how would you add a new staging subnet in the GCP network and a new on-premise 10.0.30.0/24 subnet to handle growing traffic in your data center?
- To automatically propagate network configuration changes the VPN tunnel uses Cloud Router to
- establish a BGP session between the VPC and the on-premise VPN gateway which must support BGP.
- The new subnets are then seamlessly advertised between networks.
- This means that instances in the new subnets can start sending and receiving traffic immediately as
- To set up BGP an additional IP address has to be assigned to each end of the VPN tunnel. These two IP addresses must be link-local IP addresses. Belonging to the IP address range 169.254.0.0/16. These addresses are not part of IP address space of either network and are used exclusively for establishing a BGP session.

CLOUD INTERCONNECT AND PEERING

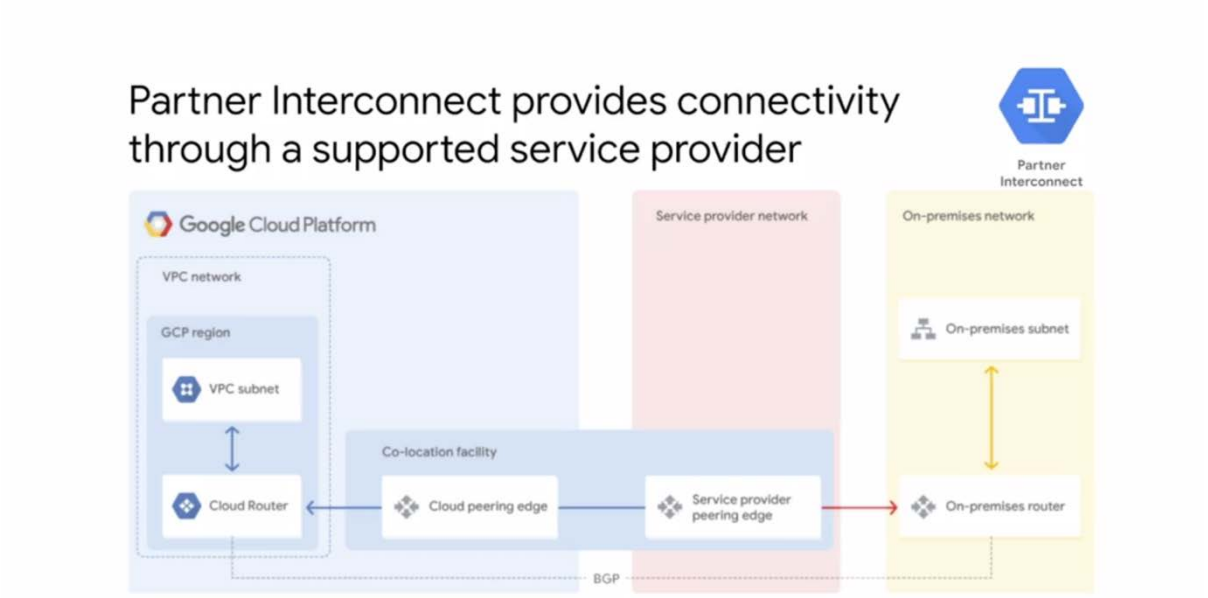


- There are different Cloud Interconnect and Peering services available to connect your infrastructure to Google's network.
- These services can be split into dedicated versus shared connections and layer two verses layer three connections.
- The services are **Direct Peering, Carrier Peering, Dedicated Interconnect, and Partner Interconnect.**
- **Dedicated connections** provide a direct connection to Google's network.
- **Shared connections** provide a connection to Google's network through a partner.
- **Layer two** connections use a VLAN that pipes directly into your GCP environment, providing connectivity to internal IP addresses in the RFC 1918 address space.
- **Layer three** connections provide access to G Suite services, YouTube and Google Cloud APIs using public IP addresses.
- Google also offers its own Virtual Private Network service called Cloud VPN. This service uses the public Internet but traffic is encrypted and provides access to internal IP addresses. That's why Cloud VPN is a useful addition to Direct Peering and Carrier Peering.
- NOTE : A **VLAN** allows a network of computers and users to communicate in a simulated environment as if they exist in a single LAN and are sharing a single broadcast and multicast domain.

DEDICATED AND PARTNER INTERCONNECT



- **Dedicated Interconnect** provides direct physical connections between your On-premise network and Google's network. This enables you to transfer a large amount of data between networks which can't be more cost-effective than purchasing additional bandwidth over the public Internet.
- In order to use Dedicated Interconnect, you need to provision a cross-connect between the Google network and your own router in a common co-location facility.
- To exchange routes between the networks, you configure a BGP session over the interconnect between the Cloud Router and the On-premise router. This will allow user traffic from the on-premise network to reach GCP resources on the VPC network and vice-versa.
- Dedicated Interconnect can be configured to offer a 99.9 percent or a 99.99 percent uptime SLA.
- In order to use Dedicated Interconnect, your network must physically meet Google's network in a supported co-location facility.



- **Partner Interconnect** provides connectivity between your on-premise network and your VPC network through a supported service provider. This is useful if your data center is in the physical location that cannot reach a Dedicated Interconnect co-location facility or if your data needs don't warrant a Dedicated Interconnect.

- In order to use Partner Interconnect, you work with the supported service provider to connect your VPC and on-premise networks

- These service providers have existing physical connections to Google's network that they make

- available for their customers to use. After you establish connectivity with the service provider,

- you can request a Partner Interconnect connection from your service provider then establish a BGP session between your Cloud Router and On-premise Router to start passing traffic between your networks via the service providers network.

- Partner Interconnect can be configured to offer a 99.9 percent or 99.99 percent uptime SLA between Google and the service provider.

Comparison of Interconnect options

Connection	Provides	Capacity	Requirements	Access Type
IPsec VPN tunnel	Encrypted tunnel to VPC networks through the public internet	1.5-3 Gbps per tunnel	On-premises VPN gateway	Internal IP addresses
Dedicated Interconnect	Dedicated, direct connection to VPC networks	10 Gbps per link 100 Gbps BETA	Connection in colocation facility	
Partner Interconnect	Dedicated bandwidth, connection to VPC network through a service provider	50 Mbps – 10 Gbps per connection	Service provider	

- All of these options provide internal IP address access between resources in your On-premise network and in your VPC network. The main differences are the connection capacity and
- the requirements for using a service.
- The **IPSec VPN tunnels** that Cloud VPN offers have a capacity of 1.5 to 3 Gbps per tunnel and require VPN device on your On-premise network. The 1.5 Gbps capacity applies to the traffic that traverses the public Internet and the three Gbps capacity applies to the traffic that is traversing a Direct Peering link. You can configure multiple tunnels if you want to scale this capacity.
- **Dedicated Interconnect** has a capacity of 10 Gbps per link and requires you to have a connection in a Google supported co-location facility. You can have up to eight links to achieve multiples of
- 10 Gbps by 10 Gbps is the minimum capacity.
- **Partner Interconnect** has a capacity of 50 Mbps to 10 Gbps per connection and requirements depend on the service provider.
- My recommendation is to start with VPN tunnels. When you need enterprise-grade connection to GCP, switch to Dedicated Interconnect
- or Partner Interconnect depending on your proximity to a co-location facility and your capacity requirements.

DIRECT AND CARRIER PEERING

- These services are useful when you require access to Google and Google Cloud properties.

- Google allows you to establish a **direct peering** connection between your business network and Google's.
- With this connection, you will be able to exchange internet traffic between your network
- and Google's at one of the Google's broad-reaching Edge network locations.
- Direct Peering with Google is done by exchanging BGP routes between Google and the peering entity. After a direct peering connection is in place, you can use it to reach all the Google's services including the full suite of Google Cloud Platform products. Unlike dedicated interconnect, direct peering does not have an SLA. In order to use direct peering, you need to satisfy the peering requirements.
- GCP's Edge points of presence or PoPs are where Google's network connects to the rest of the internet via peering. PoPs are present on over 90 internet exchanges and at over 100 interconnection facilities around the world.
- If you require access to Google public infrastructure and cannot satisfy Google's peering requirements, you can connect via a **carrier peering** partner. Work directly with your service provider to get the connection you need and to understand the partners requirements.
- Now, just like direct peering, carrier peering also does not have an SLA.

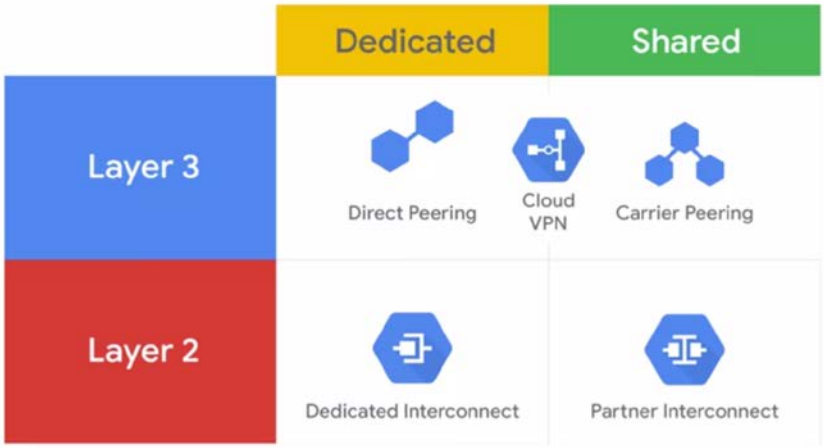
Comparison of Peering options

Connection	Provides	Capacity	Requirements	Access Type
Direct Peering	Dedicated, direct connection to Google's network	10 Gbps Per link	Connection in GCP PoPs	Public IP addresses
Carrier Peering	Peering through service provider to Google's public network	Varies based on partner offering	Service provider	

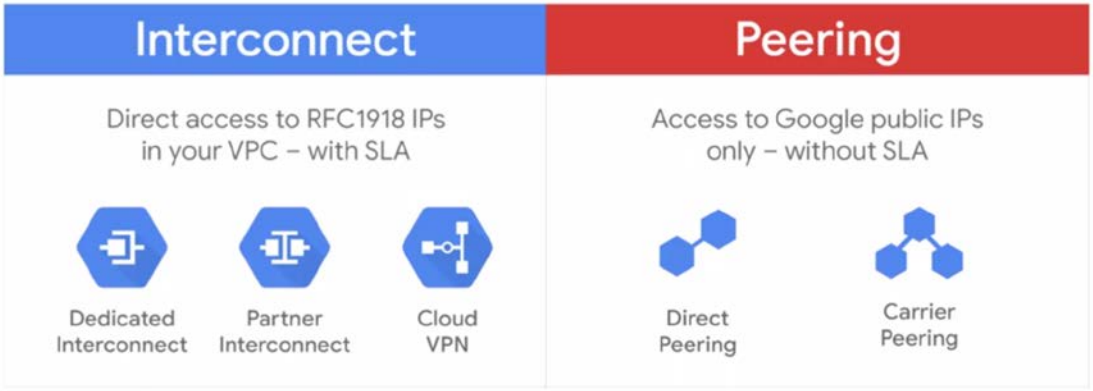
- All of these options provide public IP address access to all of Google's services. The main differences are capacity and the requirements for using a service.
- Direct peering has a capacity of 10Gbps per link and requires you to have a connection
- in a GCP Edge point of presence.
- Carrier peering's capacity and requirements vary depending on the service provider that you work with.

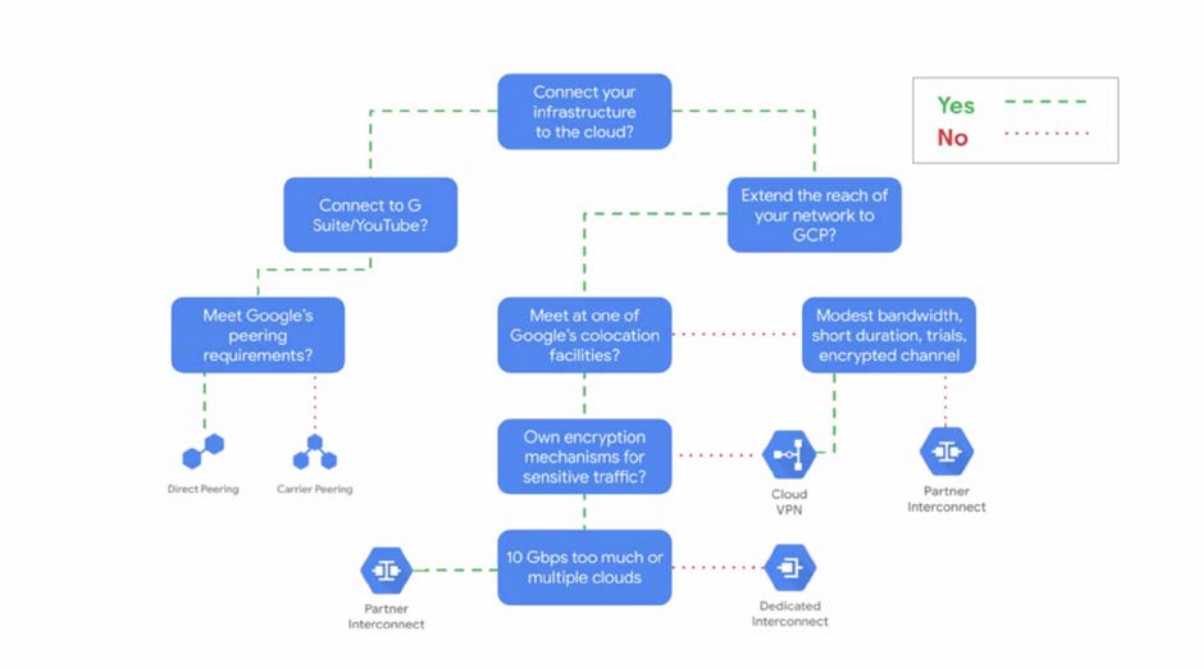
CHOOSING A CONNECTION

5 ways to connect your infrastructure to GCP



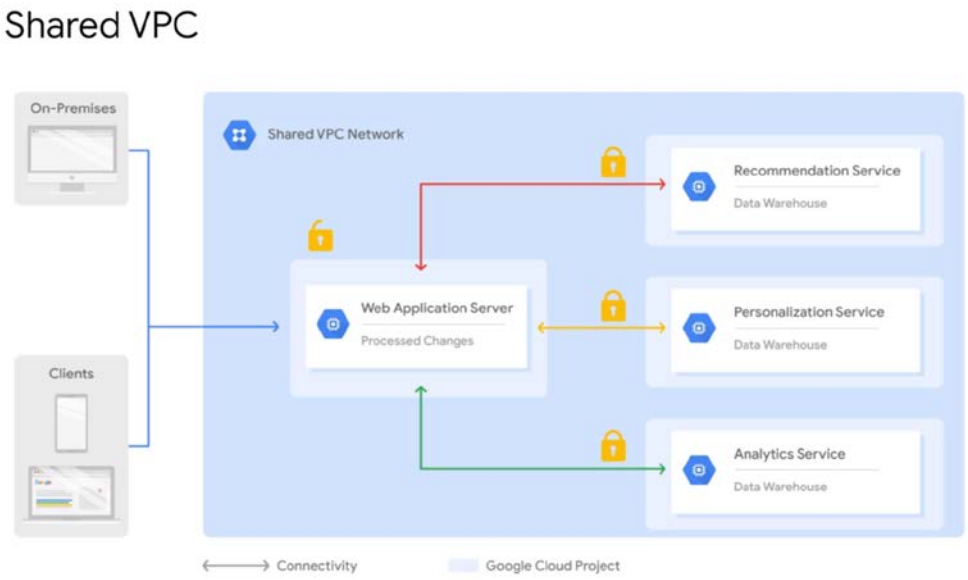
Choosing a network connection option



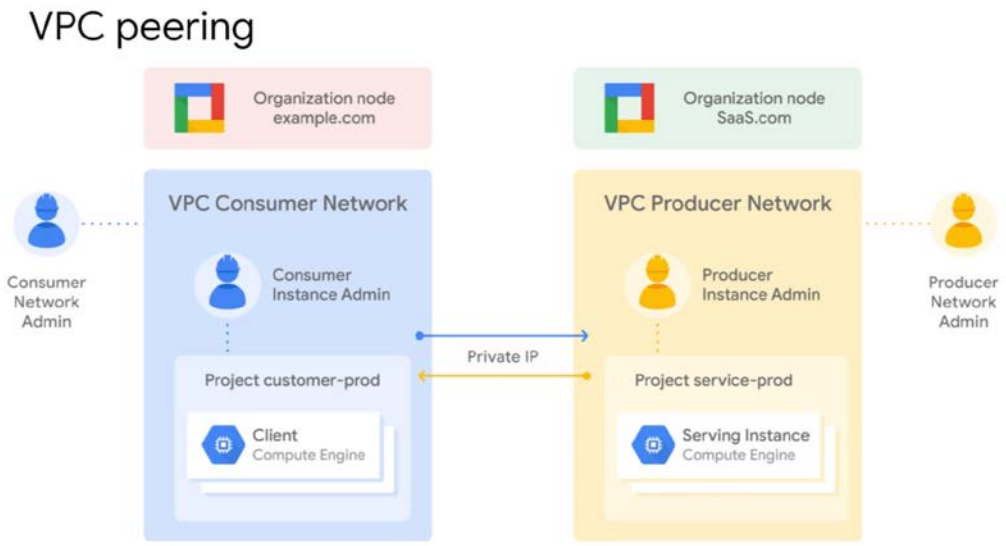


SHARED VPC AND VPC (NETWORK) PEERING

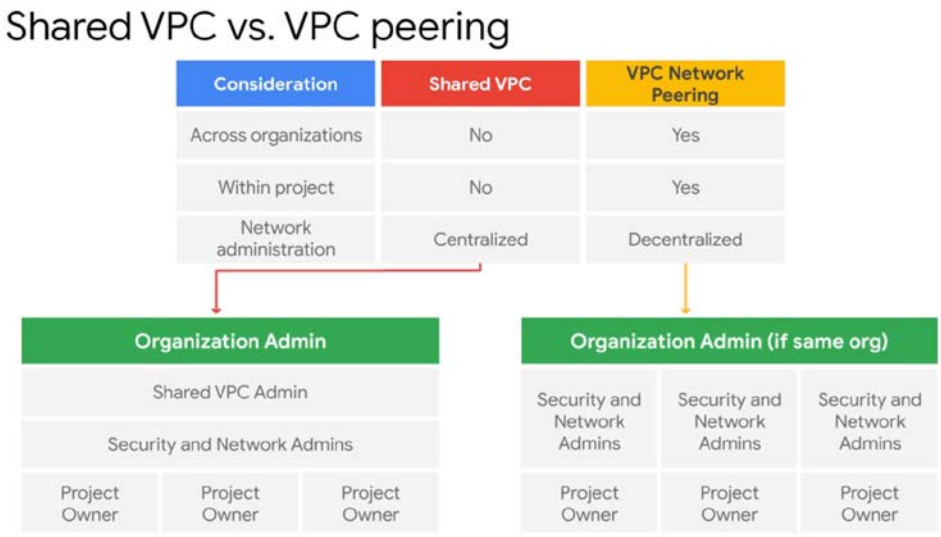
- Let's move our attention from hybrid connectivity to shared VPC networks.
- In the simplest Cloud environment, a single project might have one VPC network, spanning many regions with VM instances hosting very large and complicated applications. However, many organizations commonly deploy multiple isolated projects with multiple VPC networks and sublets. There are two configurations for sharing VPC networks across GCP projects.
    - **Shared VPC** which allows you to share a network across several projects in your GCP organization.
    - **VPC Network Peering** which allows you to configure private communication across projects
    - in same or different organizations.
- Shared VPC allows an organization to connect resources from multiple projects to a common VPC network. This allows the resources to communicate with each other securely, and efficiently using internal IPs from that network.



- For example, in this diagram, there is one network that belongs to the web application servers project. This network is shared with three other projects. Namely, the recommendation service,
- the personalization service, and the analytics service.
- Each of those service projects has instances that are in the same network as the web application server, and allow for private communication to that server using the internal IP addresses.
- The web application server communicates with clients and on-premises, using the server's external IP address. The backend services in contrast can not be reached externally because they only communicate using internal IP addresses.
- When you shared VPC, you designate a project as a **host project**, and attach one or more other service projects to it. In this case, the web application servers project is the host project. The three other projects are the **service projects**.
- The overall VPC network is called the shared VPC network.

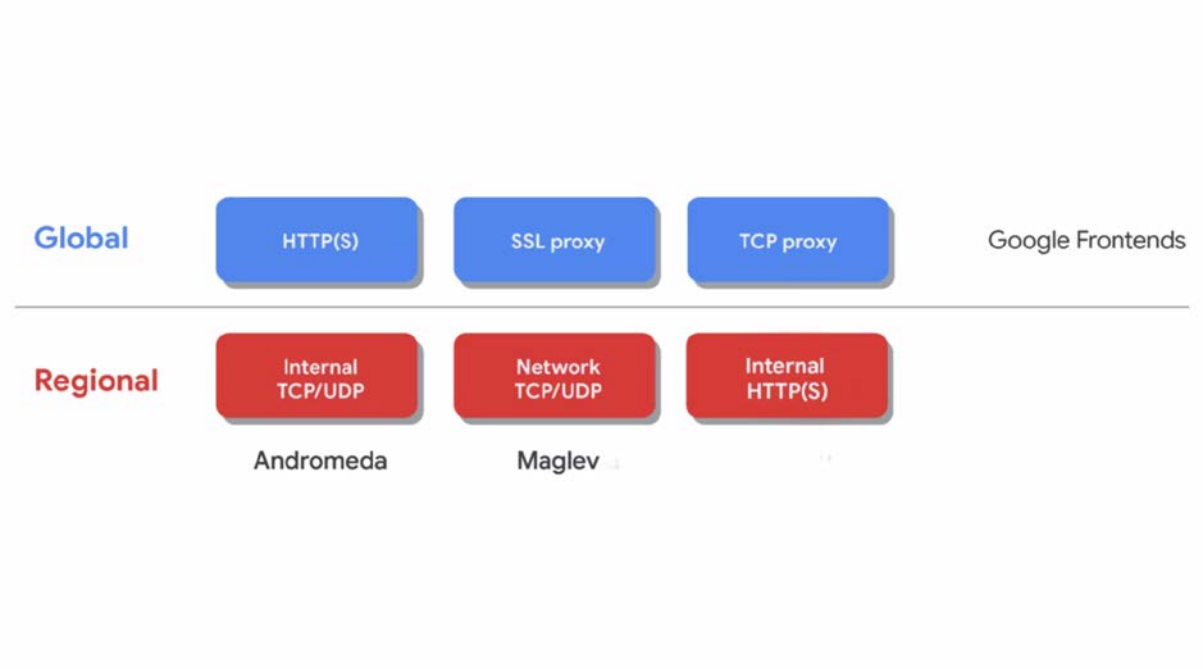


- VPC Network Peering in contrast, allows private RFC 1918 connectivity across two VPC networks, regardless of whether they belong to the same project, or the same organization
- Each VPC network will have firewall rules that define what traffic is allowed or denied between the networks.
- For example, in this diagram, there are two organizations that represent a consumer and a producer respectively.
- Each organization has its own organization node. VPC network, virtual machine instances, network admin, and instance admin.
- In order for VPC Network Peering to be established successfully, the producer network admin needs to peer the producer network with the consumer network.
- The consumer network admin needs to peer the consumer network with the producer network.
- When both peering connections are created, the VPC Network Peering session becomes active and routes are exchanged.
- This allows the virtual machine instances to communicate privately using their internal IP addresses. VPC Network Peering is a decentralized or distributed approach to multiproject networking.
- Because each VPC network, may remain under the control of separate administrator groups, and maintains its own global firewall, and routing tables.
- Historically, such projects would consider external IP addresses or VPNs to facilitate private communication between VPC networks. However, VPC Network Peering does not incur the network latency, security, and cost drawbacks that are present when using external IP addresses or VPNs.



- If you want to configure a private communication between VPC networks in different organizations, you have to use VPC Network Peering. Shared VPC only works within the same organization.
- Somewhat similarly, if you want to configure private communication between VPC networks in the same project, you have to use VPC Network Peering. This doesn't mean that the networks need to be in the same project, but they can be. Shared VPC, only works across projects.
- The biggest difference between the two configurations is the network administration models.
- Shared VPC is a **centralized approach** to multi-project networking because security and network policy occurs in a single designated VPC network.
- In contrast, VPC Network Peering is a **decentralized approach** because each VPC network can remain under the control of separate administrator groups, and maintains its own global firewall, and routing tables.

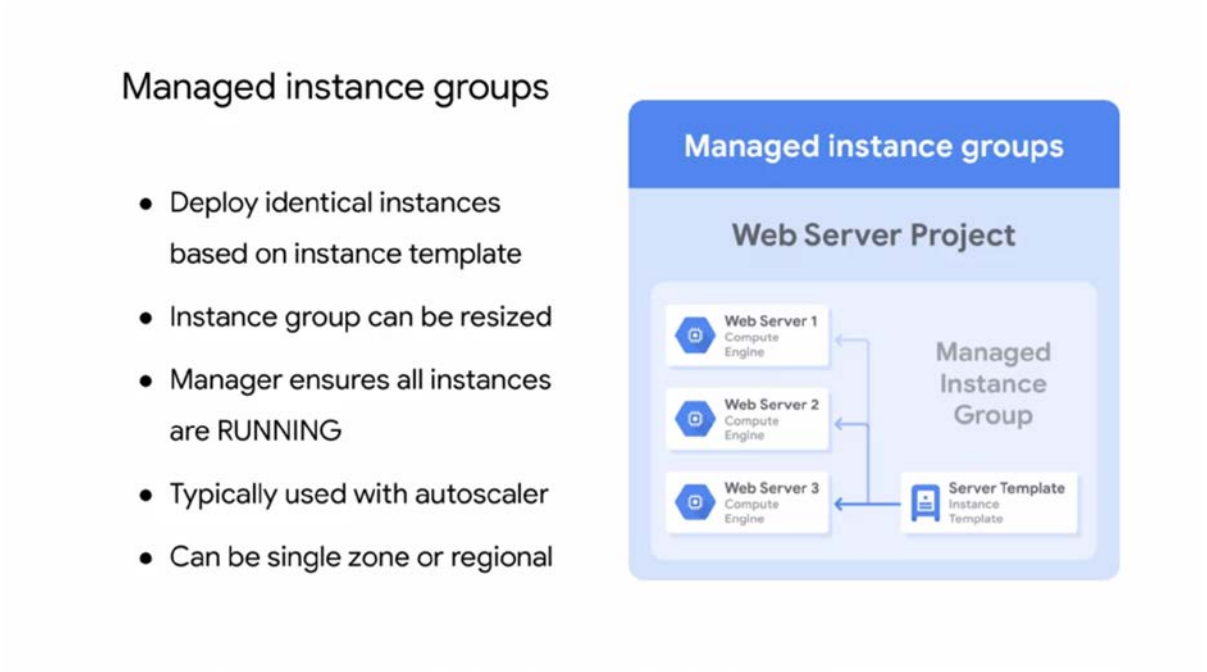
OVERVIEW OF LOAD BALANCING



- Cloud Load Balancing gives you the ability to distribute load balanced computer resources in single or multiple regions to meet your high availability requirements, to put your resources behind a single anycast IP address, and to scale your resources up or down with intelligent autoscaling.
- Using Cloud Load Balancing, you can serve content as close as possible to your users on a system that can respond to over one million queries per second.
- Cloud Load Balancing is a fully distributed software defined managed service. It is not instance or device based so you do not need to manage a physical load balancing infrastructure.
- GCP offers different types of load balancers that can be divided into two categories, **global** and **regional**.
- The global load balancers are the HTTP, HTTPS, SSL proxy, and TCP proxy load balancers.
- These load balancers leverage the Google front ends which are software defined, distributed systems that sit in Google's Point-of-Presence and are distributed globally.
- Therefore, you want to use a global load balancer when your users and instances are globally distributed. Your users need access to the same application and content and you want to provide access using a single anycast IP address.
- The regional load balancers are the internal and network load balancers and they distribute traffic to instances that are in a single GCP region.
- The internal load balancer uses **Andromeda** which is GCP's software defined network virtualization stack and the network load balancer uses **Maglev** which is a large distributed software system.
- There's also another internal load balancer for HTTP, HTTPS traffic. The sixth load balancer is a proxy based regional layer seventh load balancer that enables you to run and scale your

services behind a private load balancing IP address that is accessible only in the load balancers' region in your VPC network.

MANAGED INSTANCE GROUPS

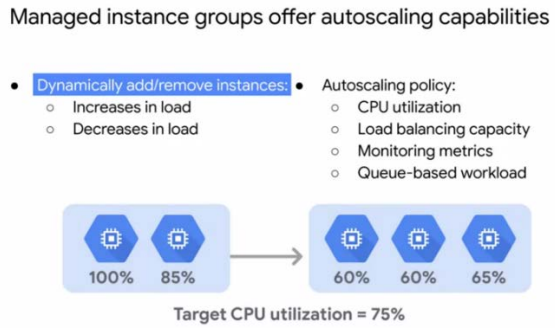


- A managed instance group is a collection of identical virtual machine instances that you control as a single entity using an instance template.
- You can easily update all the instances in the group by specifying a new template in a rolling update. Also, when your application requires additional compute resources, managed instance groups can automatically scale the number of instances in the group.
- Managed instance groups can work with load balancing services to distribute network traffic to old instances in the group.
- If an instance in the groups stops, crashes, or is deleted by an action
- other than the instance groups command, the managed instance group automatically recreates
- the instance so it can resume its processing tasks. The recreated instance uses the same name and the same instance template as the previous instance.
- Managed instance groups can automatically identify and recreate unhealthy instances in a group to ensure that all the instances are running optimally.
- Regional managed instance groups are generally recommended over zonal managed instance groups because they allow you to spread the application load across

multiple zones instead of confining your application to a single zone or you're having to manage multiple instance groups across different zones.

- This replication protects against zonal failures and unforeseen scenarios where an entire group of instances in a single zone malfunctions. If that happens, your application can continue serving traffic from instances running in another zone of the same region.
- In order to create a managed instance group, you first need to create an instance template. Next, you are going to create a managed instance group of end specific instances.
- The instance group manager then automatically populates the instance group based on the instance template.
- You can easily create instance templates using the GCP console.
- The instance template dialogue looks and works exactly like creating an instance, except that the choices are recorded so that they can be repeated.
- When you create an instance group, you define the specific rules for the instance group.
    - First, decide whether the instance group is going to be single or multi zoned and where those locations will be.
    - Second, choose the ports that you are going to allow and load balance across.
    - Third, select the instance template that you want to use.
    - Fourth, decide whether you want to auto-scale and under what circumstances.
    - Finally, consider creating a health check to determine which instances are healthy and should receive traffic.
- Essentially, you're still creating virtual machines, but you're applying more rules to that instance group.

AUTO SCALING AND HEALTH CHECKS



- As I mentioned earlier, managed instance groups offer **autoscaling capabilities** that allow you to automatically add or remove instances from a managed instance group based on increase or decrease in load.

- Autoscaling helps your applications gracefully handle increase in traffic and reduces cost when the need for resource is lower.

- You just define the autoscaling policy, and the autoscaler performs automatic scaling based on the measured load.

- Applicable autoscaling policies include scaling based **on CPU utilization, load balancing capacity, or monitoring metrics, or by a queue-based workload like Cloud Pub/Sub.**

- For example, let's assume you have two instances that are at 100 percent and 85 percent CPU utilization as shown.

- If your target CPU utilization is 75 percent, the autoscaler will add another instance to spread out

- the CPU load and stay below the 75 percent target CPU utilization.

- Similarly, if the overall load is much lower than the target, the autoscaler will remove instances as long as that keeps the overall utilization below the target.

- When you click on an instance group or even an individual virtual machine, a graph is presented. By default, you will see the CPU utilization over the past hour.

- But you can change the timeframe and visualize other metrics like disk and network usage. These graphs are very useful for monitoring your instances, utilization, and for determining how best to configure your autoscaling policy to meet changing demands.

- If you monitor the utilization of your virtual machine instances and Stackdriver monitoring,

- you can even set up alerts through several notification channels.

- Another important configuration for a managed instance group and load balancer is a **health check.**

- A health check is very similar to an Uptime check in Stackdriver. You just define a protocol, port,

- and health criteria as shown in the screenshot. Based on this configuration, GCP computes a health state for each instance.

    - The health criteria defines how often to check whether an instance is healthy. That's the check interval.

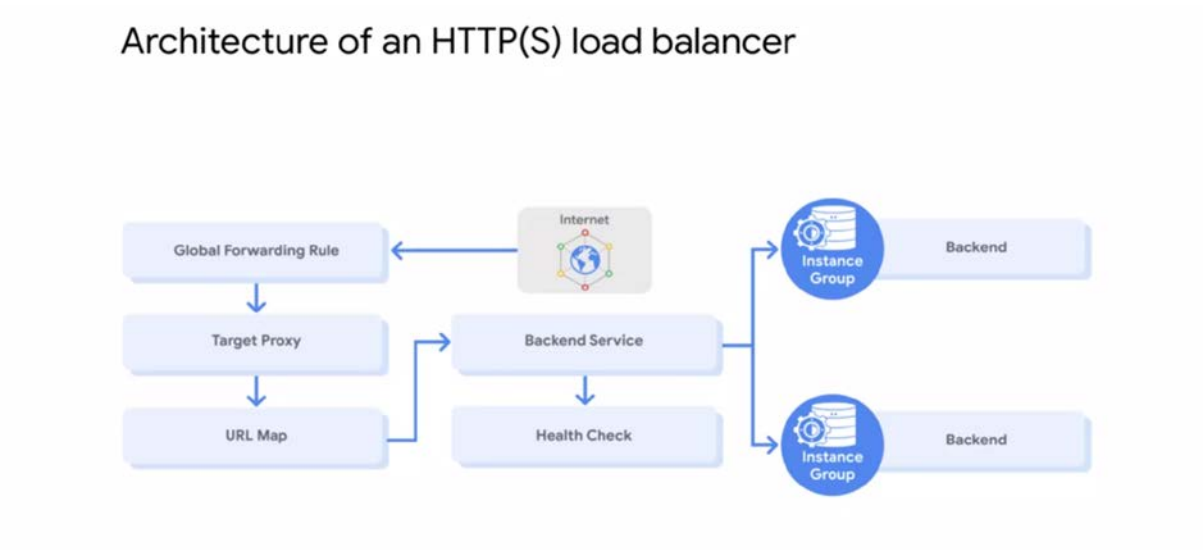
    - How long to wait for a response? That's the timeout.

    - How many successful attempts are decisive? That's the healthy threshold.

    - How many failed attempts are decisive? That the unhealthy threshold.

HTTP LOAD BALANCER

- Now, let's talk about HTTPS Load Balancing which acts at layer seven of the OSI model.
- This is the application layer which deals with the actual content of each message allowing for routing decisions based on the URL.
- GCP HTTPS load balancing provides global load balancing for HTTPS requests destined for your instances. This means that your applications are available to your customers at a single any cast IP address, which simplifies your DNS setup.
- HTTPS load balancing balances HTTP and HTTPS traffic across multiple backend instances and across multiple regions. HTTP requests are load balanced on port 80 or 8080, and HTTPS requests are load balanced on port 443.
- This load balancer supports both IPv4 and IPv6 clients, is scalable, requires no pre-warming, and enables content-based and cross-regional load balancing.
- You can configure your own **URL maps** that route some URLs to one set of instances and route other URLs to other instances.
- Requests are generally routed to the instance group that is closest to the user.
- If the closest instance group does not have sufficient capacity, the request is sent to the next closest instance group that does have the capacity.

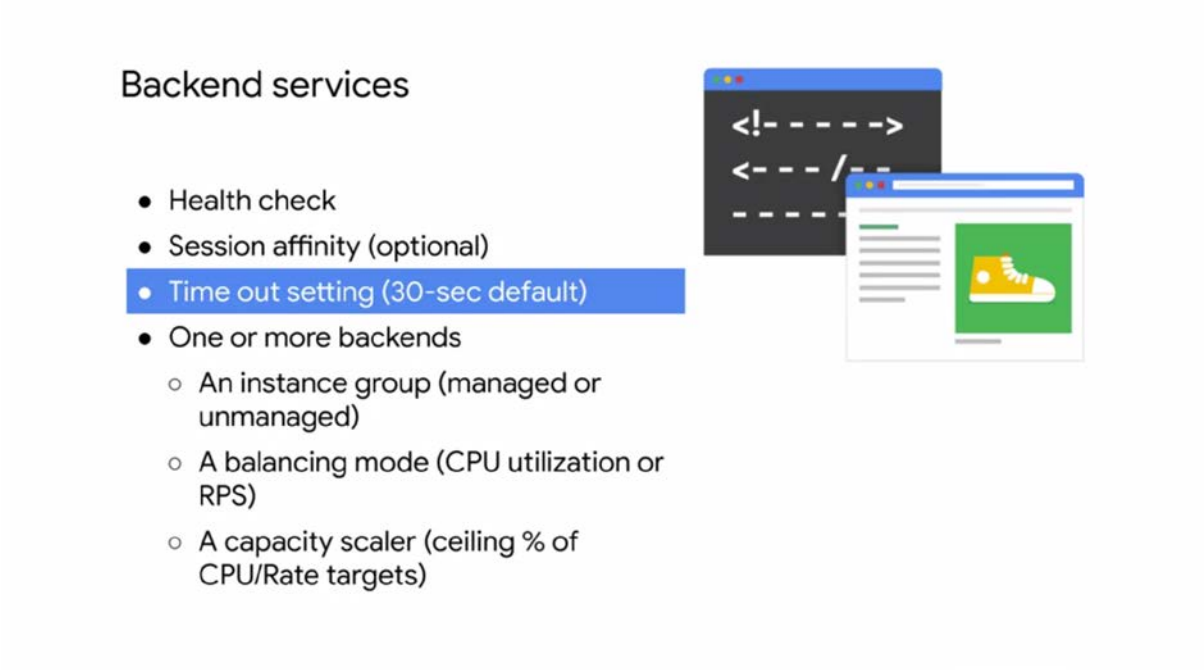


.

- A Global Forwarding Rule direct incoming requests from the Internet to a target HTTP proxy.
- The target HTTP proxy checks each request against a URL map to determine the appropriate backend service for the request.

- For example, you can send requests for www.example.com slash audio to one backend service, which contains instances configured to deliver audio files, and the request for www.example.com slash video to another backend service which contains instances configured to deliver video files.

- The backend service directs each request to an appropriate backend based on solving capacity zone and instance held of its attached backends.



- The backend services contain a health check, session affinity, a timeout setting, and one or more backends.

    - A **health check** pulls instances attached to the backend service at configured intervals. Instances that pass the health check are allowed to receive new requests. Unhealthy instances are not sent requests until they are healthy again.

      Normally, HTTPS load balancing uses a round robin algorithm to distribute requests among available instances. This can be overridden with session affinity.

    - **Session affinity** attempts to send all requests from the same client to the same Virtual Machine Instance.

    - Backend services also have a **timeout setting**, which is set to 30 seconds by default.

      This is the amount of time the backend service will wait on the backend before considering the request a failure. This is a fixed timeout not an idle timeout. If you require longer lived connections, set this value appropriately.

- The backends themselves contain an **instance group**, a **balancing mode**, and a **capacity scalar.**

    - An instance group contains Virtual Machine Instances. The instance group may be a managed instance group with or without autoscaling or an unmanaged instance group. A balancing mode tells the load balancing system how to determine when the backend is at full usage.

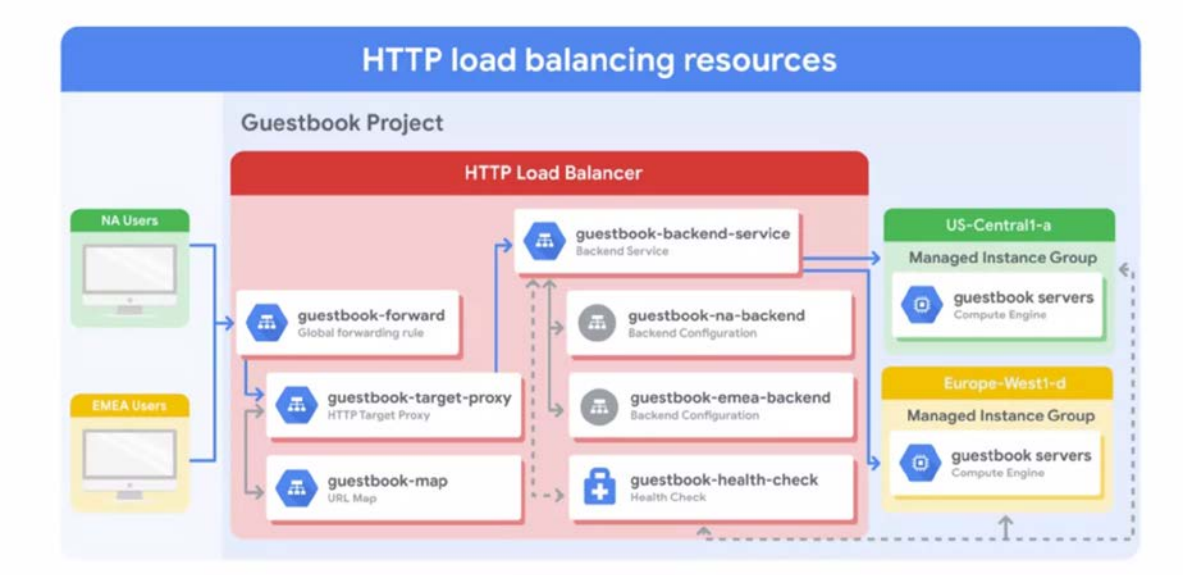
If older backends for the backend service in a region are at the full usage, new requests are automatically routed to the nearest region that can still handle requests.

  - The balancing mode can be based on CPU utilization or requests per second. A capacity setting is an additional control that interacts with the balancing mode setting.

  - For example, if you normally want your instances to operate at a maximum of 80 percent CPU utilization, you would set your balancing mode to 80 percent CPU utilization and your capacity to 100 percent. If you want to cut instance utilization in half, you could leave the balancing mode at 80 percent CPU utilization and set capacity to 50 percent.

- Now, any changes to your backend services are not instantaneous.

EXAMPLE HTTP LOAD BALANCER



- Let me work through an HTTP load balancer in action. The project on this slide has single global IP address, but users enter the Google Cloud network from two different locations, one in North America, and one in EMEA.

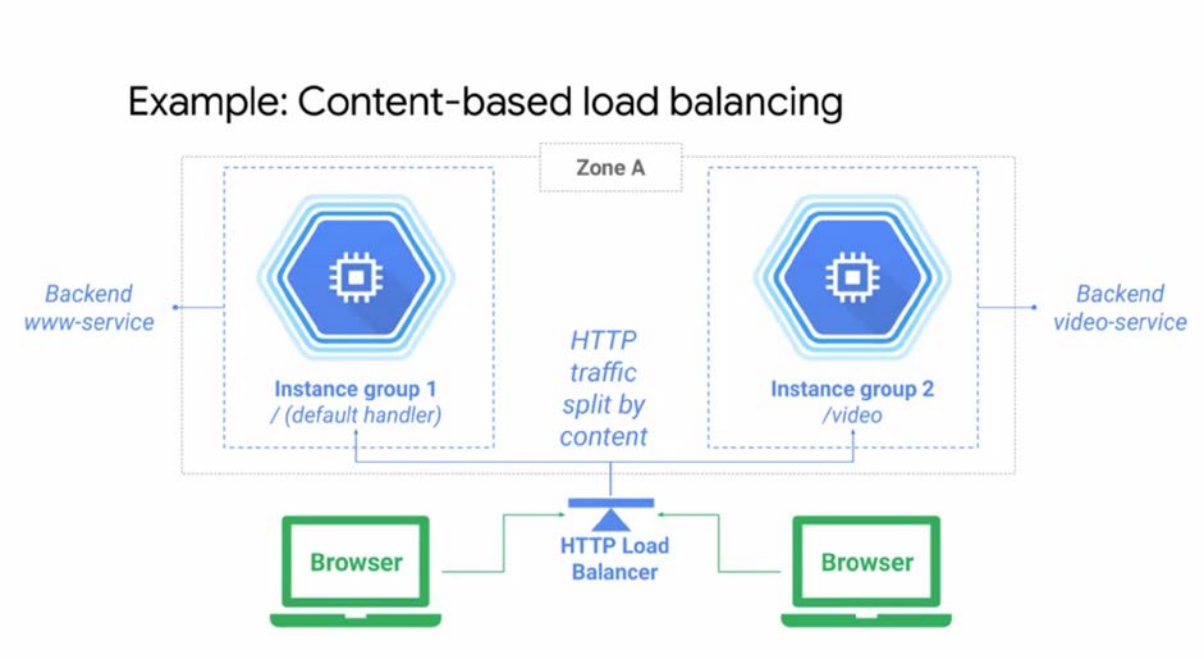
- First, the global forwarding rule directs incoming requests to the target HTTP proxy. The proxy checks the URL map to determine the appropriate back-end service for the request.

- In this case, we're serving a guestbook application with only one back-end service. The back-end service has two back-ends, one in US Central 1-a, and one in Europe West 1-d.

- Each of those back-ends consist of a managed instance group. Now, when a user request comes in, the load balancing service determines the approximate origin of the request from the source IP address.

- The load balancing service also knows the locations of the instances owned by the back-end service, their overall capacity and their overall current usage.

- Therefore, if the instances closest to the user has available capacity, the request is forwarded to that closest set of instances.
- In our example, traffic from the user in North America, would be forwarded to the managed instance group in US Central 1-a, and the traffic from the user in EMEA would be forwarded to the managed instance group in Europe West 1-d.
- If there are several users in each region, the incoming requests to the given region are distributed evenly across all available back-end services, and instances in that region.
- If there are no healthy instances, with available capacity in a given region, the load balancer instead sends the request to the next closest region with available capacity. Therefore, traffic from the EMEA user, could be forwarded to the US Central 1-a back-end, if the Europe West 1-d back-end does not have capacity, or has no healthy instances as determined by the health checker.
- This is referred to as **cross-region load balancing.**



- Another example, of an HTTPS load balancer, is a **content-based load balancer.**
- In this case there are two separate back-end services that handle either web or video traffic.
- The traffic is split by the load balancer based on the URL header as specified in the URL map, if the user's navigating to slash video, the traffic is sent to the back-end video service and if the user is navigating anywhere else, the traffic is sent to the web service back-end.
- All of that is achieved with a single global IP address.

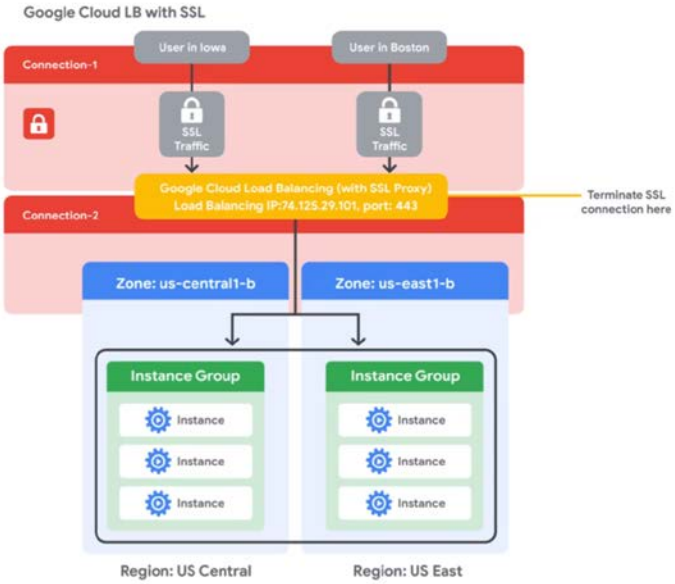
HTTPS LOAD BALANCER

- An HTTPS load balancer has the same basic structure as the HTTP load balancer, but differs in the following ways.
- An HTTPS load balancer uses a target HTTPS proxy instead of a target HTTP proxy.
- An HTTPS load balancer requires at least one signed SSL certificate installed on the target HTTPS proxy for the load balancer. The client SSL session terminates at the load balancer.
- HTTPS load balancer support the QUIC (Quick UDP Internet Connections) transport layer protocol. QUIC is a transport layer protocol that allows faster client connection initiation, eliminates head of line blocking in multiplexed streams, and supports connection migration when a client's IP address changes.
- To use HTTPS, you must create at least one SSL certificate that can be used by the target proxy for the load balancer. You can configure the target proxy with up to 10 SSL certificates.
- For each SSL certificate, you first create an SSL certificate resource which contains the SSL certificate information.
- SSL certificate resources are used only with load balancing proxies such as target HTTPS proxy or target SSL proxy

SSL PROXY LOAD BALANCING

- SSL proxy is a global load balancing service for ***encrypted***, non-HTTP traffic. This load balancer terminates user SSL connections at the load balancing layer, then balances the connections across your instances using the SSL or TCP protocols.
- These instances can be in multiple regions, and the load balancer automatically directs traffic to the closest region that has capacity.
- SSL proxy load balancing supports both IPv4 and IPv6 addresses for client traffic and provides intelligent routing, certificate management, security patching, and SSL policies.
- **Intelligent routing** means that this load balancer can route requests to backend locations where there is capacity.
- From a **certificate management** perspective, you only need to update your customer-facing certificate in one place when you need to switch those certificates. Also, you can reduce the management overhead for your virtual machine instances by using self-signed certificates on your instances.
- In addition, if vulnerabilities arise in the SSL or TCP stack, GCP will apply **security patches** at the load balancer automatically in order to keep your instances safe.

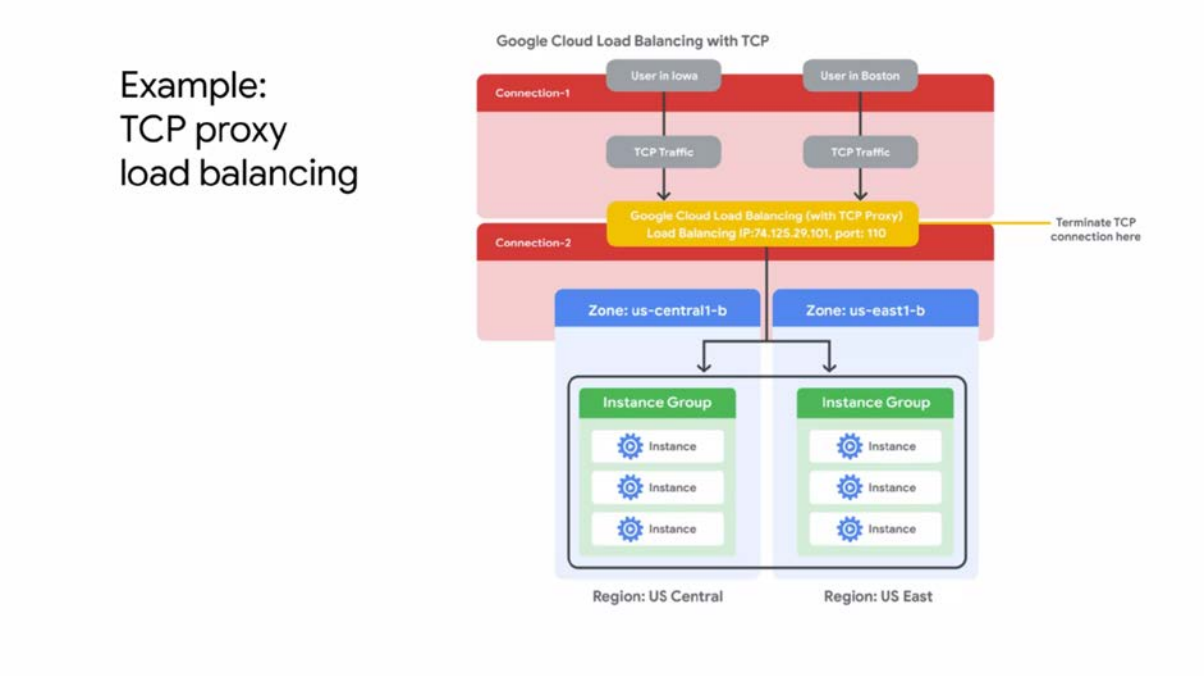
Example:
SSL proxy
load balancing



- This network diagram illustrates SSL proxy load balancing. In this example, traffic from users in Iowa and Boston is terminated at the global load balancing layer.
- From there, a separate connection established to the closest backend instance. In other words, the user in Boston would reach the US East region, and the user in Iowa would reach the US Central region, if there's enough capacity.
- Now, the traffic between the proxy and the backend can use SSL or TCP. I recommend using SSL.

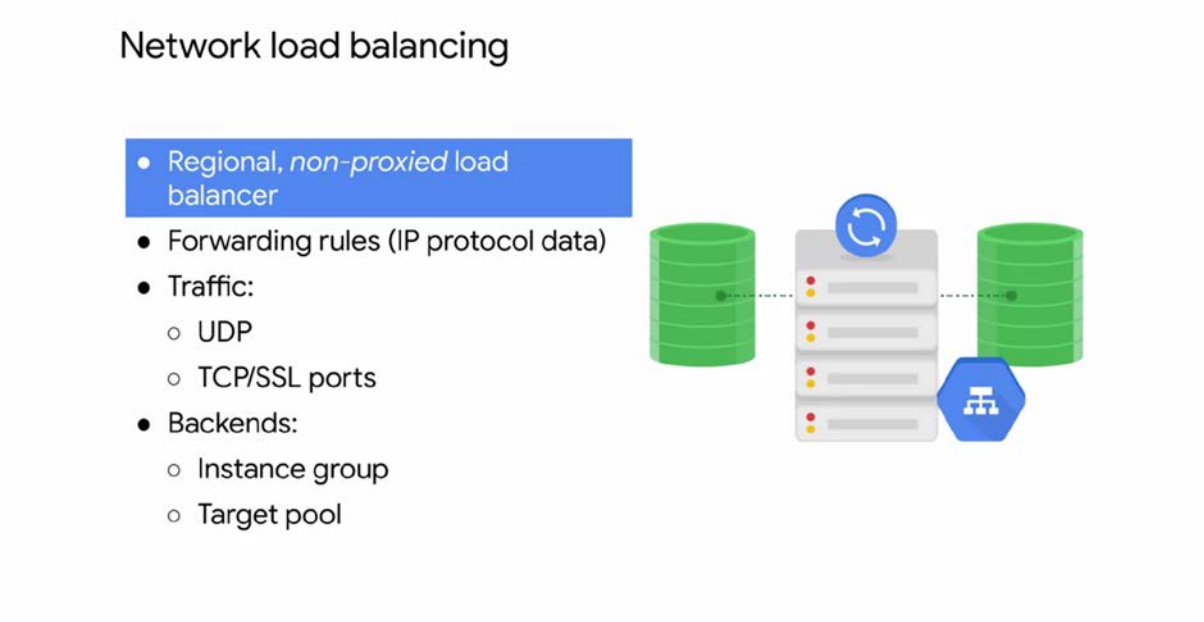
TCP LOAD BALANCING

- TCP proxy is a global load balancing service for ***unencrypted*** non-HTTP traffic. This load balancer terminates your customers TCP sessions at the load balancing layer, then forwards the traffic to your virtual machine instances using TCP or SSO.
- These instances can be in multiple regions and the load balancer automatically directs traffic to the closest region that has capacity. TCP proxy load balancing supports both IPv4 and IPv6 addresses for Client Traffic.
- Similar to SSL proxy load balancer, the TCP proxy load balancer provides **intelligent routing** and **security patching**.

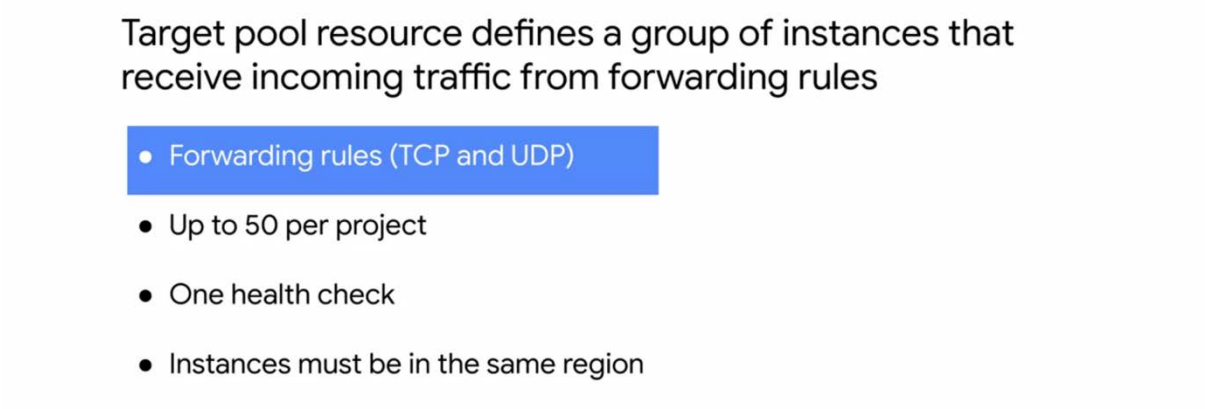


- This network diagram illustrates TCP proxy load balancing.
- In this example, traffic from users in Iowa and Boston is terminated at the Global Load Balancing layer. From there, a separate connection is established to the closest backend instance.
- As in the SSL proxy load balancing example, the users in Boston would reach the US East region and the user in Iowa which reach the US central region, if there's enough capacity.
- Now the traffic between the proxy and the backends can use SSL or TCP and I also recommend using SSL here.

NETWORK LOAD BALANCER

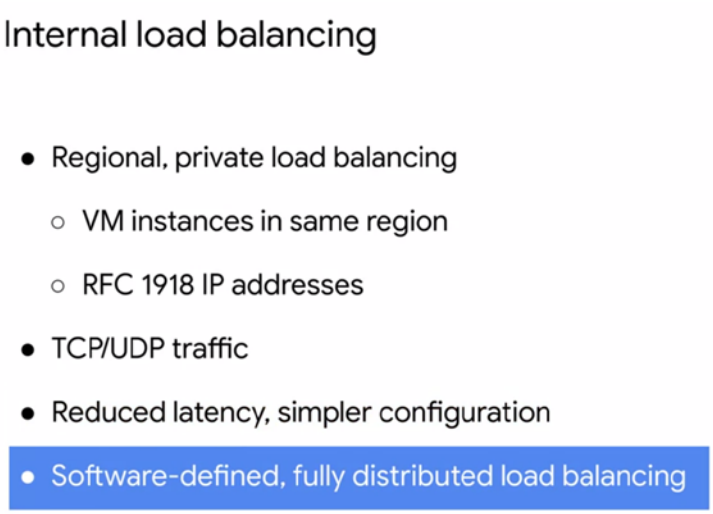


- Network load balancing is a regional **non proxied load balancing service**. In other words, all traffic is passed through the load balancer instead of being proxied and traffic can only be balanced between virtual machine instances that are in the same region unlike a global load balancer.
- This load balancing service uses forwarding rules to balance the load on your systems based on incoming IP protocol data such as address, port, and protocol type.
- You can use it to load balance UDP traffic and to load balance TCP and SSL traffic on ports that are not supported by the TCP proxy and SSL proxy load balancers.
- The back ends of a network load balancer can be a template-based instance group or target pooled resource.

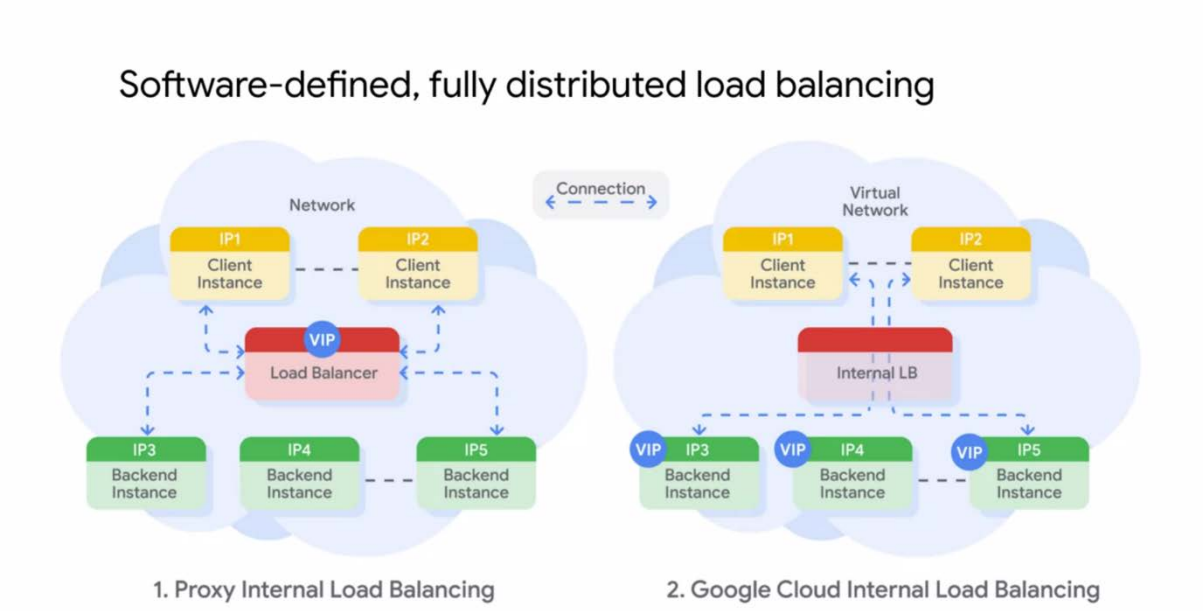


- A target pool resource defines a group of instances that receive incoming traffic from forwarding rules.
- When a forwarding rule direct traffic to a target pool, the load balancer picks an instance from these target pools based on hash of the source IP and port, and the destination IP and port.
- These target pools can only be used with forwarding rules that handled TCP and UDP traffic.
- Now each project can have up to 50 target pools and each target pool can have only one health check. Also, all the instances of a target pool must be in the same region which is the same limitation as for the network load balancer.

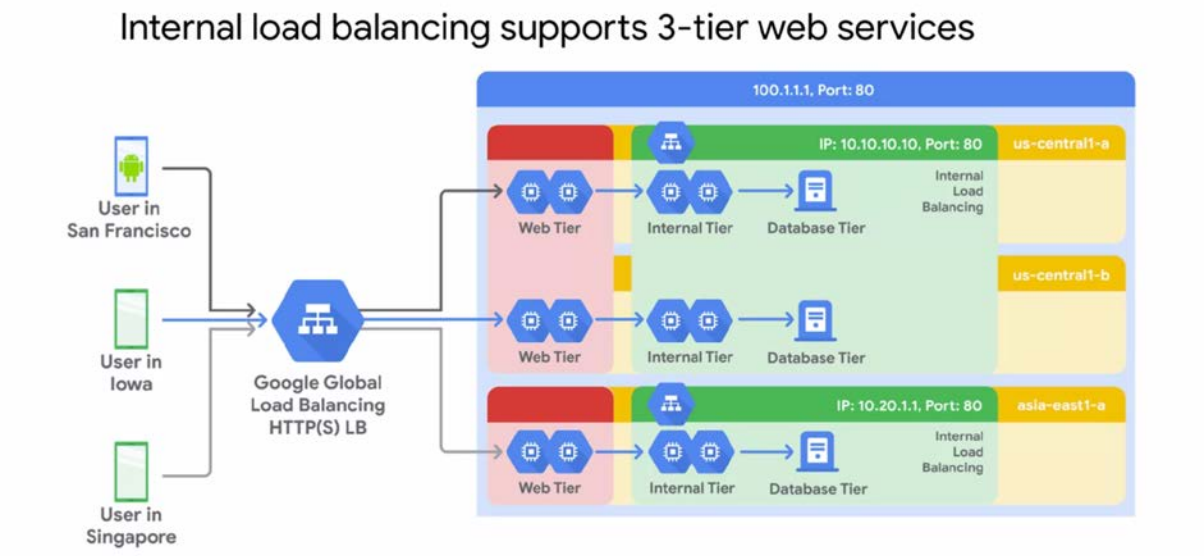
INTERNAL LOAD BALANCING



- Internal load balancing is a regional, private load balancing service for TCP and UDP based traffic.
- In other words, this load balancer enables you to run and scale your services behind a **private load balancing IP address.**
- This means that it is only accessible through the **internal IP address** of virtual machine instances that are in the **same region.**
- Therefore, use internal load balancing to configure an internal load balancing IP address, to act as the front end to your private backend instances.
- Because you don't need a public IP address for your load balanced service, your internal client requests to stay internal to your VPC network and the region.
- This often results in lower latency, because all your load balanced traffic will stay within Google's network, making your configuration much simpler.
- GCP internal load balancing is not based on a device or a virtual machine instance. Instead, it is a **software-defined, fully distributed load balancing** solution.
- In the traditional proxy model of internal load balancing as shown on the left, you configure an internal IP address on a load balancing device or instances, and your client instance connects to this IP address. Traffic coming to the IP address is terminated at the load balancer, and the load balancer selects a backend to establish a new connection to.
- Essentially, there are two connections. One between the client and the load balancer, and the one between the load balancer and the backend.



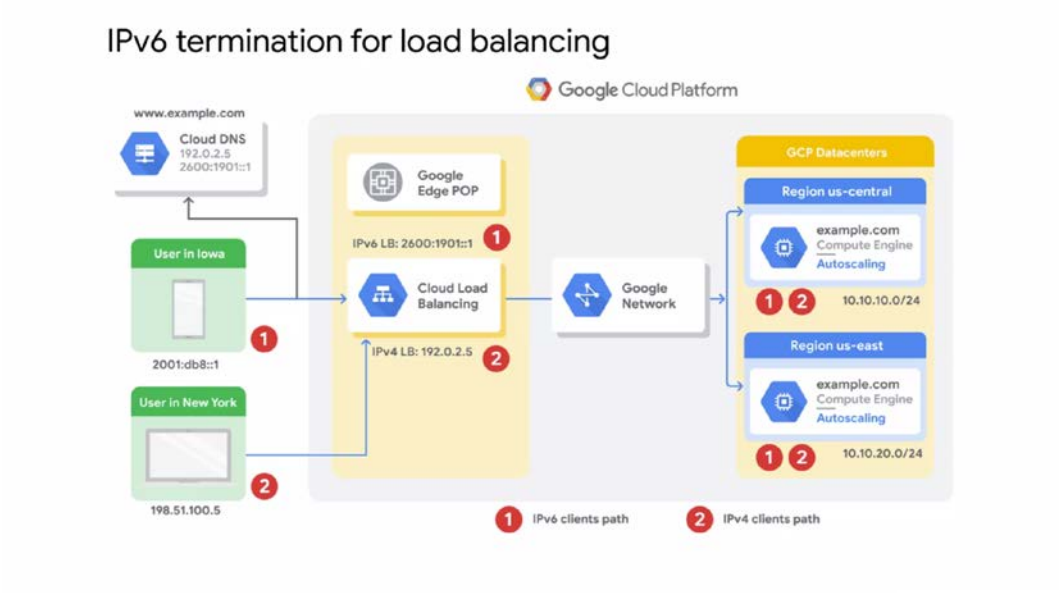
- GCP internal load balancing distributes client instance requests to the backend using a different approach, as shown on the right.
- It uses lightweight load balancing built on top of **Andromeda**, Google's network virtualization stack, to provide software-defined load balancing that directly delivers the traffic from the client instance to a backend instance.
- Now, internal load balancing enables you to support use-cases such as the traditional treaty or web service.

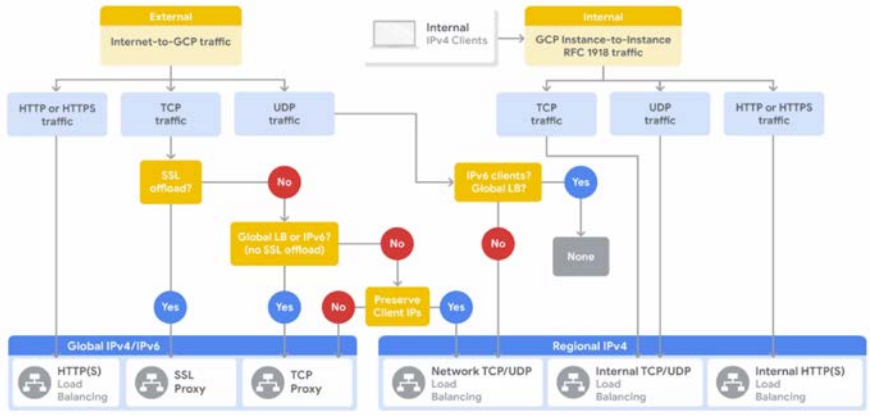


- In this example, the web tier uses an external HTTPS load balancer, that provides a single global IP address for users in San Francisco, Iowa and Singapore, and so on.
- The backends of this load balancer are located in the US-Central1 and Asia-East-1 region, because this is a global load balancer.
- These backends then access an internal load balancer in each region as the application or internal tier.
- The backends of this internal tier are located in US-Central1-A, US-Central1-B, and Asia-East1-B.

- The last tier is the database tier in each of these zones. The benefit of this three-tier approach is that neither the database tier nor the application tier is exposed externally. The simplified security and network pricing.

CHOOSING AN APT LOAD BALANCER





Summary of load balancers

Load balancer	Traffic type	Global/ Regional	External/ Internal	External ports for load balancing
HTTP(S)	HTTP or HTTPS	Global IPv4 IPv6	External	HTTP on 80 or 8080; HTTPS on 443
SSL Proxy	TCP with SSL offload			25, 43, 110, 143, 195, 443, 465, 587, 700, 993, 995, 1883, 5222
TCP Proxy	• TCP without SSL offload • Does not preserve client IP addresses			25, 43, 110, 143, 195, 443, 465, 587, 700, 993, 995, 1883, 5222
Network TCP/UDP	• TCP/UDP without SSL offload • Preserves client IP addresses	Regional IPv4		Any
Internal TCP/UDP	TCP or UDP		Internal	Any
Internal HTTP(S)	HTTP or HTTPS			HTTP on 80 or 8080; HTTPS on 443

AUTOMATION USING DEPLOYMENT MANAGER AND MARKETPLACE

- Calling the Cloud APIs from code is a powerful way to generate infrastructure. But writing code to create infrastructure also has some challenges. One issue is that the maintainability of the infrastructure depends directly on the quality of the software.

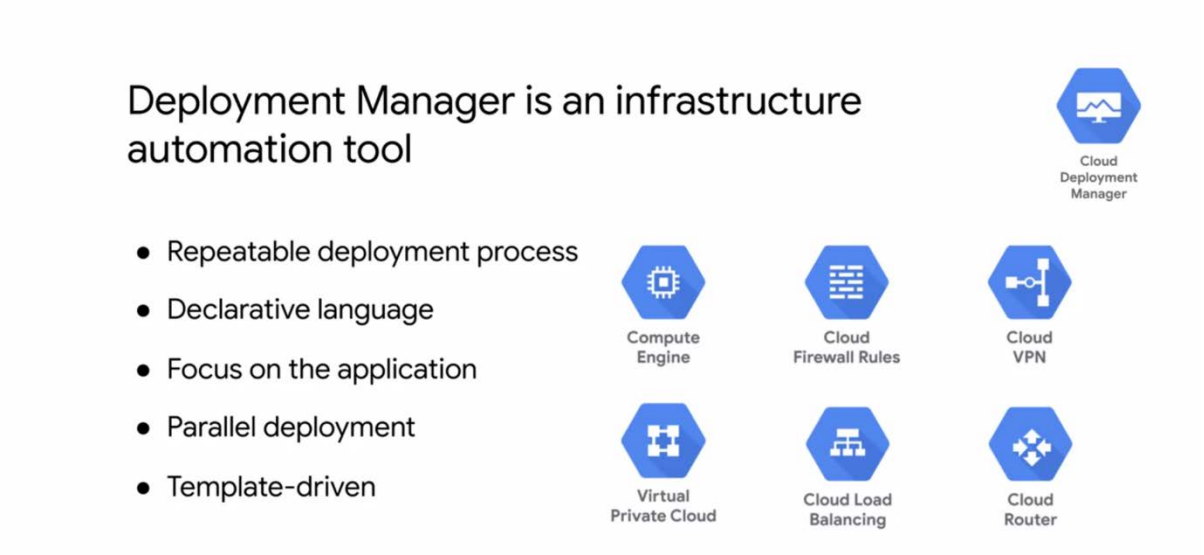
- For example, a program could have a dozen locations that call the Cloud APIs to create VMs. Fixing a problem with the definition of one VM would require first identifying which of the dozen calls actually created it.

- Standards software development best practices will apply and it's important to note that things could change rapidly requiring maintenance on your code.

- Deployment manager helps by offering another level of organization. Deployment manager uses a system of highly structured templates and configuration files to document the infrastructure in an easily readable and understandable format.

- Deployment manager conceals the actual Cloud API calls. So you don't need to write code and can focus on the definition of the infrastructure.

DEPLOYMENT MANAGER



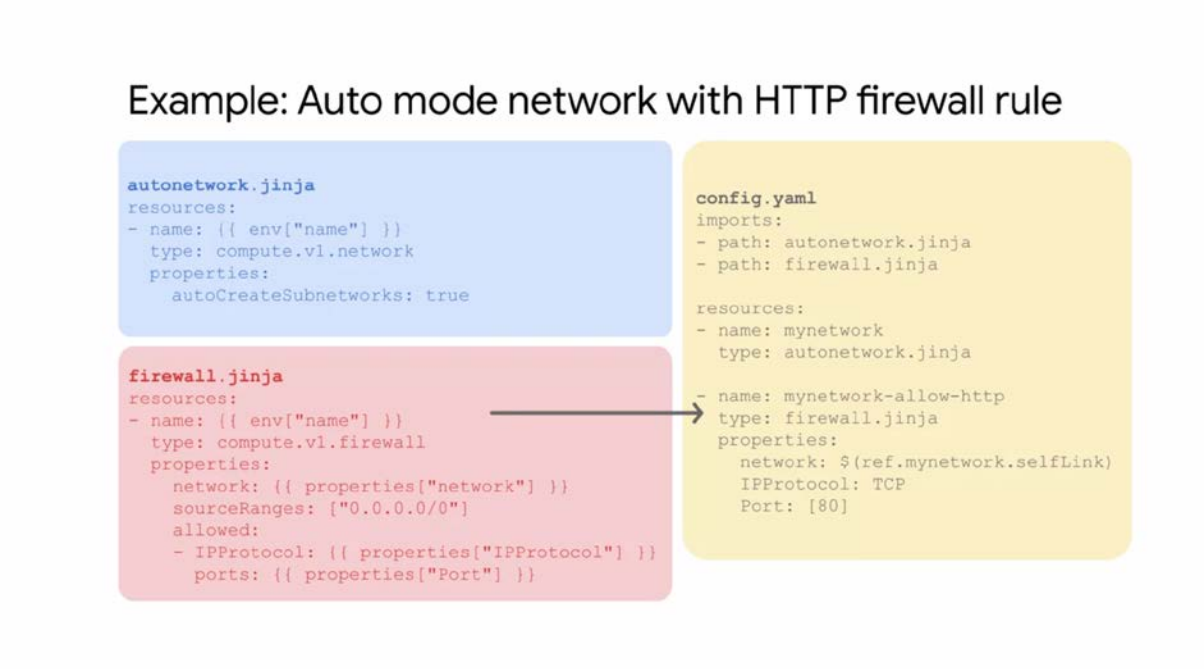
- Deployment Manager is an infrastructure deployment service that automates the creation and management of GCP resources for you. You just specify all the resources needed for your application in a **declarative format** and deploy your configuration.

- This deployment can be repeated over and over with consistent results and you can delete a whole deployment with one command or click.

- The benefit of a declarative approach is that it allows you to specify what the configuration should be and let the system figure out the steps to take. Instead of deploying each resource separately, you specify the set of resources which compose the application or service, allowing you to focus on the application.

- Unlike Cloud Shell, Deployment Manager will deploy resources in parallel. You can even abstract parts of your configuration into individual building blocks or templates that can be used for other configurations.

- Deployment Manager uses the underlying APIs of each GCP service to deploy your resources. This enables you to deploy almost everything we have seen so far from instances, instance templates and groups to VPC networks, firewall rules, VPN tunnels, Cloud routers, and load balancers.



- Before you get into the lab, let me walk you through a quick example that shows how Deployment Manager can be used to set up an auto mode network with an HTTP firewall rule.
- This whole deployment is put into into one single configuration.
- However, it's useful to parameterize your configuration with templates.
- Specifically, we're going to create one template for the auto mode network and one for the firewall rule.
- Therefore, if we want to create either of these resources somewhere else later on, we can use those templates.
- Let's start with the auto mode network template which we can write in Jinja2 or Python. Now, each resource must contain a name, type, and properties.
- Using an **invariant variable** to get the name from the top-level configuration which makes this template more flexible.
- For the type - the API for a VPC network which is compute.v1.network.
- By definition, an auto mode network automatically creates a subnetwork in each region. Therefore, I am setting the auto-create subnetworks property to true.
- Next, let's write the template for the HTTP firewall rule.
- For the name - using an invariant variable to get the name from the top-level configuration.
- For the type - defining the API for a firewall rule which is compute.v1.firewall.

- The properties section contains the network to apply this firewall rule to, the source IP ranges, and the protocols, and ports that are allowed.
- Except for the source IP ranges, I'm defining these properties as **template properties**. I will provide the exact properties from the top-level configuration, which makes this firewall rule extremely flexible.
- Essentially, I can use this firewall rule template for any network and any protocol and port combination.
- Next, let's write the top-level configuration in YAML syntax. I start by importing
- the templates that I want to use in this configuration, which are autonetwork.jinja and firewall.jinja.
- Then I define the auto mode network by giving it the name mynetwork and leveraging the auto network.jinja template. I could create more auto mode networks in this configuration with other names or simply reuse this template in other configurations later on.
- Now I define the firewall rule by giving it a name, leveraging the firewall.jinja template, referencing my network, and defining the IP protocol and port. I can easily add other ports such as 443 for HTTPS or 22 for SSH traffic.
- Using the **self link reference** for the network name ensures that the VPC network is created before the firewalled rule.
- This is very important because Deployment Manager creates all the resources in parallel unless you use references. You would get an error without the reference because you cannot create a firewall rule for a non-existing network.
- Now there are other infrastructure automation tools in addition to Deployment Manager that you can use in GCP. You can also use Terraform, CHEF, Puppet, Ansible, or Packer.
- All of these tools allow you to treat your infrastructure like software, which helps you decrease costs, reduce risk, and deploy faster by capturing infrastructure as code.

GCP MARKETPLACE

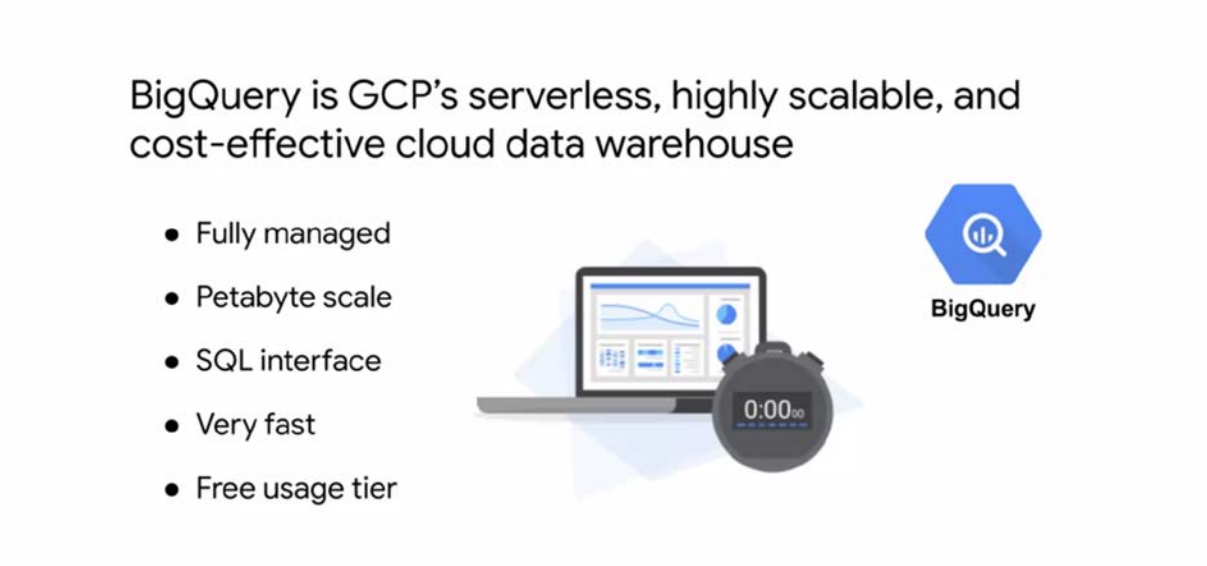
- GCP marketplace lets you quickly deploy functional software packages that run on GCP.

- Essentially, GCP marketplace offers production grade solutions from third-party vendors who have already created their own deployment configurations based on Deployment Manager.
- These solutions are built together with all of your projects GCP services.
- If you already have a license for a third party service, you might be able to use a Bring Your Own License solution.
- You can deploy a software package now and scale that deployment later when your applications require additional capacity.
- GCP even updates the images of these software packages to fix critical issues and vulnerabilities but doesn't update software that you have already deployed.
- You even get direct access to partner support.

MANAGED SERVICES

- As an alternative to infrastructure automation you can eliminate the need to create infrastructure by leveraging a managed service.
- Managed services are partial or complete solutions offered as a service. They exist on a continuum between platform as a service and software as a service depending on how much of the internal methods and controls are exposed.
- Using a managed service allows you to outsource a lot of the administrative and maintenance overhead to Google if your application requirements fits within the service offering.

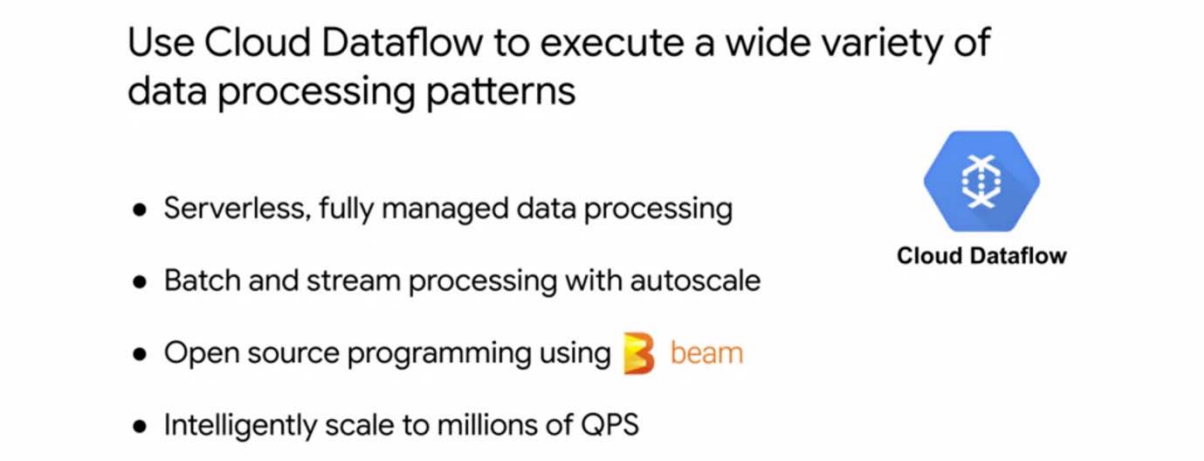
BIGQUERY



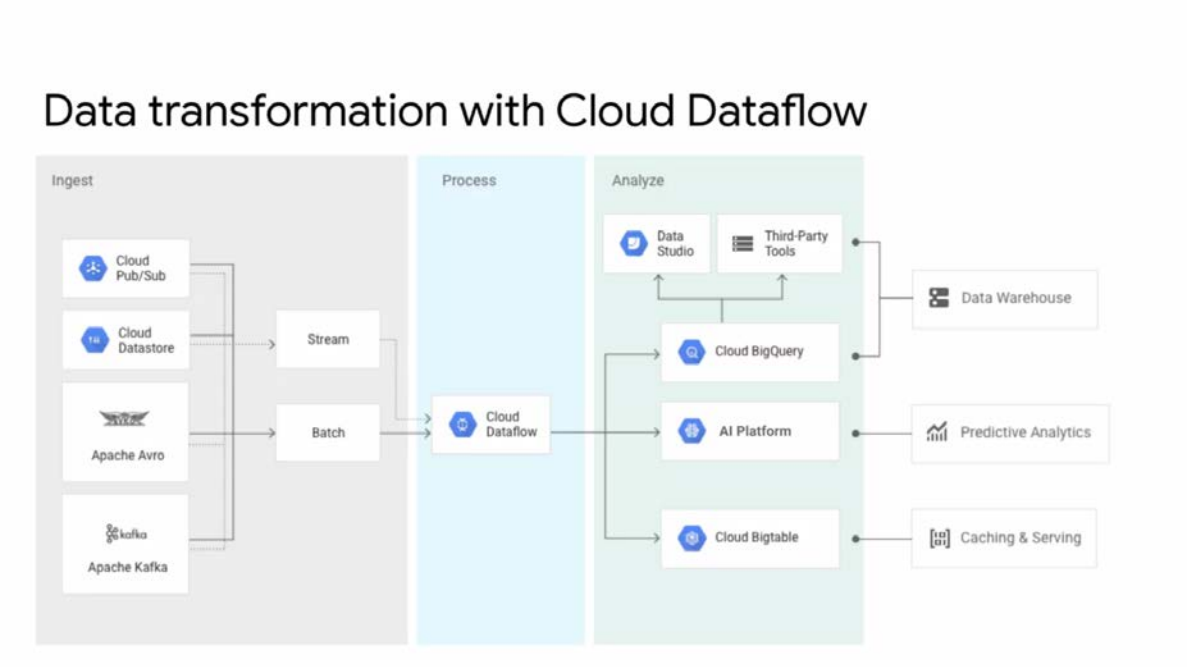
- BigQuery is GCP's serverless, highly scalable, and cost effective Cloud data warehouse.
- It's a petabyte scale data warehouse that allows for super-fast queries using the processing power of Google's infrastructure.

- Because there's no infrastructure for you to manage, you can focus on uncovering meaningful insights using familiar SQL, without the need for the database administrator.
- BigQuery is used by all types of organizations, and there's a free usage tier to help you get started.
- You can access BigQuery by using the GCP console, by using the command line tool, or by making calls to the BigQuery REST API, using the variety of client libraries such as Java,.NET or Python.
- There are also several third-party tools that you can use to interact with BigQuery, such as visualizing the data, or loading the data.
- A query on the table with over 100 billion rows processes over 4.1 terabyte, but takes less than a minute to execute. The same query would take hours if not days through a serial execution.

CLOUD DATAFLOW

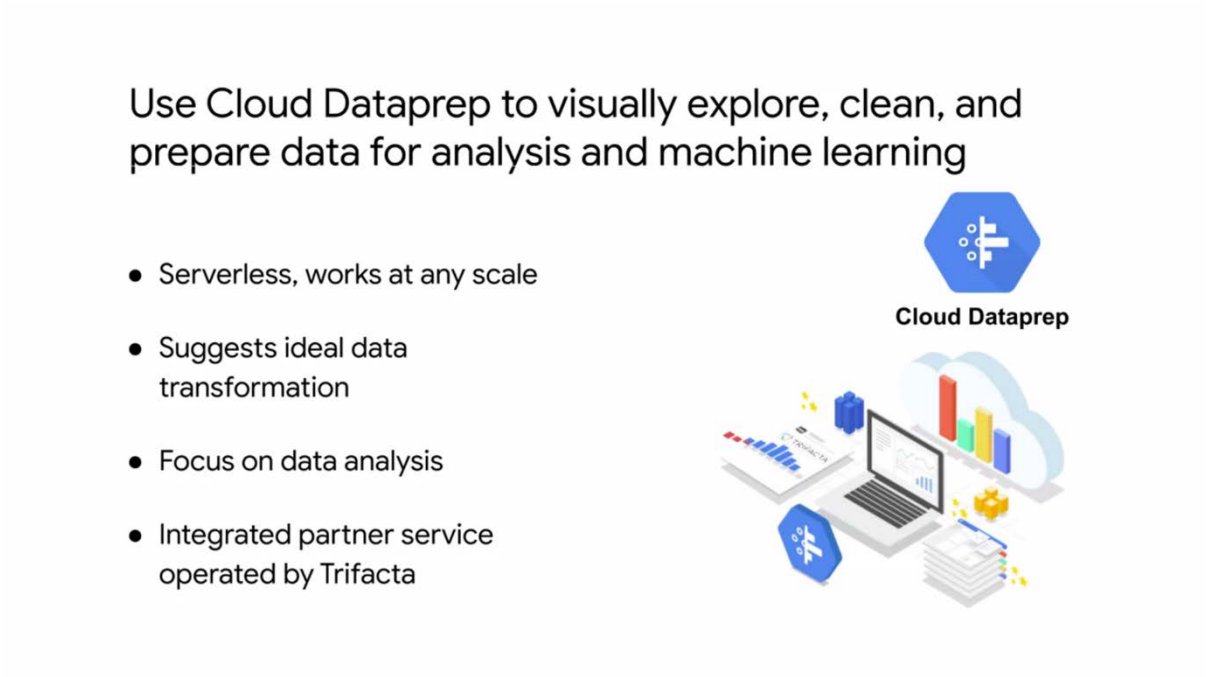


- Cloud Dataflow is a managed service for executing a wide variety of data processing patterns.
- It is essentially a fully managed service for transforming and enriching data in stream and batch modes with equal reliability and expressiveness.
- With Cloud Dataflow, a lot of the complexity of infrastructure setup and maintenance is handled for you.
- It's built on Google Cloud Infrastructure and auto-scaled to meet the demands of your data pipeline, allowing you to intelligently scale to millions of queries per second.
- Cloud Dataflow supports fast, simplified pipeline development via expressive SQL, Java, and Python APIs in the Apache Beam SDK which provides a rich set of windowing, and session analysis primitives as well as an ecosystem of source and sync connectors.
- Cloud Dataflow, is also tightly coupled with other GCP services like Stackdriver, so you can set a priority alerts and notifications to monitor your pipeline and the quality of data coming in and out.

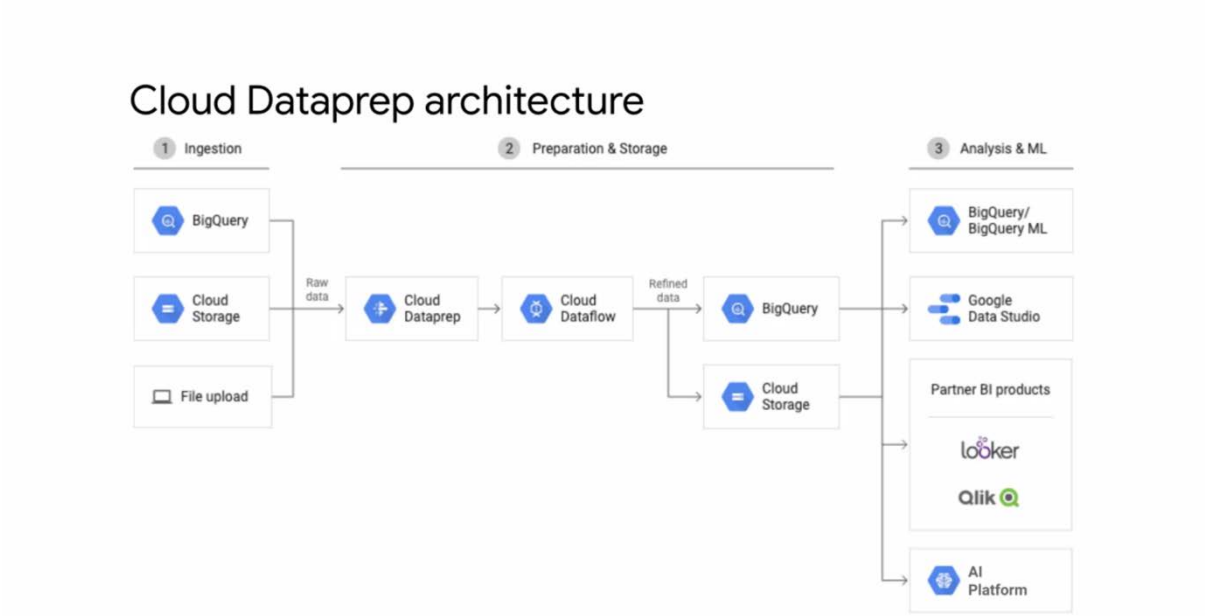


- This diagram shows some example use cases of Cloud Dataflow.
- As I just mentioned, Cloud Dataflow processes stream and batch data.
- This data could come from other GCP services like Cloud Datastore or Cloud Pub Sub which is Google's messaging and publishing service.
- The data could also be ingested from third party services like Apache Avro and Apache Kafka.
- After you transform the data with Cloud Dataflow, you can analyze it in BigQuery, AI platform, or even Cloud Bigtable. Using Data Studio, you can even build real-time dashboards for IoT devices

CLOUD DATAPREP

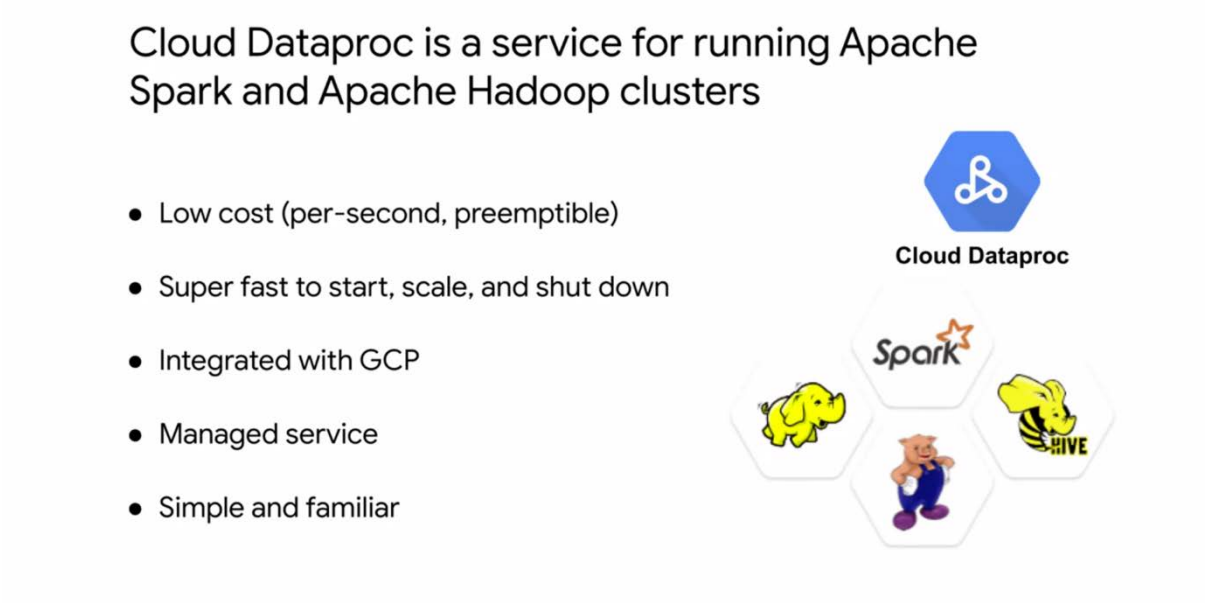


- Cloud Dataprep is an Intelligent Data Service for visually exploring, cleaning, and preparing structured and unstructured data for analysis reporting and Machine Learning.
- Because Cloud Dataprep is serverless and works at any skill, there's no infrastructure to deploy are manage.
- Your next ideal data transformation is suggested and predicted with each UI input so you don't have to write code. With automatic schema, data types, possible joins and anomaly detection,
- you can skip time-consuming Data Profiling and focus on Data Analysis.
- Cloud Dataprep is an integrated partner service operated by Trifacta and based on their industry leading Data Preparation Solution Trifacta Wrangler.
- Google works closely with Trifacta to provide a seamless user experience that removes the need for upfront software installation, separate licensing costs or ongoing operational overhead.
- Cloud Dataprep is fully-managed and scales on demand to meet your growing data preparation needs so you can stay focused on analysis.



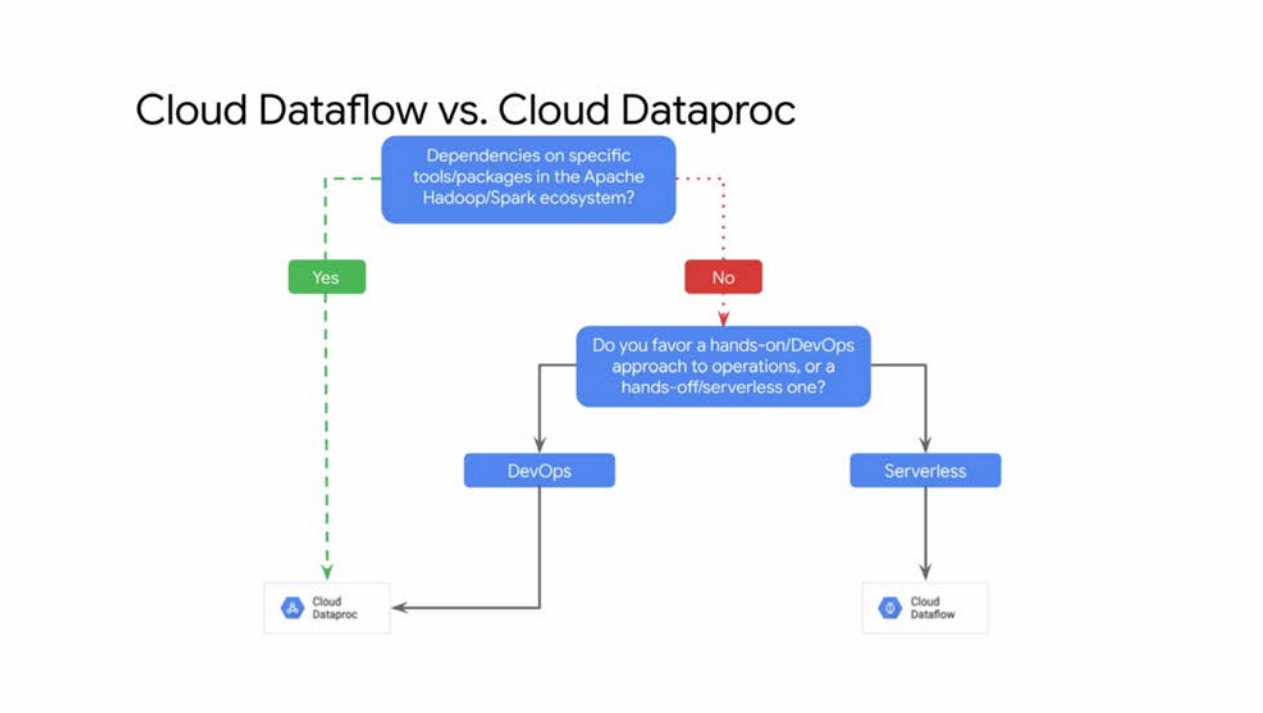
- Here's an example of a Cloud Dataprep architecture. As you can see, Cloud Dataprep can be leveraged to prepare raw data from BigQuery,
- Cloud Storage, or a file upload before ingesting it into a transformational pipeline like Cloud Data flow.
- The refined data can then be exported to BigQuery or Cloud Storage for analysis and machine learning.

CLOUD DATAPROC



- Cloud Dataproc is a fast easy to use fully managed Cloud service for running Apache Spark and Apache Hadoop clusters in a simpler way.

- You only pay for the resources you use with per second billing.
- If you leverage preemptible instances in your cluster, you can reduce your costs even further.
- Without using Cloud Dataproc, it can take from five to 30 minutes to create Spark and Hadoop clusters On-premise or through other infrastructure as a service providers.
- Cloud Dataproc clusters are quick to start, scale, and shut down with each of these operations taking 90 seconds or less on average.
- This means you can spend less time waiting for clusters and more hands-on time working with your data.
- Cloud Dataproc has built-in integration with other GCP services such as BigQuery, Cloud Storage, Cloud Bigtable, Stackdriver Logging, and Stackdriver monitoring.
- This provides you with the complete data platform rather than just a Spark or Hadoop cluster.
- As a managed service, you can create clusters quickly, manage them easily, and save money by turning clusters off when you don't need them. With less time and money spent on administration, you can focus on your jobs and your data.
- If you're already using Spark, Hadoop, Pig or Hive you don't even need to learn new tools or API's is to use Cloud Dataproc. This makes it easy to move existing projects into Cloud Dataproc without redevelopment.



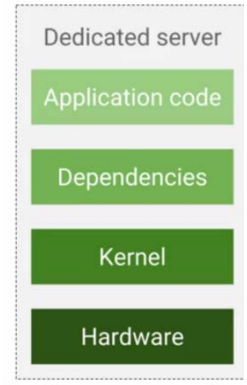
Cloud Shell provides the following features and capabilities:

- Temporary Compute Engine VM

- Command-line access to the instance through a browser
- 5 GB of persistent disk storage (`$HOME dir`)
- Preinstalled Cloud SDK and other tools
- `gcloud`: for working with Compute Engine, Google Kubernetes Engine (GKE) and many Google Cloud services
- `gsutil`: for working with Cloud Storage
- `kubectl`: for working with GKE and Kubernetes
- `bq`: for working with BigQuery
- Language support for Java, Go, Python, Node.js, PHP, and Ruby
- Web preview functionality
- Built-in authorization for access to resources and instances

After 1 hour of inactivity, the Cloud Shell instance is recycled. Only the `/home` directory persists. Any changes made to the system configuration, including environment variables, are lost between sessions

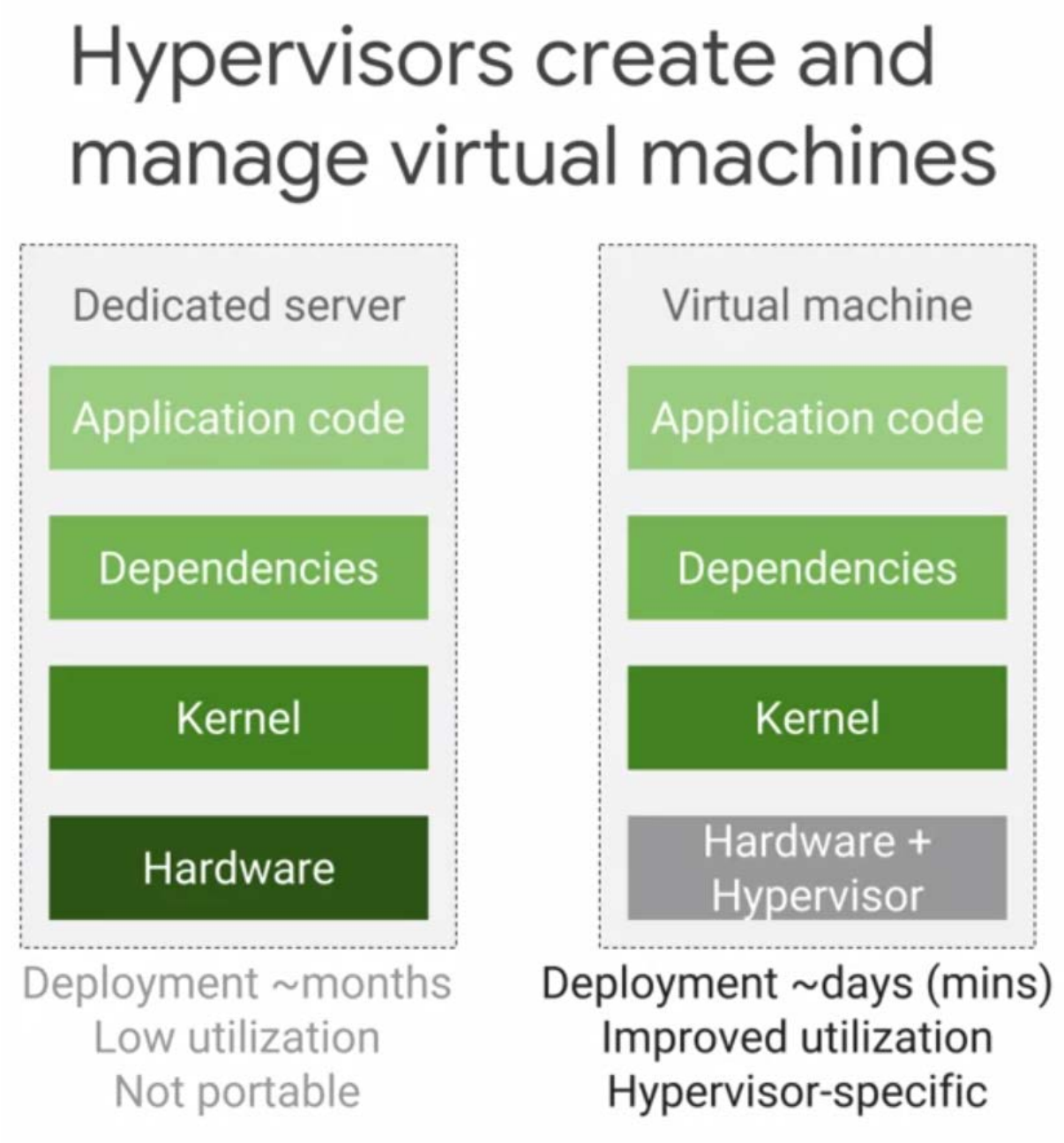
WHY CONTAINERIZE ?



- It's now not very long ago, the default way to deploy an application was on its own physical computer. To set one up, you'd find some physical space, power, cooling, network connectivity for it, and then install an operating system, any software dependencies, and then finally the application itself.
- If you need more processing power, redundancy, security, or scalability, you'd have to simply add more computers. It was very common for each computer to have a single-purpose. For example, a database, web server, or content delivery.

- This practice as you might imagine, wasted resources and it took a lot of time to deploy and maintain and scale. It also wasn't very portable at all.

- Applications were built for a specific operating system and sometimes even for specific hardware as well.



- In comes the dawn of virtualization. Virtualization helped by making it possible to run multiple virtual servers and operating systems on the same physical computer.

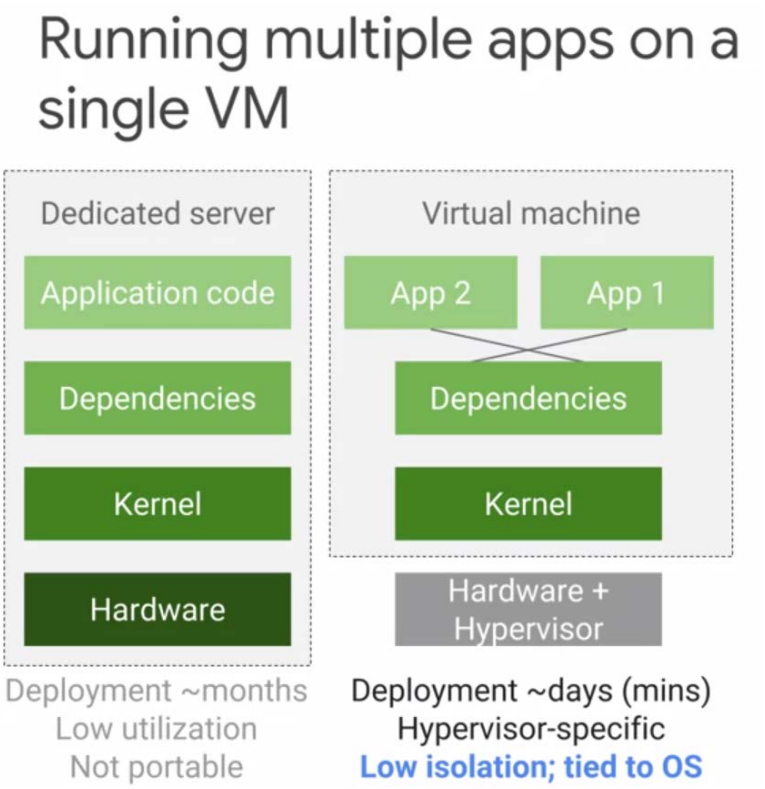
- A hypervisor is the software layer that breaks the dependencies of an operating system with its underlying hardware, and allow several virtual machines to share that same hardware. KVM is one well-known hypervisor.

- Today you can use virtualization to deploy new servers fairly quickly. Now adopting virtualization means that it takes us less time to deploy new solutions, we waste less of the resources on

- those physical computers that we're using, and we get some improved portability because virtual machines can be imaged and then moved around.

- However, the application, all of its dependencies and operating system are still bundled together and it's not very easy to move from a VM from one hypervisor product to another.

- Every time you start up a VM, it's operating system still takes time to boot up.

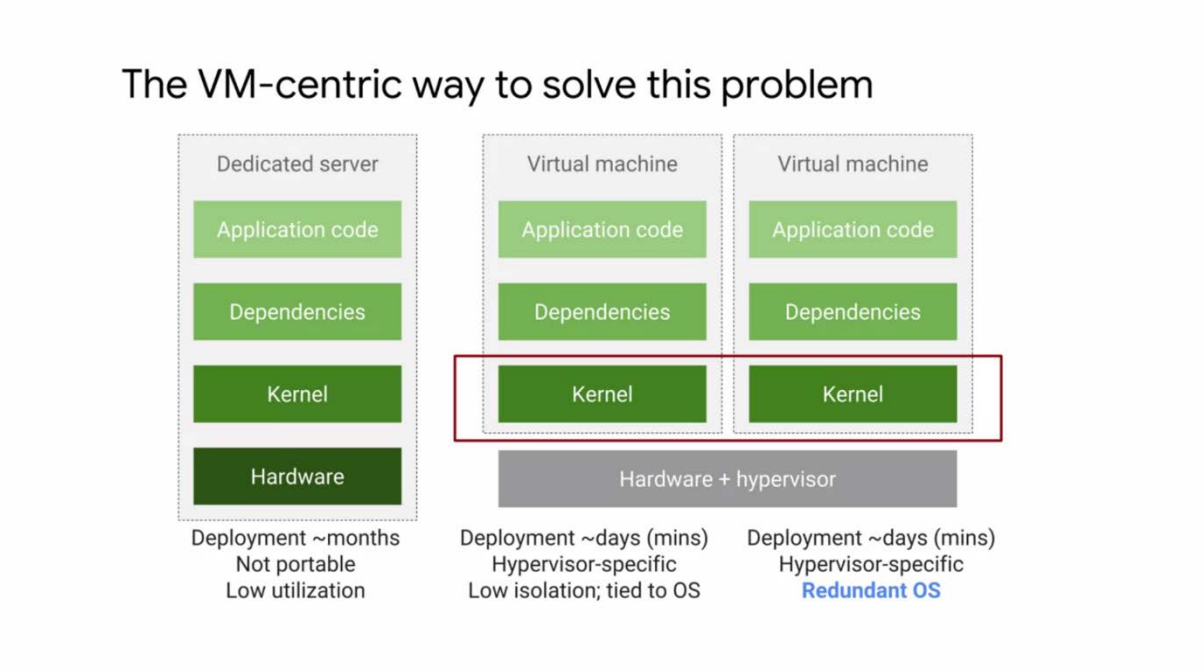


- Running multiple applications within a single VM also creates another tricky problem, applications that share dependencies are not isolated from each other, the resource requirements from one application, can starve out other applications of the resources that they need.

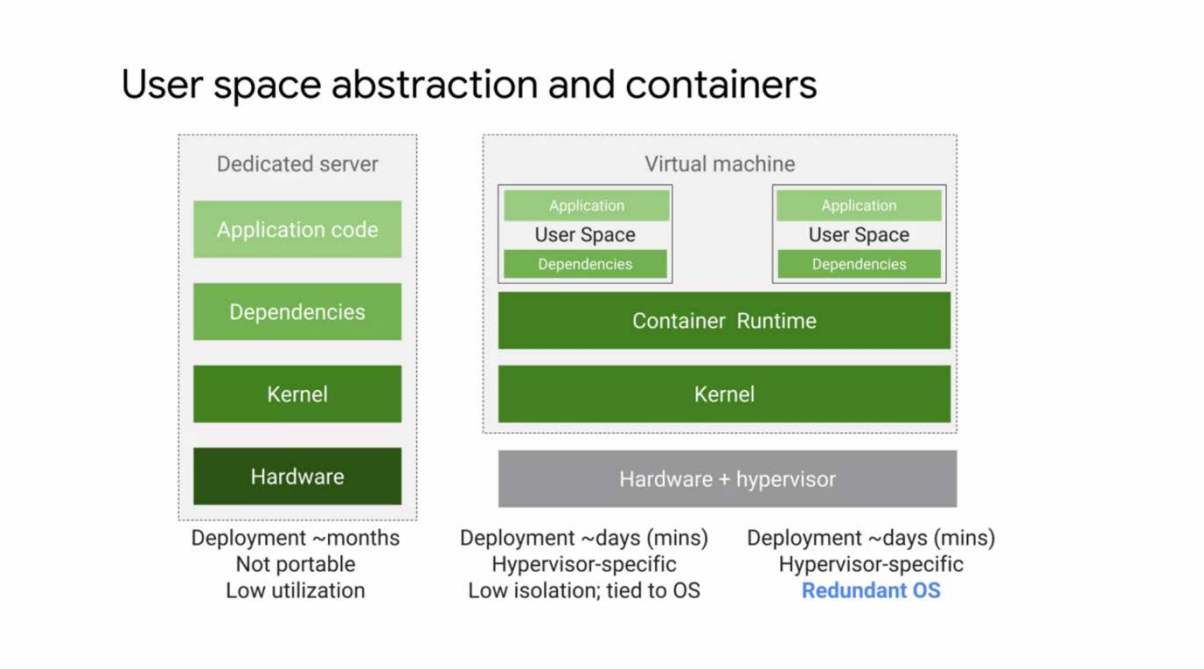
- Also, a dependency upgrade for one application might cause another to simply stop working.

- You can try to solve this problem with rigorous software engineering policies. For example, you could lock down the dependencies that no application is allowed to make changes, but this leads to new problems because dependencies do need to be upgraded occasionally.

- You can add integration tests to ensure that applications work. Integration tests are great, but dependency problems can cause new failure modes that are harder to troubleshoot, and it really slows down development if you have to rely on integration tests to simply just perform basic integrity checks of your application environment.



- Now, the VM-centric way to solve this problem is to run a dedicated virtual machine for each application. Each application maintains its own dependencies, and the kernel is isolated.
- So one application won't affect the performance of another. One you can get as you can see here, is two complete copies of the kernel that are running.
- But here too we can run into issues as you're probably thinking. Scale this approach to hundreds
- of thousands of applications, and you can quickly see the limitation. Just imagine trying to do a simple kernel update.
- So for large systems, dedicated VMs are redundant and wasteful. VMs are also relatively slow to start up because the entire operating system has to boot.



- A more efficient way to resolve the dependency problem is to implement abstraction at the level of the application and its dependencies.
- You don't have to virtualize the entire machine or even the entire operating system, but just the user space.
- Again, the user space is all the code that resides above the kernel, and includes the applications and their dependencies.
- **Containers are isolated user spaces for running application code.** Containers are lightweight because they don't carry a full operating system, they can be scheduled or packed tightly onto the underlying system, which is very efficient.
- They can be created and shut down very quickly because you're just starting and stopping the processes that make up the application and not booting up an entire VM and initializing an operating system for each application.
- Developers appreciate this level of abstraction because they don't want to worry about the rest of the system.
- You now understand containers as delivery vehicles for application code, they're lightweight, stand-alone, resource efficient, portable execution packages.
- You develop application code in the usual way, on desktops, laptops, and servers. The container allows you to execute your final code on VMs without worrying about software dependencies like application run times, system tools, system libraries, and other settings.
- You package your code with all the dependencies it needs, and the engine that executes your container, is responsible for making them available at runtime.
- Containers appeal to developers because they're an application-centric way to deliver high performance and scalable applications.
- Containers also allow developers to safely make assumptions about the underlying hardware and software. With a Linux kernel underneath, you no longer have code that

works in your laptop but doesn't work in production, the container's the same and runs the same anywhere.

- You make incremental changes to a container based on a production image, you can deploy it very quickly with a single file copy, this speeds up your development process.

- Finally, containers make it easier to build applications that use the microservices design pattern.

- That is, loosely coupled, fine-grained components. This modular design pattern allows the operating system to scale and also upgrade components of an application without affecting the application as a whole.

CONTAINERS

- An application and its dependencies are called an **image.** A **container** is simply a running instance of an image. By building software into Container images, developers can easily package and ship an application without worrying

- about the system it will be running on.

- You need software to build Container images and to run them. Docker is one tool that does both.

- **Docker** is an open source technology that allows you to create and run applications in containers but it doesn't offer a way to orchestrate those applications at scale like Kubernetes does.

- Google's **Cloud build** : helps create Docker formatted Container images.

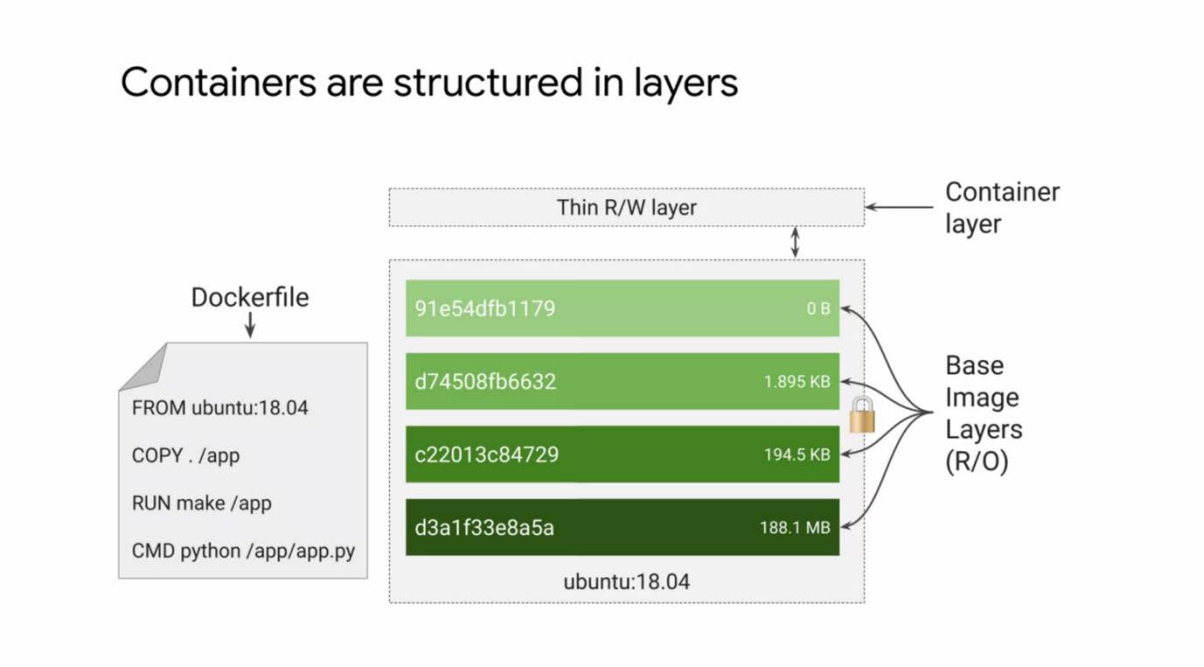
- Containers are not an intrinsic primitive feature of Linux. Instead their power to isolate workloads is derived from the composition of several technologies.

    - One foundation is the Linux process. Each Linux process has its own virtual memory address phase, separate from all others. Linux processes are rapidly created and destroyed.

    - Containers use Linux namespaces to control what an application can see, process ID numbers, directory trees, IP addresses, and more. By the way, Linux namespaces are not the same thing as Kubernetes Namespaces,

    - Containers use Linux cgroups to control what an application can use, its maximum consumption of CPU time, memory, IO bandwidth, other resources.

    - Finally, Containers use Union File Systems to efficiently encapsulate applications and their dependencies into a set of clean minimal layers.



- A container image is structured in layers.
- The tool you use to build the image reads instructions from a file called, "The Container manifest." In the case of a Docker formatted Container Image, that's called a Docker file.
- Each instruction in the Docker file specifies a layer inside the container image.
- Each layer is, "Read only." When a Container runs from this image, it will also have a writable ephemeral top-most layer.
- This Docker file will contain four commands, each of which creates a layer.
    - The From statement starts out by creating a base layer pulled from a public repository. This one happens to be the Ubuntu Linux runtime environment of a specific version.
    - The Copy command adds a new layer containing some files copied in from your build tools current directory.
    - The Run command builds your application using the make command and puts the result of the build into a third layer.
    - Finally, the last layer specifies what command to run within the container when it's launched.
- Each layer is only a set of differences from the layer before it. When you write a Docker file, you should organize the layers least likely to change through to the layers that are most likely to change. .
- These days, the best practice is not to build your application in the very same container that you ship and run. After all, your build tools are at best just cluttered and deployed Container and at worst, are an additional attack service.

- Today, Application Packaging relies on a **multi-stage build process** in which one Container builds the final executable image. A separate container receives only what's needed to actually run the application.

- When you launch a new container from an image, the Container Runtime adds a new writable layer on the top of the underlying layers. This layer is often called the **Container layer**. All changes made to the running container, such as writing new files, modifying existing files, and deleting files are written to this thin writable Container layer in their ephemeral, when the Containers deleted the contents of this writeable layer are lost forever.

- The underlying Container Image itself remains unchanged. This fact about Containers has an implication for your application design. Whenever you want to store data permanently, you must do so somewhere other than a running container image. Because each Container has its own writable Container layer and all changes are stored in this layer.

- Multiple Containers can share access to the same underlying image and yet have their own data state.

- Because each layer is only a set of differences from the layer before it, you get smaller images.

- For example, your base application image, maybe 200 megabytes, but the difference, the next point release might only be 200 kilobytes. When you build a container, instead of copying the whole image, it creates a layer with just the differences.

- When you run a container, that Container Runtime pulls down the layers it needs. When you update, you only need to copy the difference, this is much faster than running a new virtual machine.

- Google Container Registry is integrated with Cloud IAM. So for example, you can use it to store your images that aren't public.

- Instead, they're private to your project. You can also find Container images and other public repositories,

- Docker Hub Registry, GitLab and others. The open source Docker command is a popular way to build your own Container images.

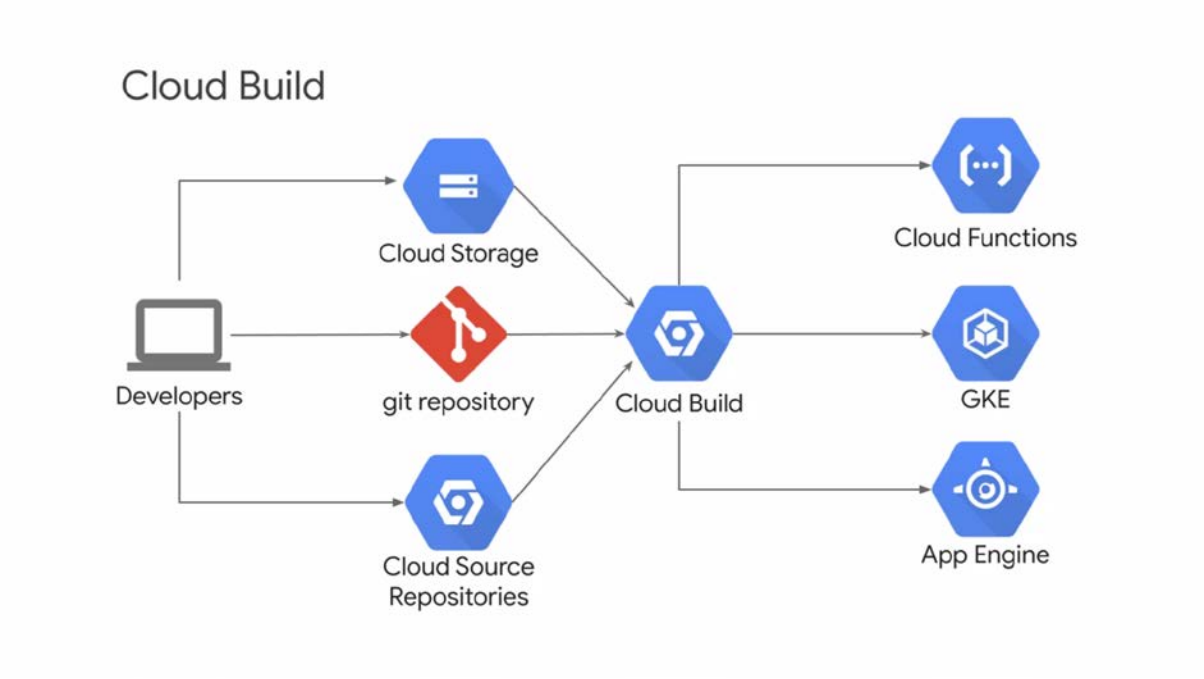
- It's widely known and widely available.

- One downside, whoever a building Containers with a Docker command is that you must trust the computer that you do your builds on. Google provides a managed service for building Containers that's also integrated with Cloud IAM. This service's called, **"Cloud Build,"** and we'll use it in this course.

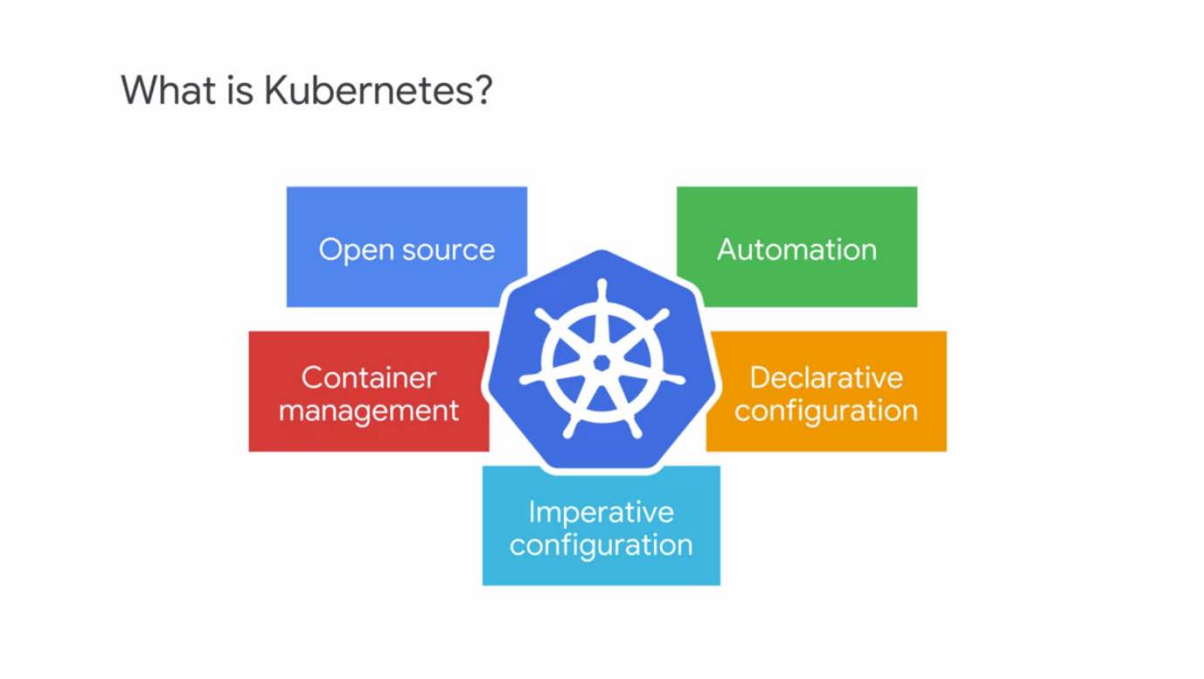
- Cloud Build can retrieve the source code for your builds from a variety of different storage locations. Cloud Source Repositories, Cloud Storage, which is GCP is Object Storage service or git compatible repositories like GitHub and Bitbucket to generate a buildup Cloud Build, you define as series of steps.

- For example, you can configure build steps to fetch dependencies, compile source code, run integration tests, or use tools such as Docker, Gradle, and Maven. Each build step and Cloud build runs in a Docker container.

- Then Cloud build can deliver your newly built images to various execution environments, not only GKE, but also App Engine and Cloud Functions.

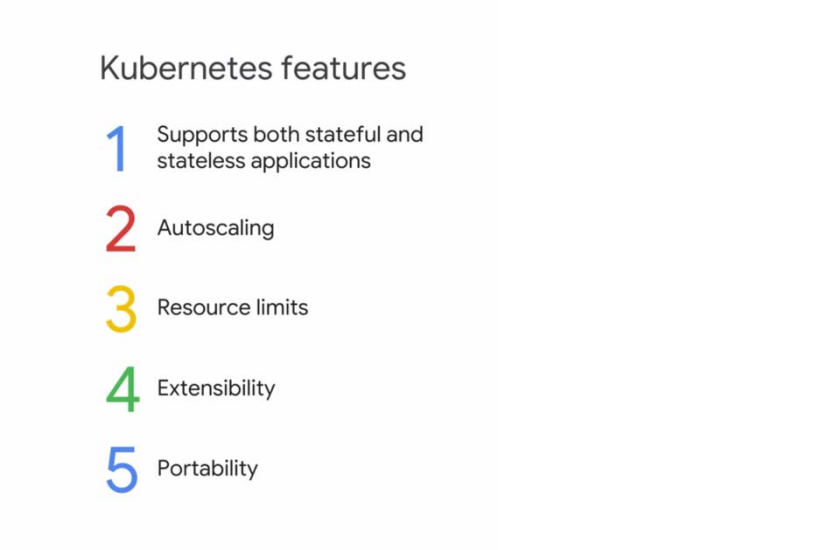


KUBERNETES



- We need to have a network fabric that lets containers find each other.
- Kubernetes is an open source platform that helps you orchestrate and manage your container infrastructure On-premises or in the Cloud.

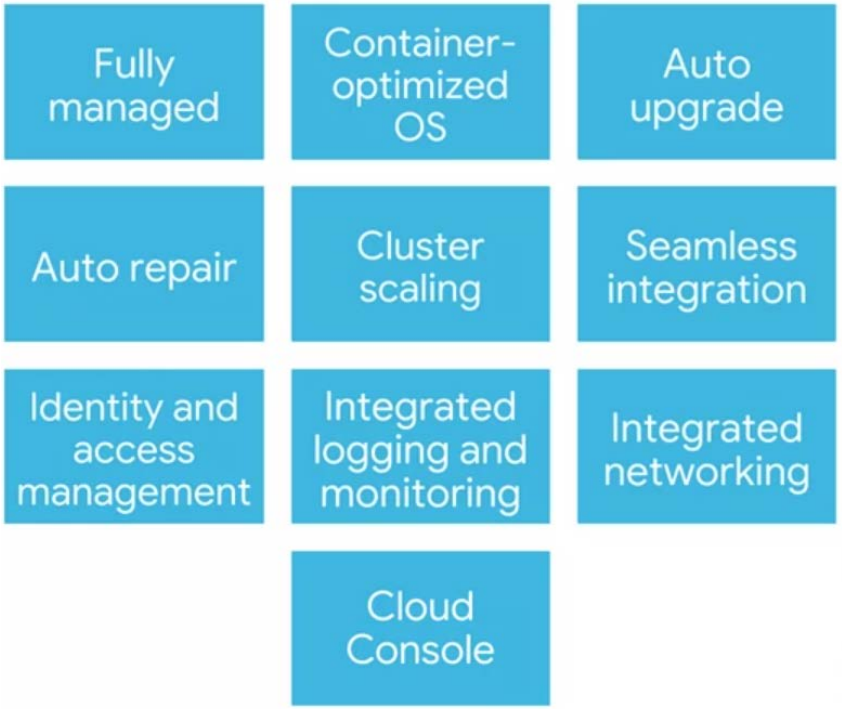
- It's a container centric management environment. It automates the deployment scaling, load balancing, logging, monitoring, and other management features of containerized applications.
- These are the features that are characteristic of a typical platform as service solutions.
- Kubernetes also facilitates the features of an infrastructure as a service, such as allowing a wide range of user preferences and configuration flexibility.
- Kubernetes supports declarative configurations. When you administer your infrastructure declaratively, you describe the desired state you want to achieve instead of issuing a series of commands to achieve that desired state.
- Kubernetes job is to make the deployed system conform to your desired state and then keep it there in spite of failures. Declarative configuration saves you work. Because the system is desired state is always documented, it also reduces the risk of error.
- Kubernetes also allows imperative configuration in which you issue commands to change the system state. But administering Kubernetes as scale imperatively, will be a big missed opportunity. One of the primary strengths of Kubernetes is its ability to automatically keep a system in a state that you declare. Experienced Kubernetes administrators use imperative configuration only for quick temporary fixes and as a tool in building a declarative configuration.



- Kubernetes supports different workload types. It supports stateless applications such as an Nginx or Apache web server, and stateful applications where user in session data can be stored persistently.
- It also supports batched jobs and demon tasks.
- Kubernetes can automatically scale in and out containerized applications based on resource utilization.
- You can specify resource requests levels and resource limits for your workloads and Kubernetes will obey them.
- These resource controls like Kubernetes, improve overall workload performance within the cluster.

- Developers extend Kubernetes through a rich ecosystem of plugins and add-ons.
- Because it's open source, Kubernetes also supports workload portability across On-premises or multiple Cloud service providers such as GCP and others. This allows Kubernetes to be deployed anywhere. You can move Kubernetes workloads freely without a vendor login.

GOOGLE K8S ENGINE – K8S AS A MANAGED SERVICE



- Google cloud's managed service offering for Kubernetes is called Google, Kubernetes Engine or GKE.
- It will help you deploy, manage and scale Kubernetes environments for your containerized applications on GCP.
- More specifically, GKE is a component of the GCP compute offerings. It makes it easy to bring your Kubernetes workloads into the cloud.
- GKE is fully managed, which means that you don't have to provision the underlying resources.
- GKE uses a **container-optimized operating system**. These operating systems are maintained by Google. And they're optimized to scale quickly and with a minimal resource footprint.
- When you use GKE, you start by directing the service to instantiate a Kubernetes system for you. This system is called a **cluster**. GKE's auto upgrade feature can be enabled to ensure that your clusters are automatically upgraded with the latest and greatest version of Kubernetes.

- The virtual machines that host your containers inside of a GKE cluster are called **nodes.** If you enable GKE's auto repair feature, the service will automatically repair unhealthy nodes for you.

- It will make periodic health checks on each node in the cluster. If a node is determined to be unhealthy and requires repair, GKE would drain the node. In other words, it will cause it's workloads to gracefully exit and then recreate that node.

- Just as Kubernetes support scaling workloads, GKE support scaling the cluster itself. GKE seamlessly integrates with Google Cloud build and container registry. This allows you to automate deployment using private container images that you've securely stored in container registry.

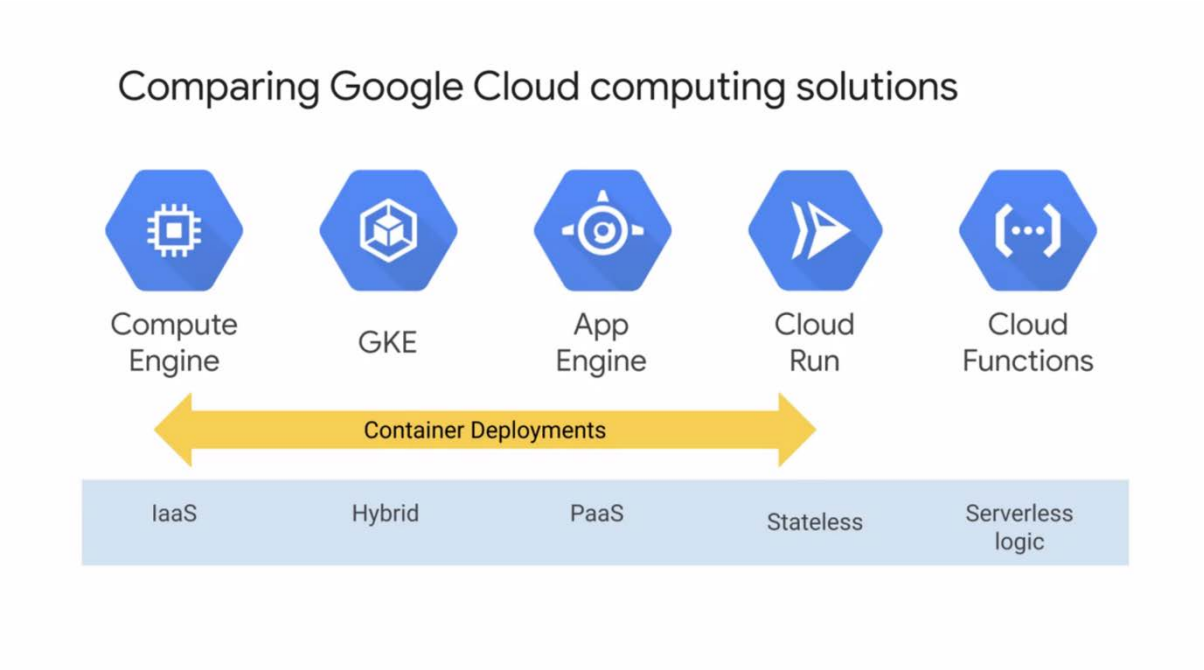
- GKE also integrates with Google's identity and access management, which allows you to control access through the use of accounts and role permissions.

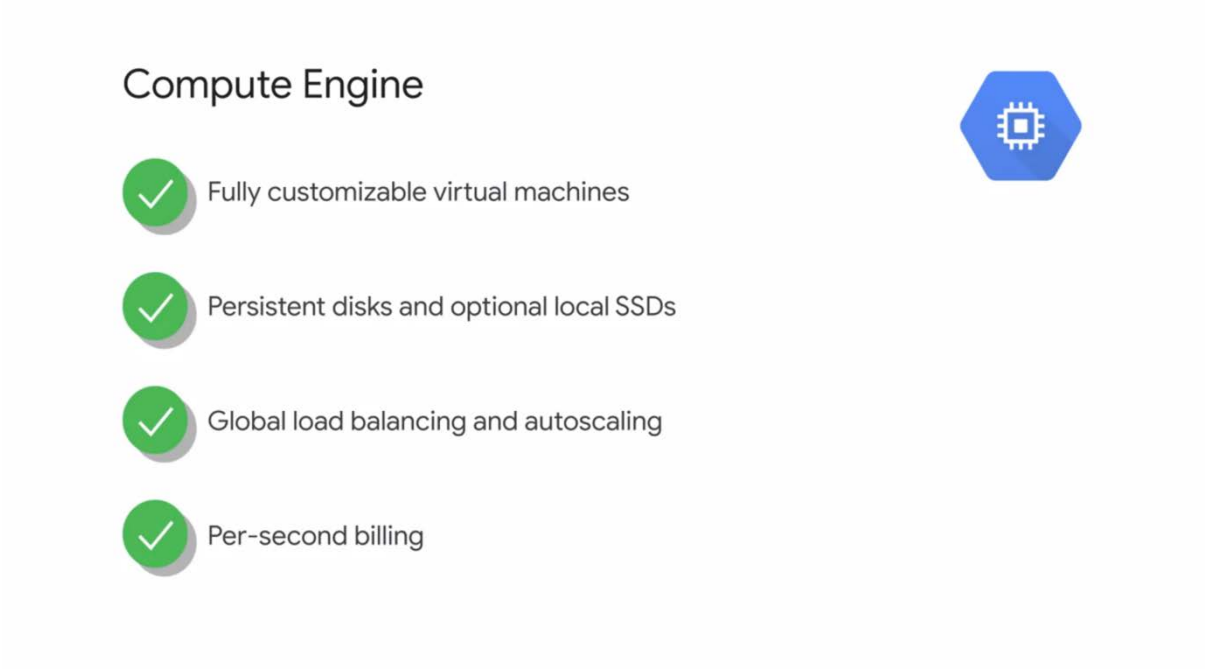
- **Stackdriver** is Google Cloud system for monitoring and management for services, containers, applications, and infrastructure. GKE integrates with Stackdriver monitoring to help you understand your applications performance.

- GKE is integrated with Google virtual private clouds or VPCs, it makes use of GCP's networking features.

- And finally, the GCP console provides insights into GKE clusters and the resources, and it allows you to view, inspect and delete resources in those clusters. You might be aware that open source Kubernetes contains a dashboard, but it takes a lot of work to set it up securely. But the GCP console is a dashboard for GKE clusters and workloads that you don't have to manage. And it's more powerful than the Kubernetes dashboard.

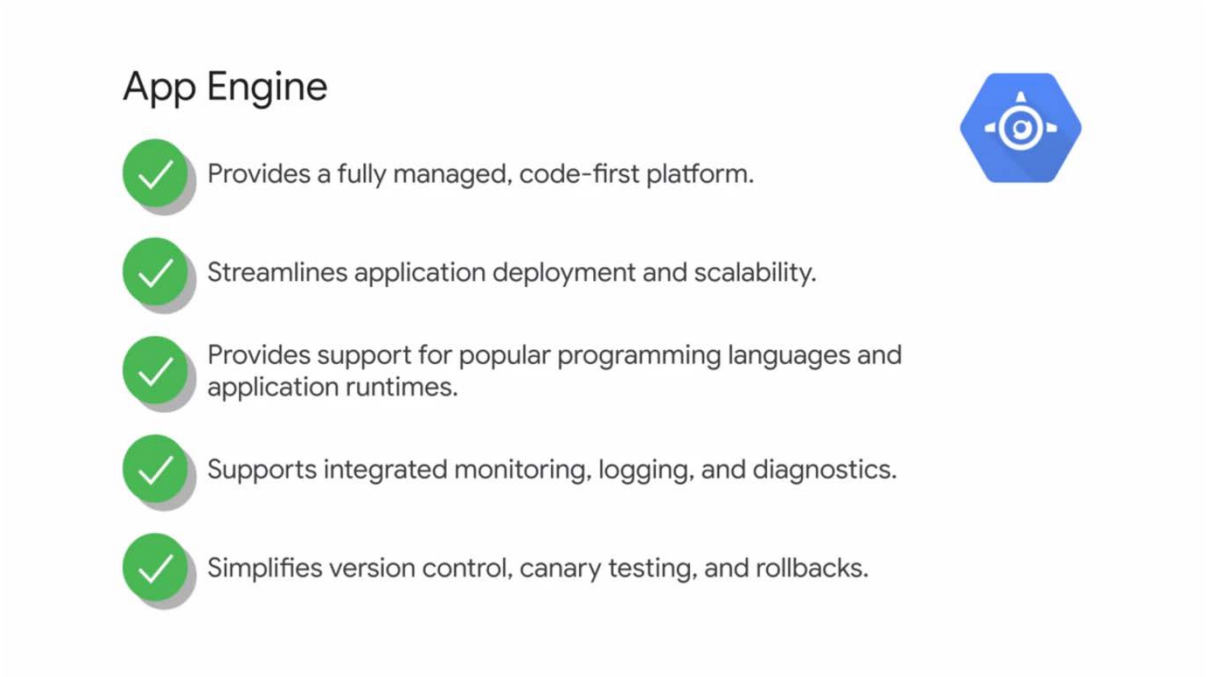
COMPUTE OPTIONS



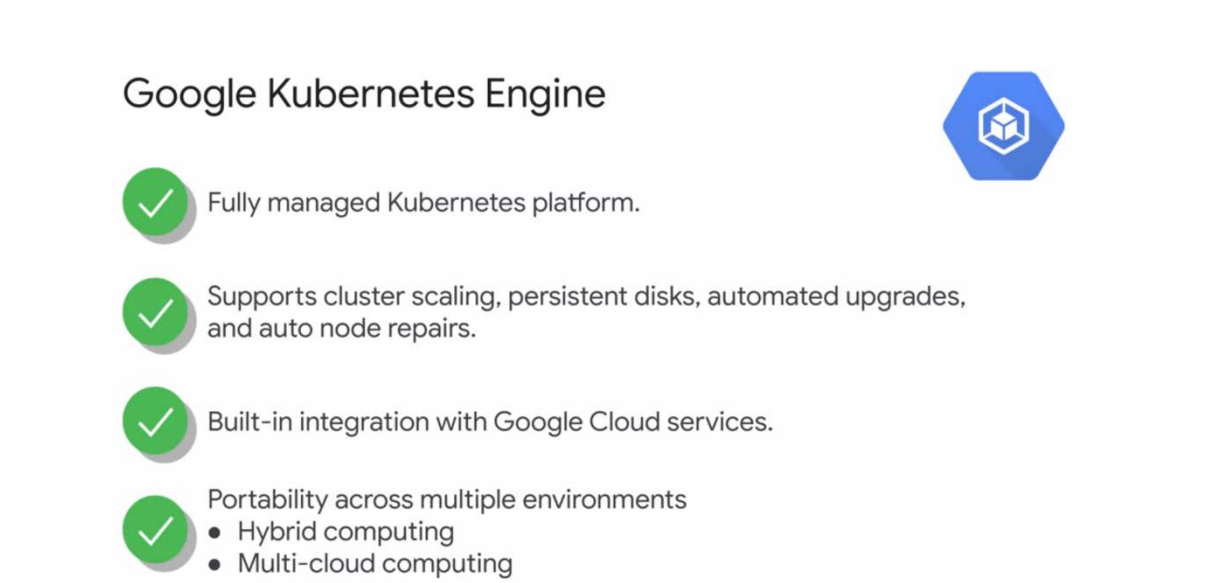


- You can select predefined VM configurations. You could also create customized configurations to precisely match your performance and cost requirements.
- Virtual machines need block storage. Compute Engine offers you two main choices, persistent disks and local SSDs. Persistent disks offer network stores that can scale up to 64 terabytes and
- you can easily take snapshots of these disks for backup and mobility. You could also choose local SSDs which enable very high input/output operations per second.
- You can place your Compute Engine workloads behind global load balancers that support autoscaling.
- Compute Engine offers a feature called **managed instance groups**.With these you can define resources that are automatically deployed to meet demand.
- GCP enables fine grained control of costs of Compute Engine resources by providing per second billing. This granularity helps reduce your costs when deploying compute resources for short periods of time, such as batch processing jobs.
- Compute Engine offers preemptible virtual machines which provide significantly cheaper pricing for your workloads that can be interrupted safely.
- **USE CASES :**
    - With Compute Engine you have complete control over your infrastructure. You can customize operating systems and even run applications that rely on a mix of operating systems.
    - You can easily lift and shift your on-premises workloads into GCP without rewriting your applications or making any changes.

- Compute Engine is the best option when other computing options don't support your applications or requirements.



- App Engine has a completely different orientation from Compute Engine. App Engine is a fully managed application platform. Using App Engine means **zero server management and zero configuration deployments**. So if you're a developer, you can focus on building applications and not really worrying about the deployment part.
- App Engine will deploy that required infrastructure for you. App Engine supports popular languages like Java and Node.js, Python, PHP, C#, .NET, Ruby, and Go.
- You could also run container workloads.
- Stackdriver monitoring, logging, and diagnostics, such as debugging and error reporting are also tightly integrated with App Engine. You can use Stackdriver's real time debugging features to analyze and debug your source code. Stackdriver integrates with tools such as Cloud SDK, cloud source repositories, IntelliJ, Visual Studio, and PowerShell.
- App Engine also supports version control and traffic splitting.
- App Engine is a good choice if you simply want to focus on writing code, and you don't want to worry about building the highly reliable and scalable infrastructure that'll run on. You can just focus on building applications instead of deploying and managing the environment.
- **USE CASES :**
    - websites, mobile apps,
    - gaming backends,
    - and as a way to present a RESTful API to the Internet. (it's an application program interface that resembles the way a web browser interacts with the web server. RESTful APIs are easy for developers to work with and extend.)



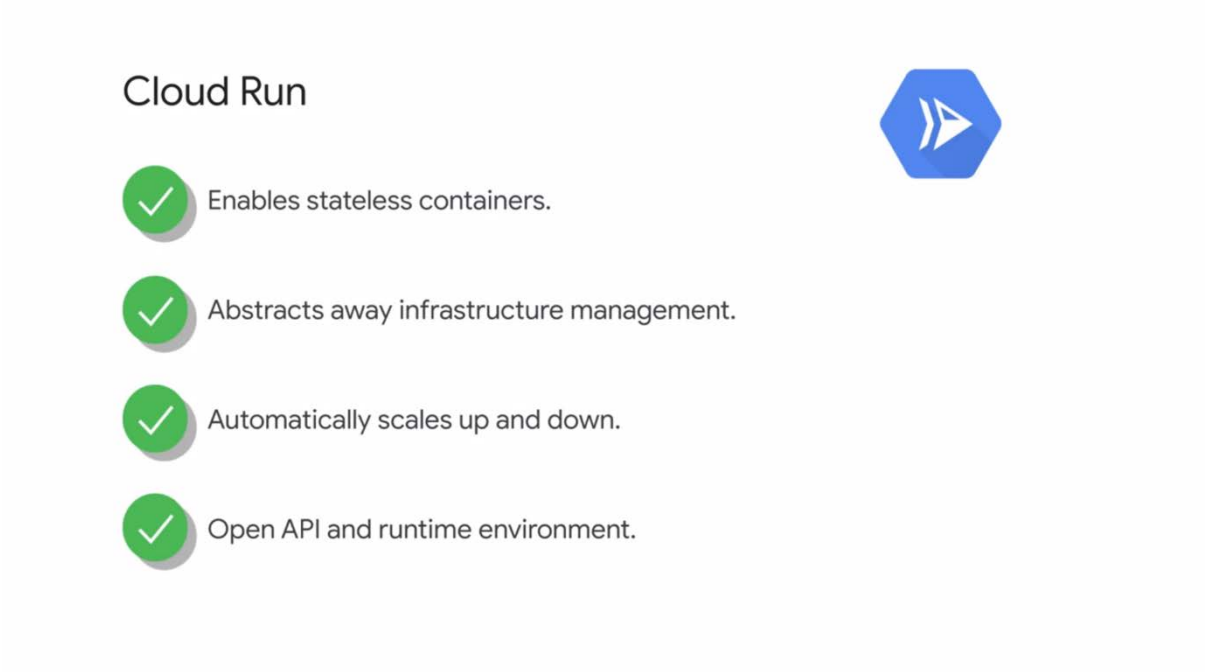
- Finally, the main topic of this course, Google Kubernetes Engine. We learned that Kubernetes is an orchestration system for applications in containers. It automates deployment, scaling, load balancing, logging, and monitoring, and other management features.

- Google Kubernetes Engine extends Kubernetes management on GCP by adding features and integrating with other GCP services automatically. GKE supports cluster scaling, persistent disks, automated updates to the latest version of Kubernetes, and auto repair for unhealthy nodes.

- It has built-in integration with cloud build, container registry, Stackdriver monitoring, and Stackdriver logging.

- Existing workloads running within on-premise clusters can easily be moved on to GCP. There's no vendor login.

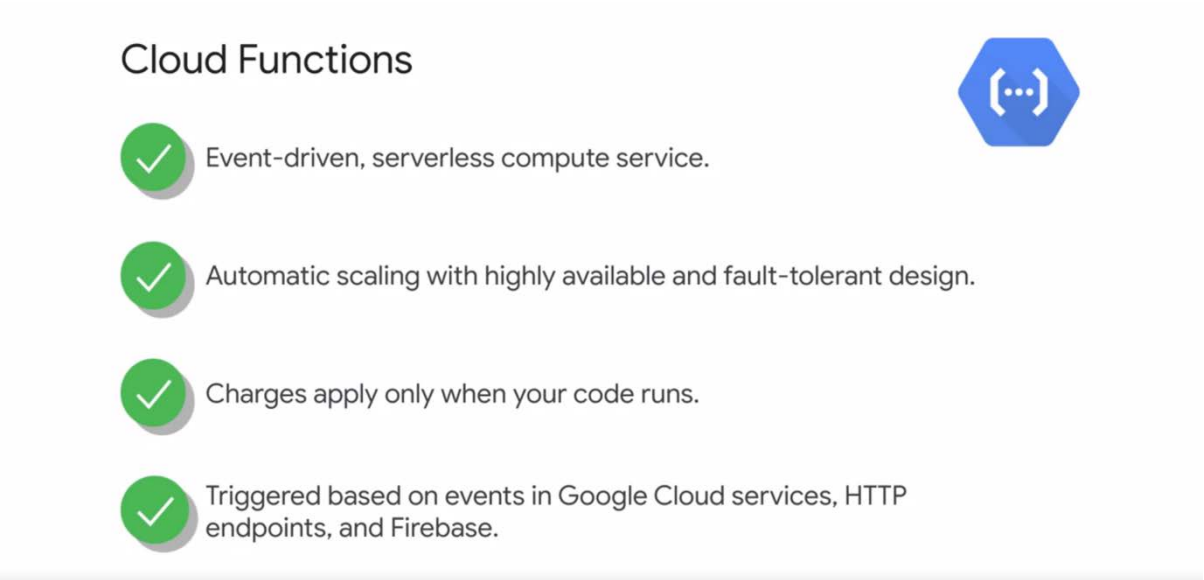
- Overall, GKE is very well suited for containerized applications. Cloud-native distributed systems and hybrid applications. Web requests, or cloud Pub/Sub events.



- Cloud Run is serverless, it distracts way all the infrastructure management so you can focus on developing applications.
- It's built on Knative, an open source Kubernetes based platform. It builds, deploys, and manages modern stateless workloads.
- Cloud Run gives you the choice of running your containers easier with fully managed or in your own GKE cluster.
- Cloud Run enables you to run request or event driven stateless workloads without having to worry about servers.
- It abstracts away all the infrastructure management such as provisioning, configuring, managing those servers so you can focus on just writing code.
- It automatically scales up and down from zero depending upon traffic almost instantaneously, so you never have to worry about scale configuration.
- Cloud Run charges you for only the resources that you use calculated down to the nearest 100 milliseconds. So you don't have to pay for those over provisioned resources.
- USE CASES :
    - With Cloud Run you can choose to deploy your stateless containers with a consistent developer experience to a fully managed environment or to your own GKE cluster.

      This common experience is enabled by Knative an open API and runtime environment built on top of Kubernetes. And it gives you the freedom to move your workloads across different environments and platforms, either fully managed on GCP, on GKE or anywhere a Knative runs.
    - Cloud Run enables you to deploy stateless containers that listen for requests or events delivered via HTTP requests.

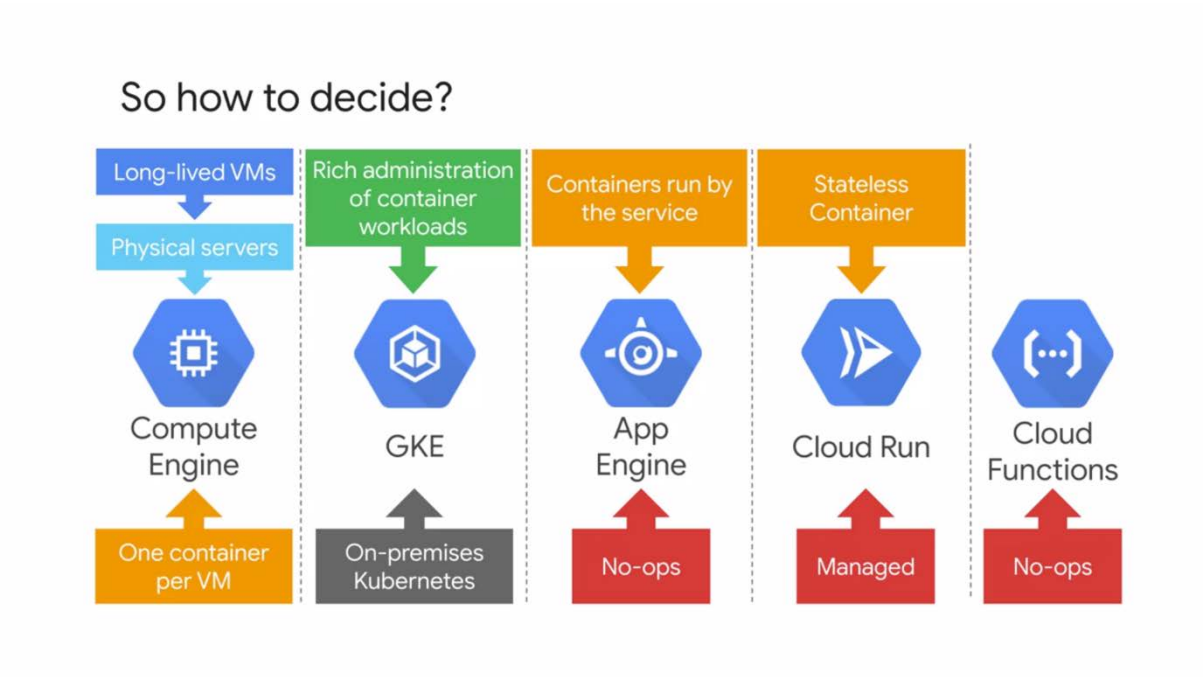
- With Cloud Run, you can build your applications in any language using whatever frameworks and tools you wish and deploy them in seconds without having to manage and maintain that server infrastructure.



- Cloud Functions is an event-driven, serverless compute service for simple single purpose functions that are attached to events. In Cloud Functions, you simply upload your code written in JavaScript or Python, or Go and then GCP will automatically deploy the appropriate computing capacity to run that code.
- These servers are automatically scaled and are deployed from highly available and a fault-tolerant design.
- You're only charged for the time that your code runs. For each function, invocation memory and CPU use is measured in the 100 millisecond increments, rounded up to the nearest increment.
- Cloud Functions also provides a perpetual free tier. So many cloud function use cases could be free of charge.
- With Cloud Functions, your code is triggered within a few milliseconds based on events.
- For example, a file is uploaded to Google cloud storage or a message is received from Cloud Pub/Sub.
- Cloud Functions can also be triggered based on HTTP endpoints that you define, and events in the fire based mobile application back end.
- **USE CASES :**
    - They're generally used as part of a microservices application architecture. You can also build symbols, serverless, mobile IoT backends, or integrate with third party services and APIs.
    - Files uploaded into your GCS bucket can be processed in real time. Similarly, the data can be extracted, transformed and loaded for querying in analysis.

- GCP customers often use Cloud Functions as part of intelligent applications, such as virtual assistance, video or image analysis, and sentiment analysis.

CHOOSING A SERVICE :



- If you're running applications on physical server hardware, it will be the path of least resistance to move into compute engine.
- What if you're running applications in long-lived virtual machines in which each VM is managed and maintained? In this case, you'll also find moving to compute engine is the quickest
- GCP services for getting your applications to the cloud. What do you want to to think about operations at all? Well, App Engine and Cloud Functions are good choices.
- Containerization is the most efficient, importable way to package you an application. The popularity of containerization is growing very fast. In fact, both Compute Engine and App Engine can launch containers for you.

  Compute Engine will accept the container image from you and launch a virtual machine instance that contains it.

  You can use Compute Engine technologies to scale and manage the resulting VM. And App Engine flexible environment will accept the container image from you and then run it with the same No-ops environment that App Engine delivers for code.

  But what if you want more control over your containerized workloads than what App Engine offers? And denser packing than what Compute Engine offers? That increasingly popular use case is what GKE is designed to address.

The Kubernetes paradigm of container orchestration is incredibly powerful, and its vendor neutral, and a broad and vibrant community is developed all around it.

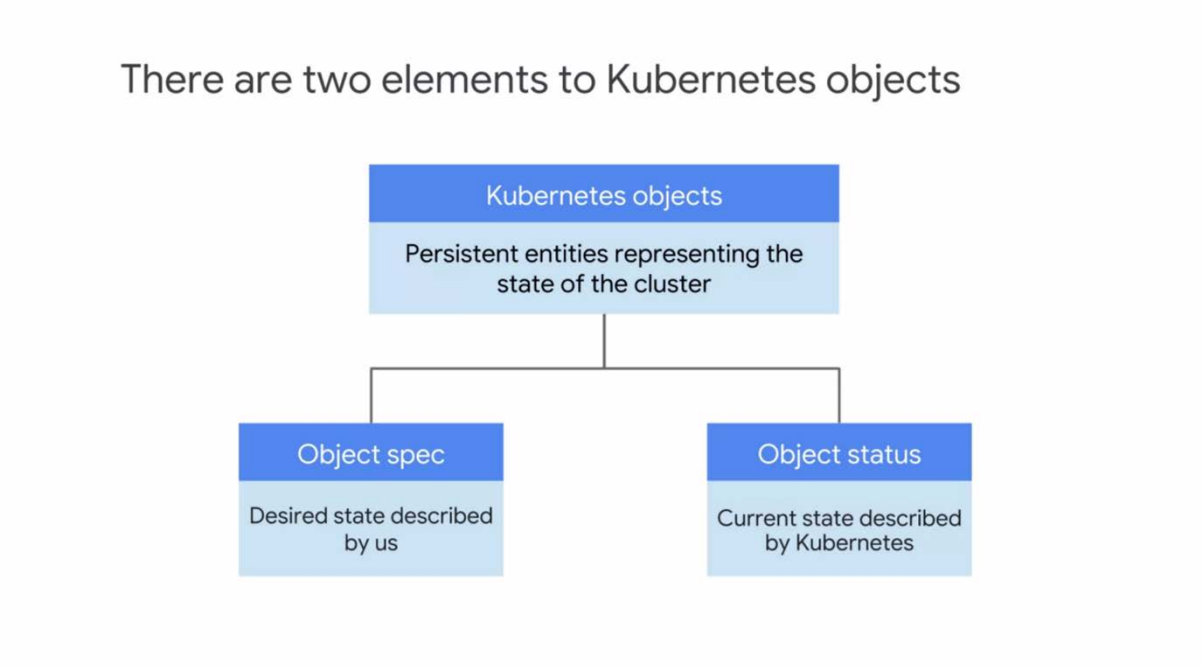
Using Kubernetes as a managed service from GCP saves you work and let's you benefit from all the other GCP resources too.

- You can also choose Cloud Run to run stateless containers on a managed compute platform.

- And of course, if you're already running Kubernetes in your on-premises data centers, moving your GKE is a great choice. Because you'll be able to bring along both your workloads and your management approach.

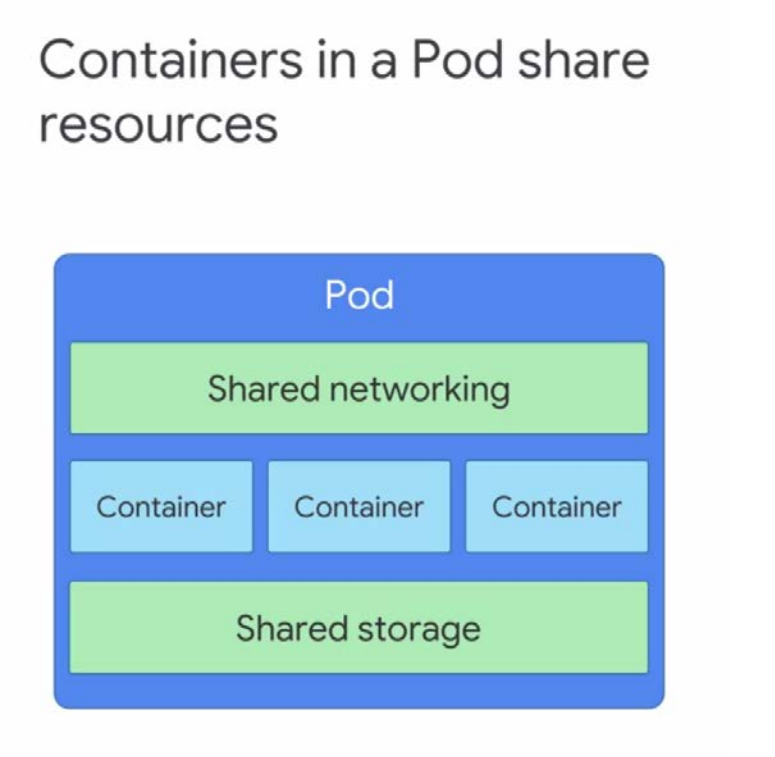
K8S OBJECT MODEL AND PRINCIPLE OF DECLARATIVE MANAGEMENT

- **K8S Object Model** : Each thing Kubernetes manages is represented by an object. You can view and change these objects, attributes, and state.
- **Principle of declarative management** : Kubernetes expects you to tell it what you want, the state of the objects under each management to be. It will work to bring that state into being and keep it there.



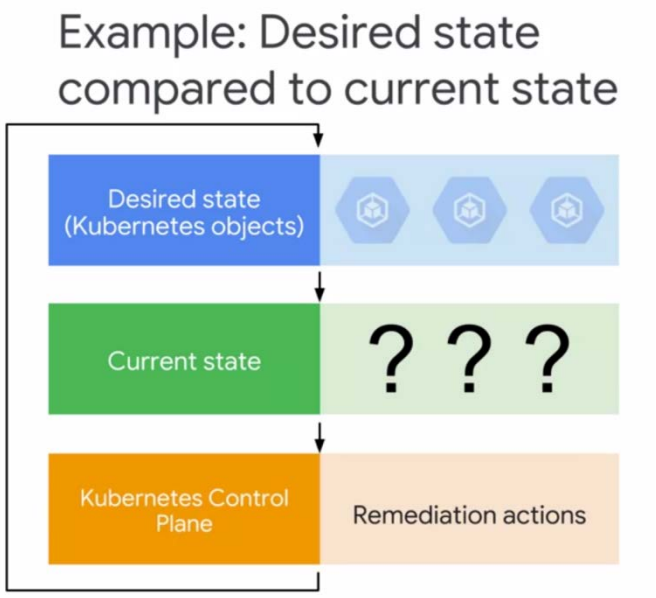
- Formally, a Kubernetes object is defined as a persistent entity that represents the state of something running in a cluster, it's desired state and its current state.
- Various kinds of objects represent containerized applications, the resources that are available to them, and the policies that affect their behavior.
- Kubernetes objects have two important elements.
    - **Objects spec** - for each object you wanted to create. With this spec, you define the desired state of the object by providing the characteristics that you want.
    - **Object's status** - is simply the current state of the object provided by the Kubernetes control plane.
- **K8S Control Plane** refers to the various system processes that collaborate to make a Kubernetes cluster work.

- Each object is of a certain type or kind, as Kubernetes calls them.
- **Pods** are the basic building block of the standard Kubernetes model, and they're the smallest deployable Kubernetes object.



- Not so, every running container and Kubernetes system is in a pod.
- A pod embodies the environment where the containers live. That environment can accommodate one or more containers. If there is more than one container in a pod, they are tightly coupled and share resources including networking and storage.
- Kubernetes assigns each pod a unique IP address. Every container within a pod shares the network namespace, including IP address and network ports.
- Containers within the same pod can communicate through localhost.
- A pod can also specify a set of Storage volumes to be shared among its containers.

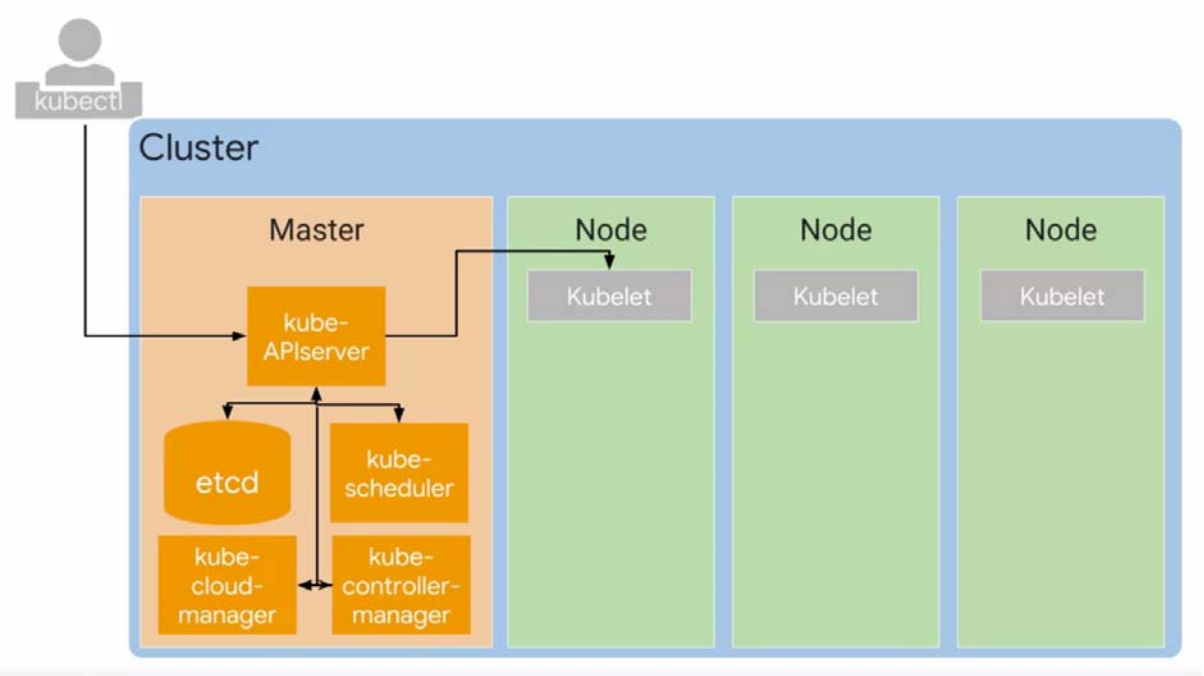
For Example :



- Let's consider a simple example where you want three instances of the NginX Web server,

- each in its own container, running all the time. How is this achieved in Kubernetes?
- Kubernetes embodies the principle of declarative management.
- You declare some objects to represent those NginX containers. What kind of object? Perhaps pods.
- Now it is Kubernetes job to launch those pods and keep them in existence. But be careful, pods are not self healing. If we want to keep all our NginX Web servers not just in existence, but also working together as a team, we might want to ask for them using a more sophisticated object.
- Let's suppose that we have given Kubernetes a desired state that consists of three NginX pods always kept running. We did this by telling Kubernetes to create and maintain one or more objects that represent them.
- Now, Kubernetes compares the desired state to the current state.
- Let's imagine that our declaration of three NginX containers is completely new. The current state does not match the desired state. Kubernetes, specifically it's control plane, will remedy the situation because the number of desired pods running we declared as three and zero while presently running, three will be launched.
- The Kubernetes control plane will continuously monitor the state of the cluster, endlessly comparing reality to what has been declared and remedying the state as needed.

K8S CONTROL PLANE



Kubernetes control plane : the fleet of cooperating processes that make a Kubernetes cluster work.

COMPUTERS :

First and foremost, your cluster needs **computers**. Nowadays, the computers that compose your clusters are usually virtual machines. They always are in GKE, but they could be physical computers too.
One computer is called the **master** and the others are called simply, **nodes**. The job of the nodes is to run pods. The job of the master is to coordinate the entire cluster.

MASTER : Several critical Kubernetes components run on the master.

1. **kube-API server**

- The single component that you interact with directly is the **kube-APIserver**. This component's job is to accept commands that view or change the state of the cluster,including launching pods.
- You will use the **kubectl** command frequently. This command's job is to connect to
- kube-APIserver and communicate with it using the Kubernetes API. Kube-API server also authenticates incoming requests, determines whether they are authorized, invalid, and manages admission control.
- But it's not just kubectl that talks with kube-APIserver. In fact, any query or change to the cluster state must be addressed to the kube-APIserver. \

2. **Etcd** is the cluster's database. Its job is to reliably store the state of the cluster. This includes all the cluster configuration data and more dynamic information such as what nodes are part of the cluster, what pods should be running, and where they should be running.

   You never interact directly with etcd. Instead, kube-APIserver interacts with the database on behalf of the rest of the system.

3. **Kube-scheduler**

- Responsible for scheduling pods onto the nodes. To do that, it evaluates the requirements of each individual pod and selects which node is most suitable.
- But it doesn't do the work of actually launching pods onto nodes. Instead, whenever it discovers a pod object that doesn't yet have an assignment to a node, it chooses a node and simply write the name of that node into the pod object.
- **kube-scheduler** It knows the state of all the nodes, and it will also obey constraints that you define on where a pod may run, based on hardware, software, and policy.
- For example, you might specify that a certain pod is only allowed to run on nodes with a certain amount of memory. You can also define affinity specifications, which cause groups of pods to prefer running on the same node. Or anti-affinity specifications which ensure that pods do not run on the same node.

4. **Kube-controller manager** has a broader job.

- It continuously monitors the state of a cluster through kube-APIserver. Whenever the current state of the cluster doesn't match the desired state, kube-controller manager will attempt to make changes, to achieve the desired state.

- It's called the controller manager because many Kubernetes objects are maintained by loops of code called controllers. These loops of code handle the process of remediation. Controllers will be very useful to you.

- To be specific, you all use certain kinds of Kubernetes controllers to manage workloads.

- For example : problem of keeping three engine x pods always running. We can gather them together into a controller object called a deployment, that not only keeps them running, but also lets us scale them and bring them together underneath our front end.

- Other kinds of controllers have system-level responsibilities.

- For example, **node controller's** job is to monitor and respond when a node is offline.

- **Kube-cloud-manager** manages controllers that interact with underlying cloud providers. For example, if you manually launched a Kubernetes cluster on Google Compute Engine, kube-cloud-manager would be responsible for bringing in GCP features like load balancers and storage volumes when you needed them.

5. **Kubelet**

- Each node runs a small family of control-plane components too. For example, each node runs a **kubelet.** You can think of kubelet as Kubernetes agent on each node.

- When the kube-APIserver wants to start a pod on a node, it connects to that node's kubelet.

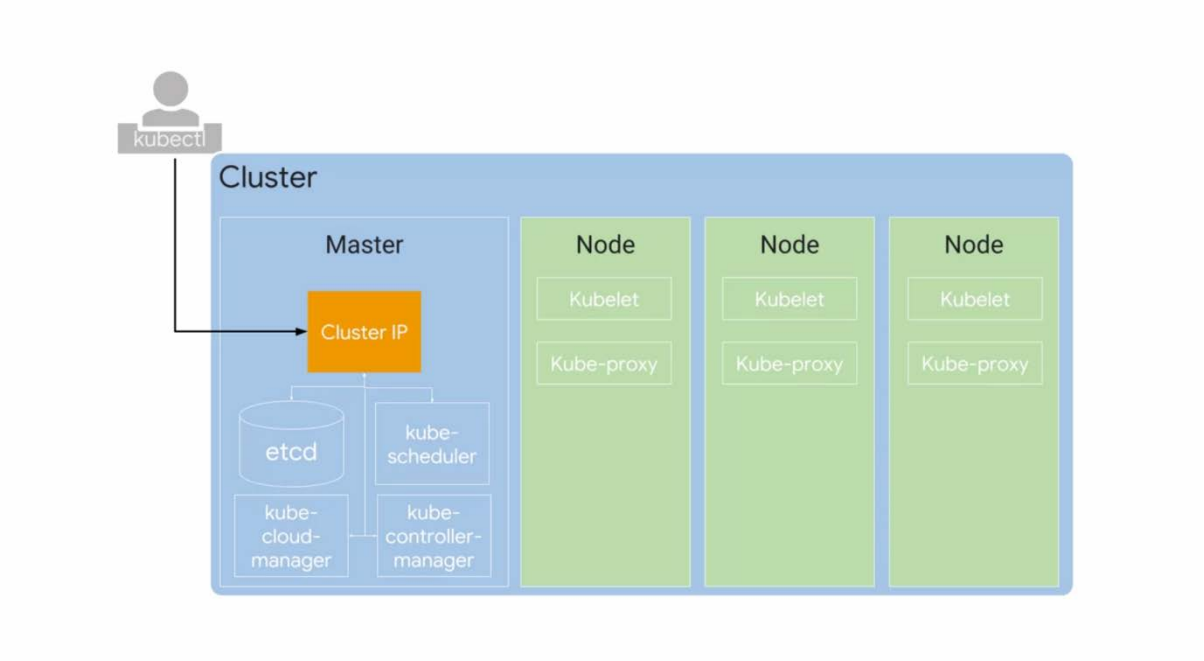
- Kubelet uses the container runtime to start the pod and monitor its lifecycle, including readiness and liveness probes, and reports back to kube-APIserver.

- **Container Runtime** : This is the software that knows how to launch a container from a container image. The world of Kubernetes offers several choices of container runtimes, but the Linux distribution, that GKE uses for its nodes, launches containers using container D. The runtime component of docker.

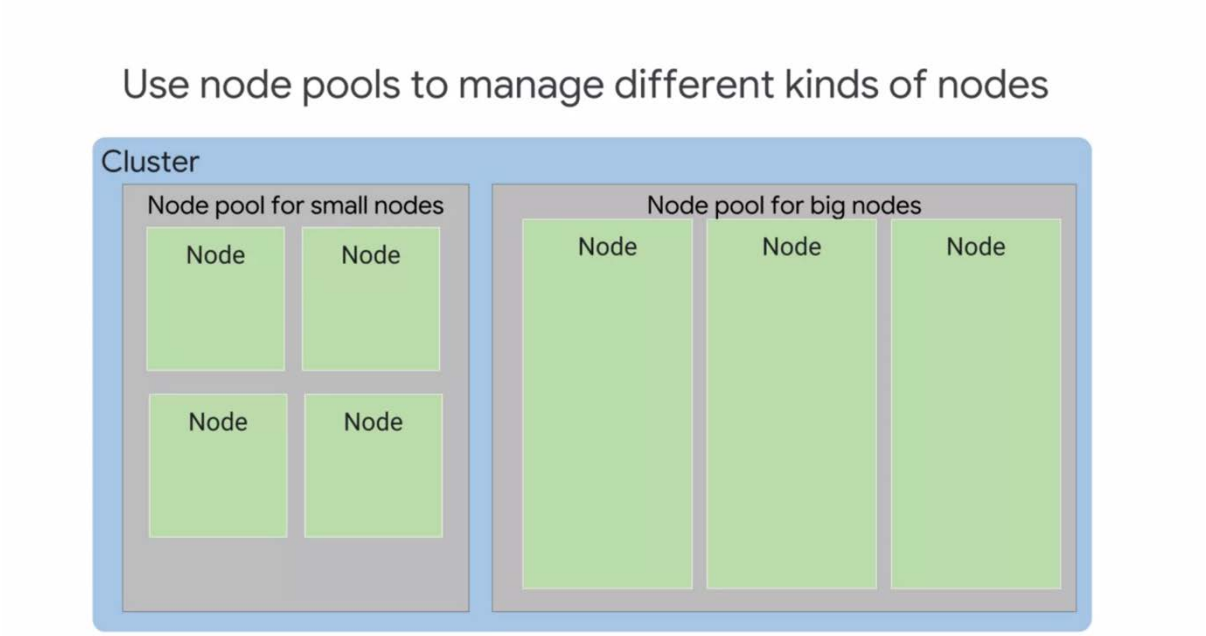
6. **Kube proxy's** job is to maintain network connectivity among the pods in a cluster. In open-source Kubernetes, it does so using the firewalling capabilities of IP tables, which are built into the Linux kernel.

HOW GKE ABSTRACTS K8S PLANE MGMT

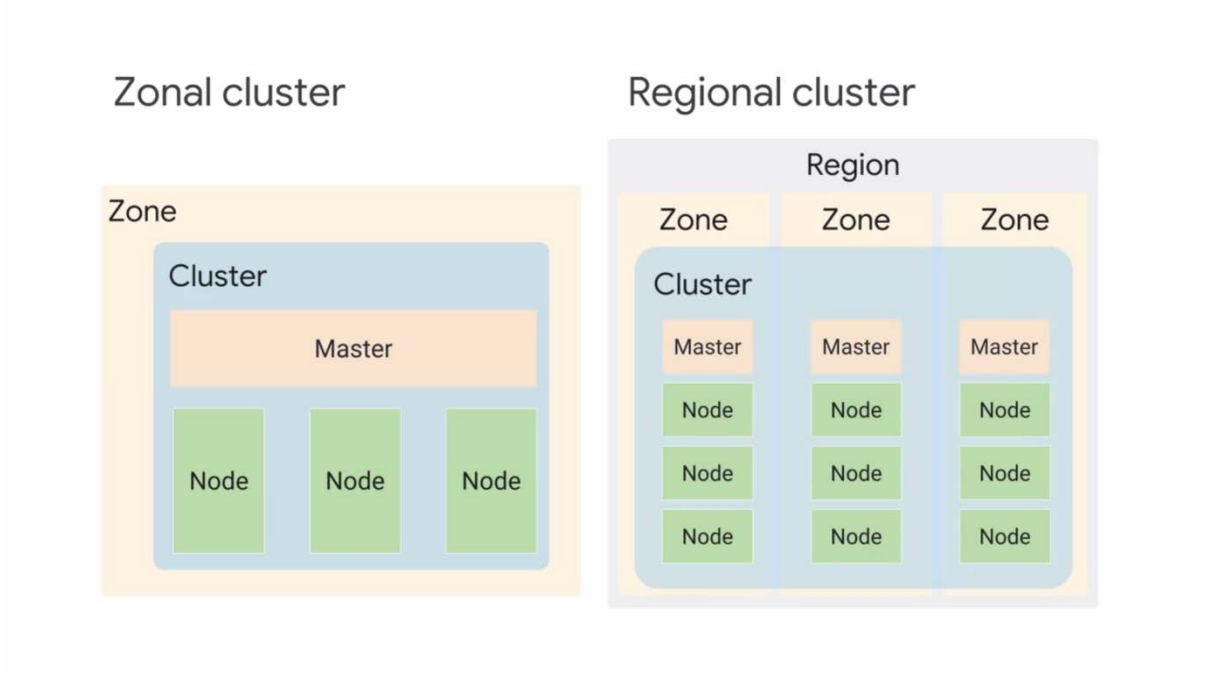
- Setting up a Kubernetes cluster by hand is tons of work. Fortunately, there's an open-source command called cube ADM that can automate much of the initial setup of a cluster.
- But if a node fails or needs maintenance, human administrator has to respond manually.
- When using GKE, from the user's perspective, it's a lot simpler.



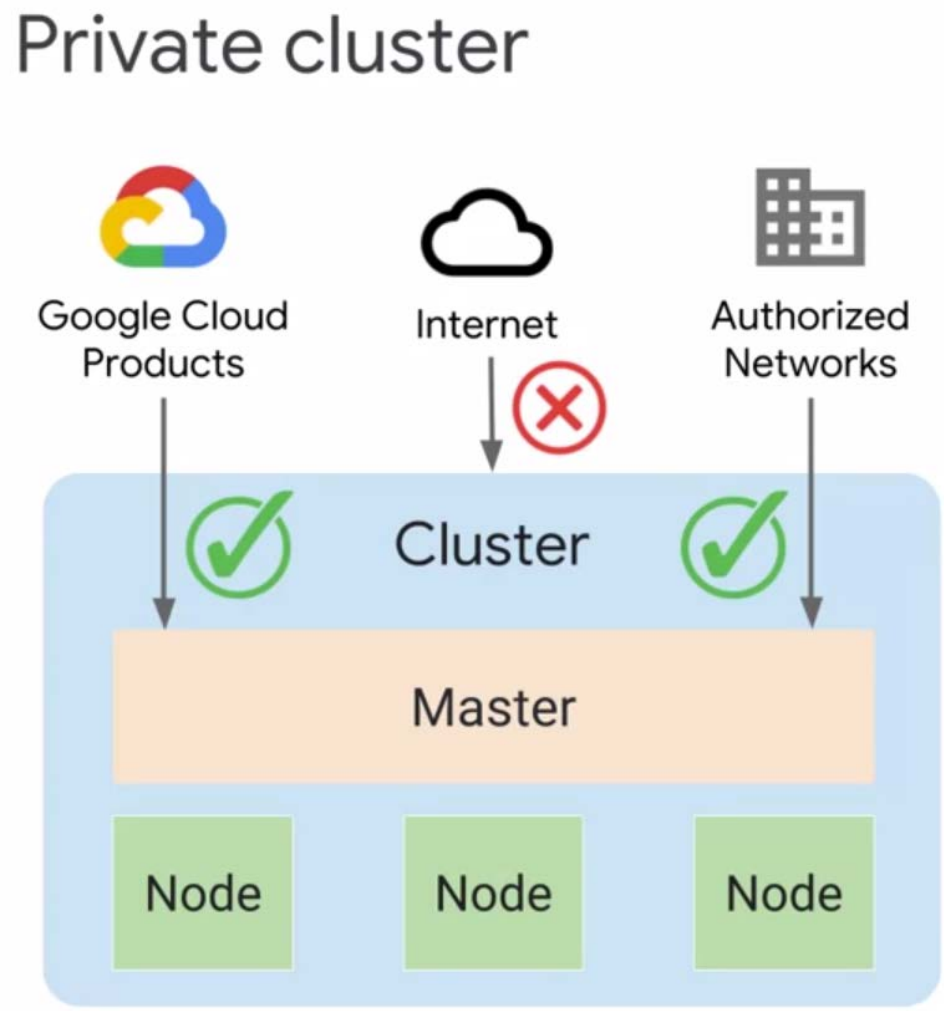
- GKE manages all the control-plane components for us. It's still exposes an IP address to which we send all of our Kubernetes API requests.
- But GKE takes responsibility for **provisioning and managing all the master infrastructure** behind it. It also abstracts away having a separate master. The responsibilities of the master are absorbed by GCP, and you are not separately billed for your Master.
- In any Kubernetes environment, nodes are created externally by cluster administrators,
- not by Kubernetes itself. GKE automates this process for you. It launches Compute Engine virtual machine instances and registers them as nodes.
- You can manage node settings directly from the GCP console. You pay per hour of life of your nodes, not counting the master.
- Because nodes run on Compute Engine, you choose your node machine type when you create your cluster. By default, the node machine type is N1 standard one, which provides one virtual CPU and 3.75 gigabytes of memory.
- Google Cloud offers a wide variety of Compute Engine options.
- You can customize your nodes, number of cores, and their memory capacity. You can select a CPU Platform. You can choose a baseline minimum CPU platform for the nodes or **node pool.**
- This allows you to improve node performance.
- GKE will never use a platform that is older than the CPU platform you specify. If it picks a newer platform, the cost will be the same as the specified platform.
- You can also select multiple node machine types by creating multiple node pools. A **node pool** is a subset of nodes within a cluster that share a configuration, such as their amount of memory or their CPU generation.
- Node pools also provide an easy way to ensure that workloads run on the right hardware within your cluster. You just label them with a desired node pool. By the way, node pool are GKE feature rather than a Kubernetes feature. You can build an analogist mechanism within open-source Kubernetes, but you would have to maintain it yourself.
- You can enable automatic node upgrades, automatic node repairs, and cluster auto-scaling at this node pool level.
- Some of each node CPU and memory are needed to run the GKE and Kubernetes components that let it work as part of your cluster. For example, if you allocate nodes with 15 gigabytes of memory, not quite all of that 15 gigabytes will be available for use by pods.
- By **default**, a cluster launches in a single GCP Compute Zone with three identical nodes, all in one node pool.



- The number of nodes can be changed during or after the creation of the cluster.
- Adding more nodes and deploying multiple replicas of an application will improve an applications availability, but only up to a point.
- You can address this concern by using a GKE regional cluster.
- Regional clusters have a single API endpoint for the cluster.
- However, it's masters and nodes are spread out across multiple Compute Engine zones within a region. Regional clusters ensure that the availability of the application is maintained across multiple zones in a single region.
- In addition, the availability of the master is also maintained so that both the application and management functionality can withstand the loss of one or more, but not all zones.
- By default are regional cluster is spread across three zones, each containing one master and three nodes.



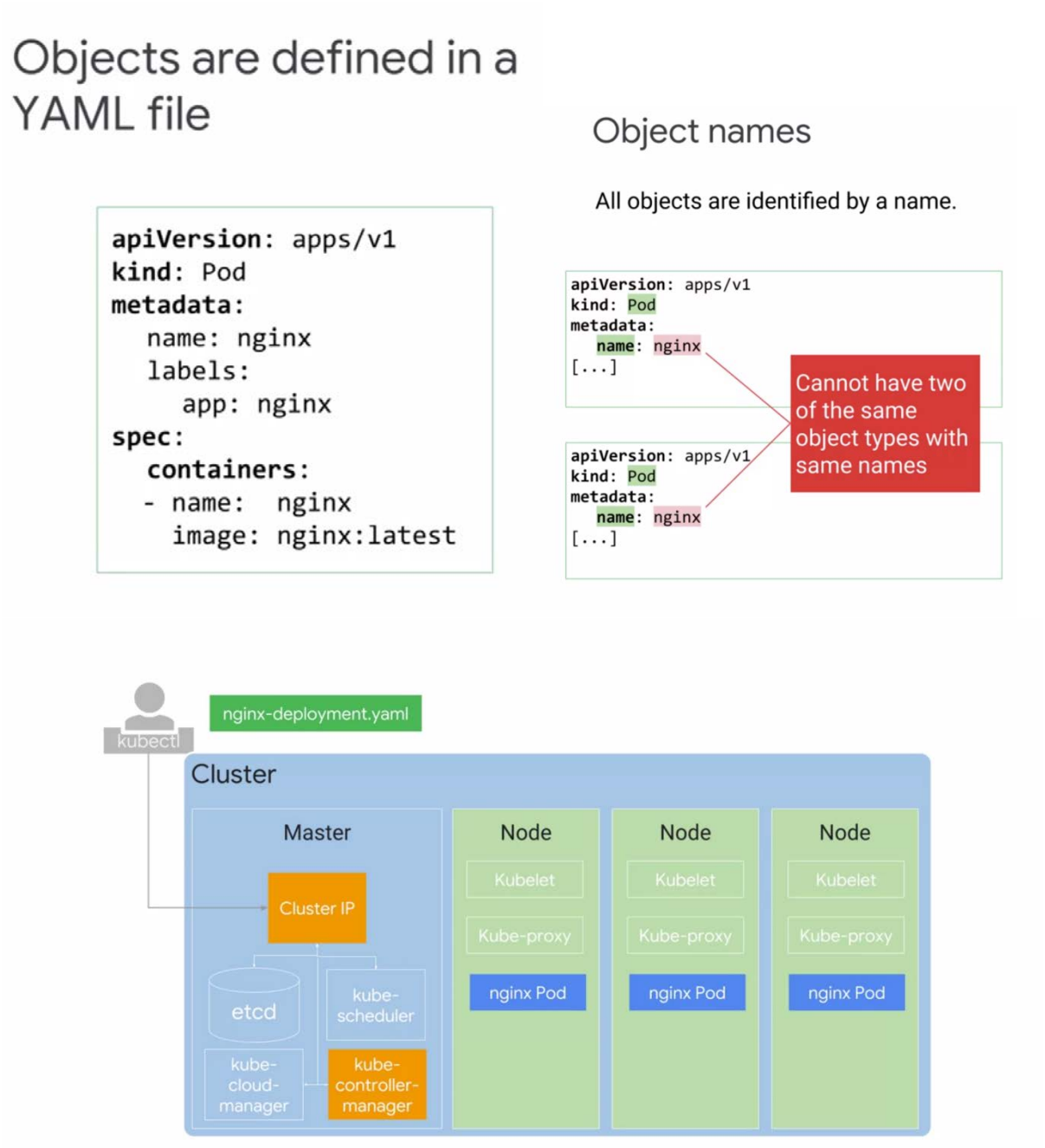
- These numbers can be increased or decreased. For example, if you have five nodes in zone one, you will have exactly the same number of nodes in each of the other zones for a total of 15 nodes.
- Once you build a zonal cluster, you can't convert it into a regional cluster or vice versa.
- Regional and zonal GKE clusters can also be setup as a private cluster. The entire cluster that is the master and it's nodes are hidden from the public Internet.



Cluster masters can be accessed by Google Cloud products such as Stack driver through an internal IP address.
They can also be accessed by authorized networks through an external IP address. Authorize networks are basically IP address ranges that are trusted to access the master.
In addition, nodes can have limited outbound access through private Google access, which allows them to communicate with other GCP services. For example, nodes can pull Container images from Google Container Registry without needing external IP addresses.

K8S OBJECT MGMT

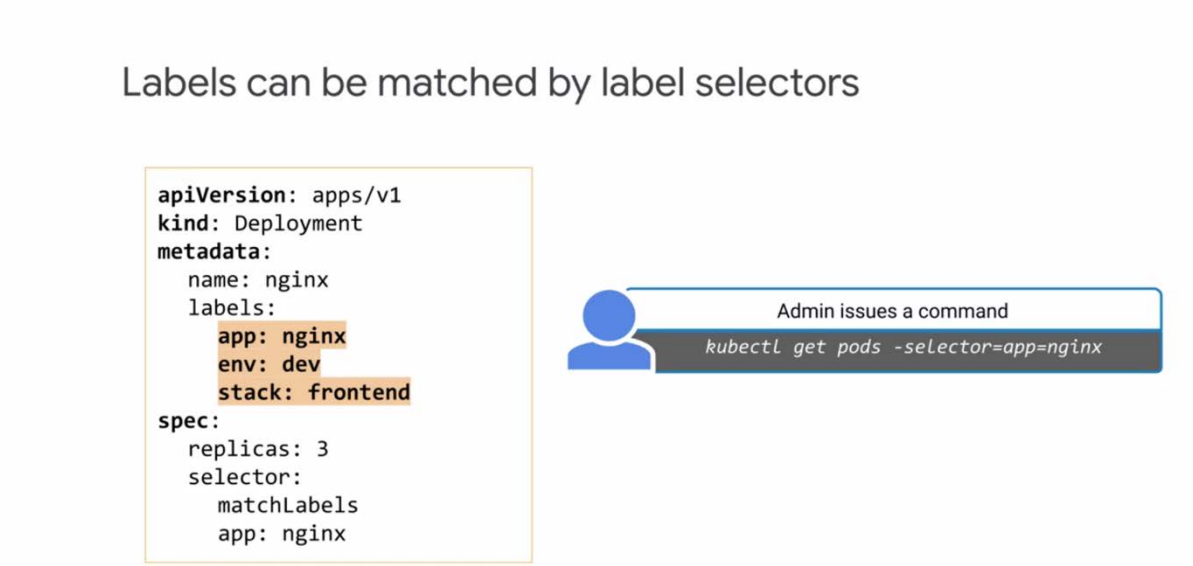
- All Kubernetes objects are identified by a unique name and a unique identifier.
- Understanding this entire concept through an example : We want three nginx web servers running all the time.
- Well, the simplest way would be to declare three pod objects and specify their state.
- For each, a pod must be created and an nginx container image must be used.
- You define the objects you want Kubernetes to create and maintain with **manifest files.**
- These are ordinary text files. You may write them in **YAML or JSON format.**
- YAML is more human-readable and less tedious to edit and we will use it throughout this specialization.



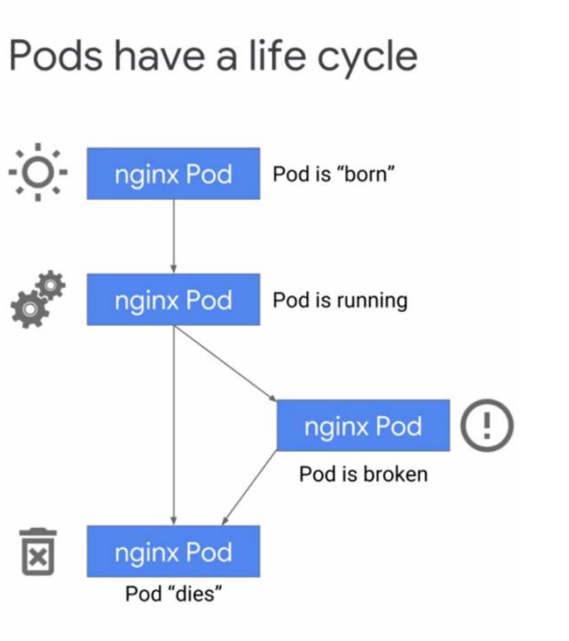
- This YAML file defines a desired state for a pod, its name, and a specific container image for it to run.
- Manifest files have certain required fields.
    - API version describes which Kubernetes API version is used to create the object. The Kubernetes protocol is versioned so as to help maintain backwards compatibility.
    - Kind defines the object you want, in this case a pod.
    - Metadata helps identify the object using name, unique ID, and an optional namespace.
- You can define **several related objects in the same YAML file** and it is a best practice to do so. One file is often easier to manage than several.
- You should save your YAML files in **version controlled** repositories. This practice makes it easier to track and manage changes and to back out those changes when necessary. It's also a big help when you need to recreate or restore a cluster. Many GCP

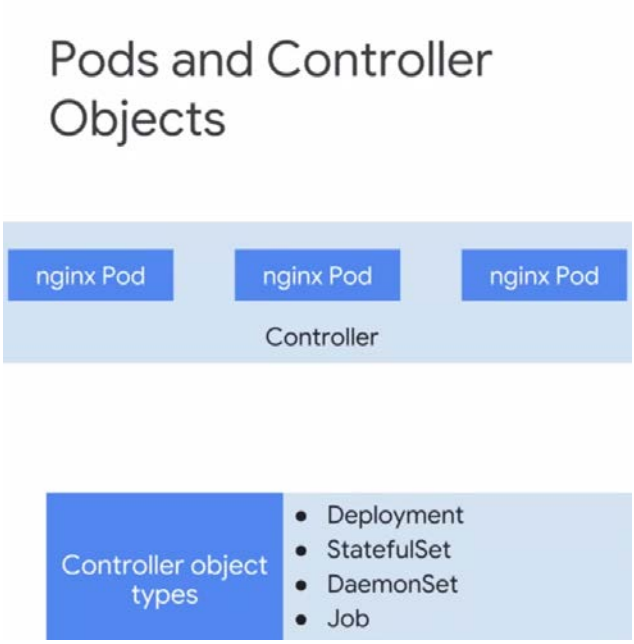
customers use **Cloud Source Repositories** for this purpose because that service lets them control their permissions of those files in the same way as their other GCP resources.

- When you create a Kubernetes object, you name it with a string. Names must be unique.
- Only one object of a particular kind can have a particular name at the same time in a Kubernetes namespace. However, if an object is deleted, it's name can be reused. Alphanumeric characters, hyphens and periods are allowed in the names with a maximum character length of 253.
- Each object generated throughout the life of a cluster has a **unique ID generated by Kubernetes**. This means that no two objects will have the same UID throughout the life of a cluster.
- **Labels** are key value pairs with which you tag your objects during or after their creation. Labels help you identify and organize objects and subsets of objects. For example, you could create a label called app and give as its value, the application of which this object is a part.
- In this simple example, a deployment object is labeled with three different key values. It's application, it's environment, and which stack it forms a part of.

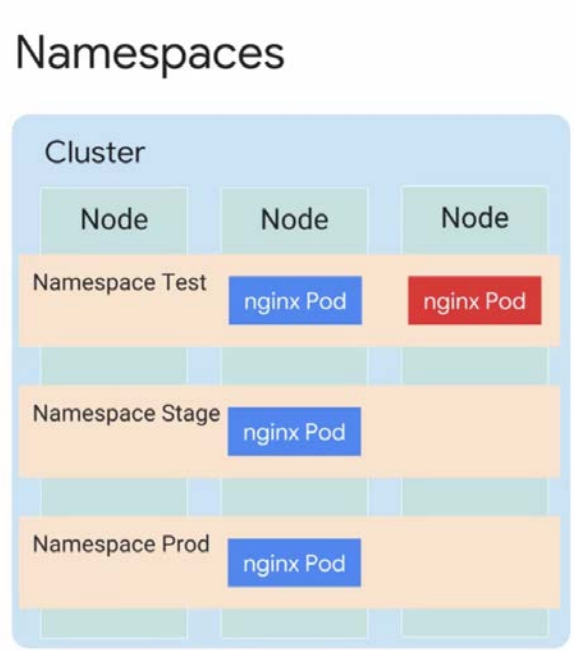


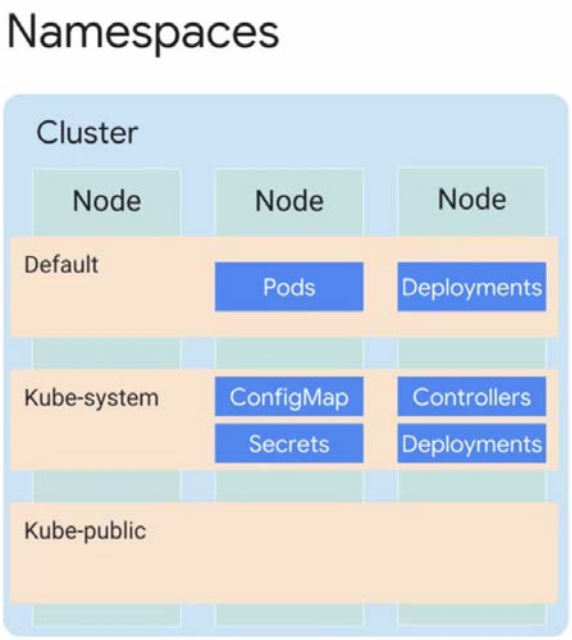
- Various contexts offer ways to select Kubernetes resources by their labels. In this specialization, you will spend plenty of time with the kubectl command. Here's an example of using it to show all the pods that contain a label called app with a value of nginx.
- Label selectors are very expressive. You can ask for all the resources that have a certain value for a label, all those that don't have a certain value, or even all those that have a value in a set you supply.



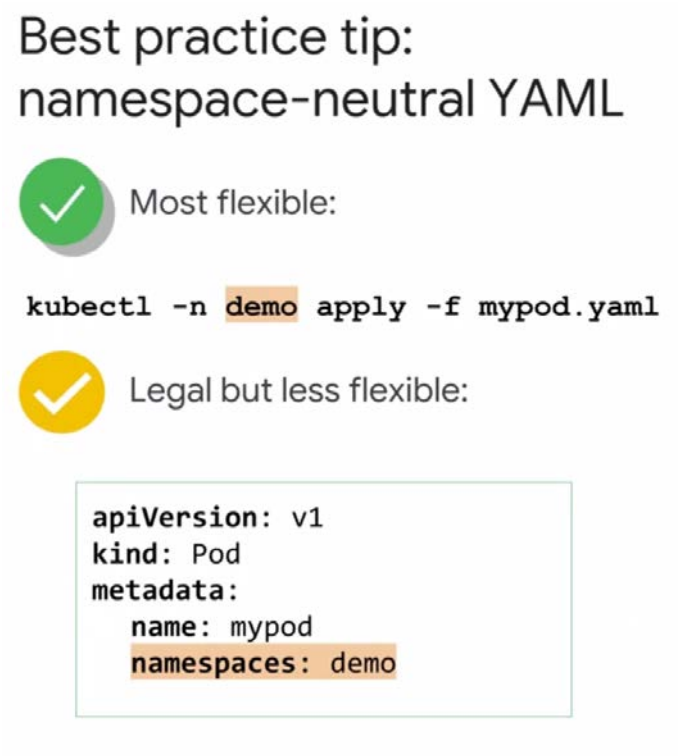


- One way to bring three nginx web servers into being, would be to declare three pod objects, each with its own section of YAML. Kubernetes default scheduling algorithm prefers to spread the workload evenly across the nodes available to it. Suppose I want 200 more nginx instances. Managing 200 more sections of YAML sounds very inconvenient.
- Also, Pods don't heal or repair themselves and they're not meant to run forever. They are designed to be ephemeral and disposable. For these reasons, there are better ways to manage what you run in Kubernetes than specifying individual pods. You need a set up like this to maintain an application's high availability along with horizontal scaling.
- We can instead declare a **controller object** whose job is to manage the state of the pods. Some examples of these objects, **Deployments, StatefulSets, DaemonSets,** and **Jobs.**
- **Deployments** are a great choice for long lives software components, like web servers, especially when we want to manage them as a group. In our example, when **kube-scheduler**, schedules pods for a deployment, it notifies the **kube-IP server**.
- These changes are constantly monitored by controllers, especially by the **deployment controller.**
- The deployment controller will monitor and maintain three nginx Pods. If one of those pods fails, the deployment controller will recognize the difference between the current state and the desired state, and we'll try to fix it by launching a new pod. Instead of using multiple YAML manifests or files for each pod, you used a single deployment YAML to launch three replicas of the same container.
- A deployment ensures that a defined set of pods is running at any given time. Within its objects spec, you specify how many **replica pods** you want, how pods should run, which containers should run within these pods, and which volumes should be mounted.
- Based on these templates, controllers maintain the pods desired state within a cluster. Deployments can also do a lot more than this,
- It's very probable that you'll be using a single cluster for multiple projects (informal meaning of the word). At the same time, it's essential to maintain resource quotas based on projects or teams.
- Each Kubernetes clusters associated with one GCP project in the formal sense of the word Project. That's how IAM policies apply to it and how you're built for it.
- Kubernetes allows you to abstract a single physical cluster into multiple clusters known as namespaces.





- **Namespaces** provides scope for naming resources such as pods, deployments, and controllers.
- As you can see in this example, there are three namespaces in this cluster; test, stage, and prod.
- You cannot have duplicate object names in the same namespace. You can create three pods with the same name in genetics in this case, but only if they don't share the same namespace. If you attempt to create another pod with the same name, nginx Pod in namespace test, you won't be allowed. Object names need only be unique within a namespace, not across all namespaces.
- Namespaces also led to implement resource quotas across the cluster. These quotas defined limits for resource consumption within a namespace. They're not the same as your GCP quotas. These quotas apply specifically to the Kubernetes cluster they are defined on. You're not required to use namespaces for your day to day management.
- You can also use labels. Still namespaces are a valuable tool. Suppose you want to spin up a copy of a deployment as a quick test. Doing so in a new namespace makes it easy and free of name collisions.
- There are three initial namespaces in a cluster.
    - The first is a default namespace, for objects with no other namespace defined. Your Workload resources will use this namespace by default.
    - Then there is the kube-system namespace, for objects created by the Kubernetes system itself.
    - When you use the kubeCTL command, by default, items in the kube-system namespace are excluded, but you can choose to view its contents explicitly.
    - The third namespaces that kube-public namespace, for objects that are publicly readable to all users.
- **Kube-public** is a tool for disseminating information to everything running in a cluster. You're not required to use it, but it can come in handy, especially when everything running in a cluster is related to the same goal and needs information in common.
- You can apply a resource to a namespace when creating it using a command line namespace flag. Or you can specify a namespace in the YAML file for the resource. Whenever possible, apply namespaces at the command line level.
- This practice makes your YAML files more flexible. For example, someday, you might want to create two completely independent instances of one of your deployments, each in its own namespace. This is difficult if you've chosen to embed namespace names in your YAML files.

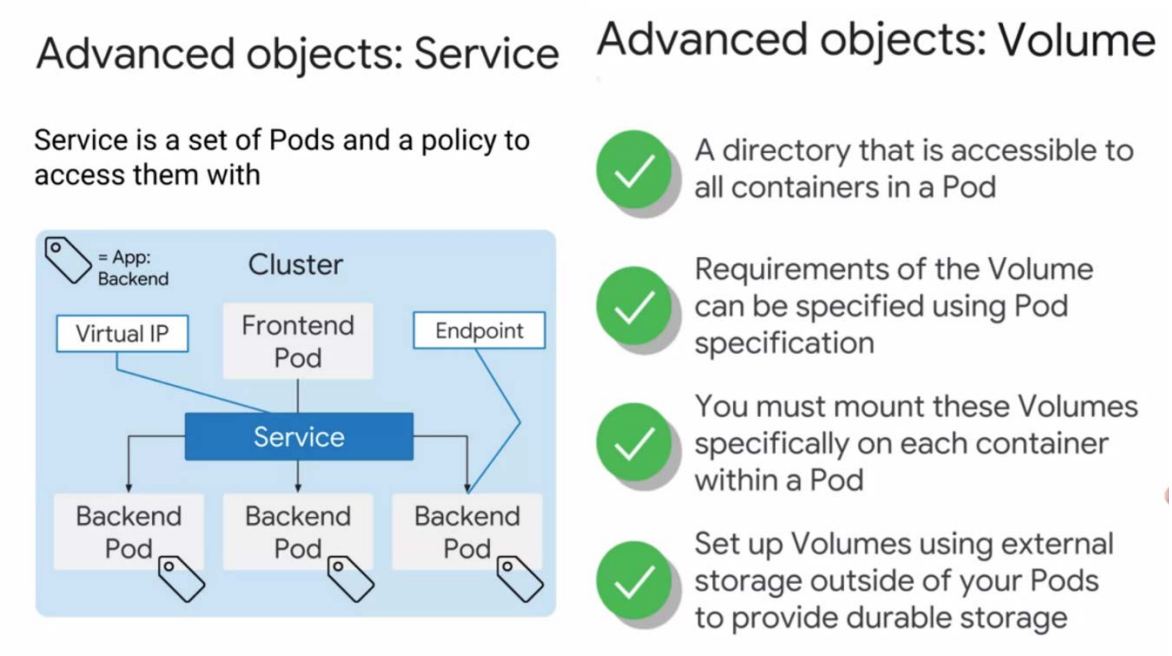


DEPLOYMENTS AND REPLICASETS

The Deployment object would create a ReplicaSet object to manage Pods' replicase.

You will work with Deployment objects directly much more often than ReplicaSet objects. But it's still helpful to know about ReplicaSets, so that you can better understand how Deployments work. For example, one capability of a Deployment is to allow a rolling upgrade of the Pods it manages. To perform the upgrade, the Deployment object will create a second ReplicaSet object, and then increase the number of (upgraded) Pods in the second ReplicaSet while it decreases the number in the first ReplicaSet.

SERVICES AND VOLUMES



Services provide load-balanced access to specified Pods. There are three primary types of Services:

- **ClusterIP**: Exposes the service on an IP address that is only accessible from within this cluster. This is the default type.
- **NodePort**: Exposes the service on the IP address of each node in the cluster, at a specific port number.
- **LoadBalancer**: Exposes the service externally, using a load balancing service provided by a cloud provider.

In Google Kubernetes Engine, LoadBalancers give you access to a regional Network Load Balancing configuration by default. To get access to a global HTTP(S) Load Balancing configuration, you can use an Ingress object.

Remember that Pods are created and destroyed dynamically.
Although Pods can communicate using
their assigned Pod IP addresses,
these IP addresses are ephemeral.
They're not guaranteed to remain constant when Pods are
restarted or when scaling
changes which nodes are used to run Pods.
Imagine you have two sets of Pods,
frontend Pods, and backend Pods.
How will the frontend Pods discover and keep track
of dynamically scaling backend Pods?
This is where the concept of
Kubernetes Services comes in.
A Kubernetes service is a static IP address that
represents a service or
a function in your infrastructure.
It's a network abstraction for

a set of Pods that deliver that service,
and it hides the ephemeral nature of
the IP addresses of the individual Pods.
In the example, a set of backend Pods are
exposed to the frontend Pod using a Kubernetes service.
Basically, the service defines a set of Pods and assigns
a policy by which you can access those Pods.
The Pods are selected using a label selector.
By the way, you can also get a service quickly
by asking Kubernetes to expose a deployment.
When you do that Kubernetes
handles selecting the right Pods for you.
Whenever a service is created,
Kubernetes automatically creates endpoints
for the selected Pods,
by creating endpoint resources.
By default, the master assigns a virtual IP address,
also known as a cluster IP,
to the service from internal IP tables.
With GKE, this is assigned from the clusters VPC network.
You will learn more about services in
a later module in this specialization.
GKE offers other ways your service can be exposed,
not just through cluster IPs.
Overall, a service provides durable endpoints for Pods.
These endpoints can be accessed by
exposing the service internally within a cluster,
or externally to the outside world.
The option to expose a service internally
or externally depends on the service type itself.
The frontend Pod can reliably access
the backend Pods internally
within the cluster using a service.
A Container application can easily write data to
the read-write layer inside
the container, but it's ephemeral.
When the container terminates,
whatever was written will be lost.
What if you want to store data permanently?
Or what if you need storage to be shared
between tightly coupled containers within a Pod?
That's why a Kubernetes volume
is used for more persistent storage.
Kubernetes' volume is another abstraction.
A volume is simply a directory that is
accessible to all the containers in a pod.
The requirements for a volume are
defined through the pods' specification.
This declares how the directory is created,
what storage medium should be used,
and its initial contents.
You don't want failure of containers or
restarts of containers to
affect the data within these volumes,
and you want your volume to be shared
among multiple containers within a Pod.
Docker containers have their own file system.

Therefore, in order to access these volumes,
they must be mounted specifically
on each container within a Pod.
However, Pods themselves are also ephemeral.
A failing node or deleted Pod
could lead to its volume being deleted too.
To avoid this, you can configure
volumes using network-based storage from
outside of your Pods to provide durable storage
that is not lost when a Pod or node fails.
You'll learn about persistent volumes
later in this specialization.

Controller objects to know about

- ReplicaSets
- Deployments
- Replication Controllers
- StatefulSets
- DaemonSets
- Jobs

A **ReplicaSet** controller ensures that a population of Pods, all identical to one another, are running at the same time. Deployments let you do declarative updates to ReplicaSets and Pods. In fact, Deployments manage their own ReplicaSets to achieve the declarative goals you prescribe, so you will most commonly work with Deployment objects.

Deployments let you create, update, roll back, and scale Pods, using ReplicaSets as needed to do so. For example, when you perform a rolling upgrade of a Deployment, the Deployment object creates a second ReplicaSet, and then increases the number of Pods in the new ReplicaSet as it decreases the number of Pods in its original ReplicaSet.

Replication Controllers perform a similar role to the combination of ReplicaSets and Deployments, but their use is no longer recommended. Because Deployments provide a helpful "front end" to ReplicaSets, this training course chiefly focuses on Deployments.

If you need to deploy applications that maintain local state, **StatefulSet** is a better option. A StatefulSet is similar to a Deployment in that the Pods use the same container spec. The Pods created through Deployment are not given persistent identities, however; by contrast, Pods created using StatefulSet have unique persistent identities with stable network identity and persistent disk storage.

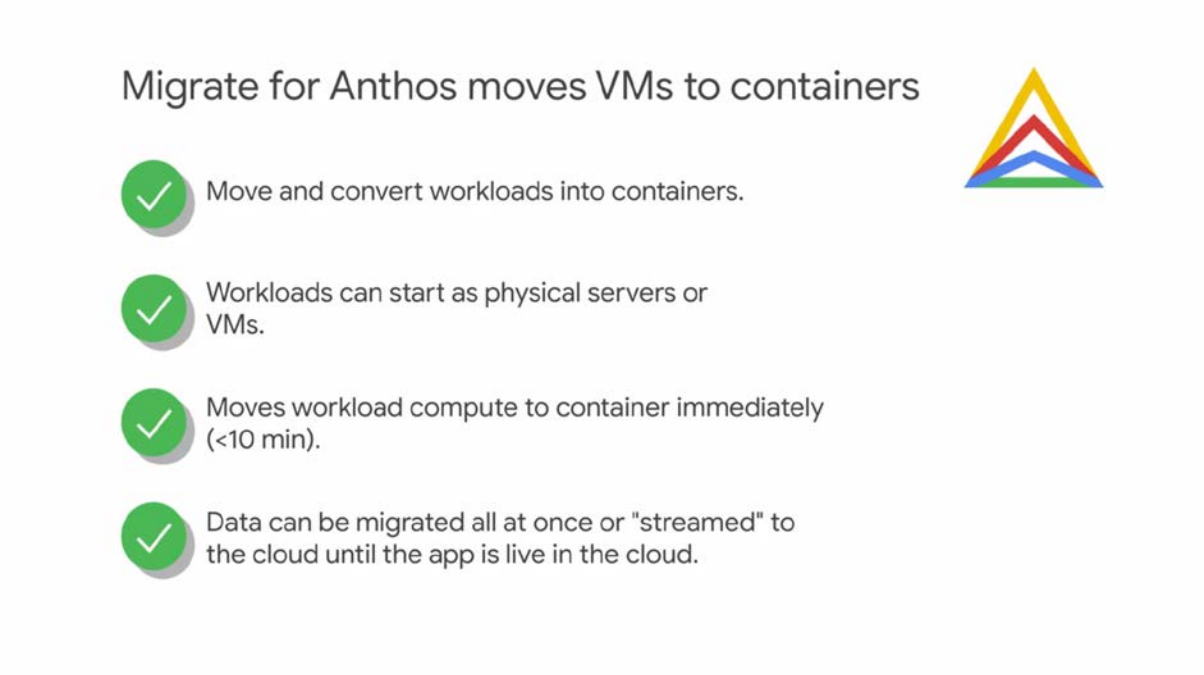
If you need to run certain Pods on all the nodes within the cluster or on a selection of nodes, use **DaemonSet**. DaemonSet ensures that a specific Pod is always running on all or some subset of the nodes. If new nodes are added, DaemonSet will automatically set up Pods in those nodes with the required specification. The word "daemon" is a computer science term meaning a non-interactive process that provides useful services to other processes. A Kubernetes cluster might

use a DaemonSet to ensure that a logging agent like fluentd is running on all nodes in the cluster.

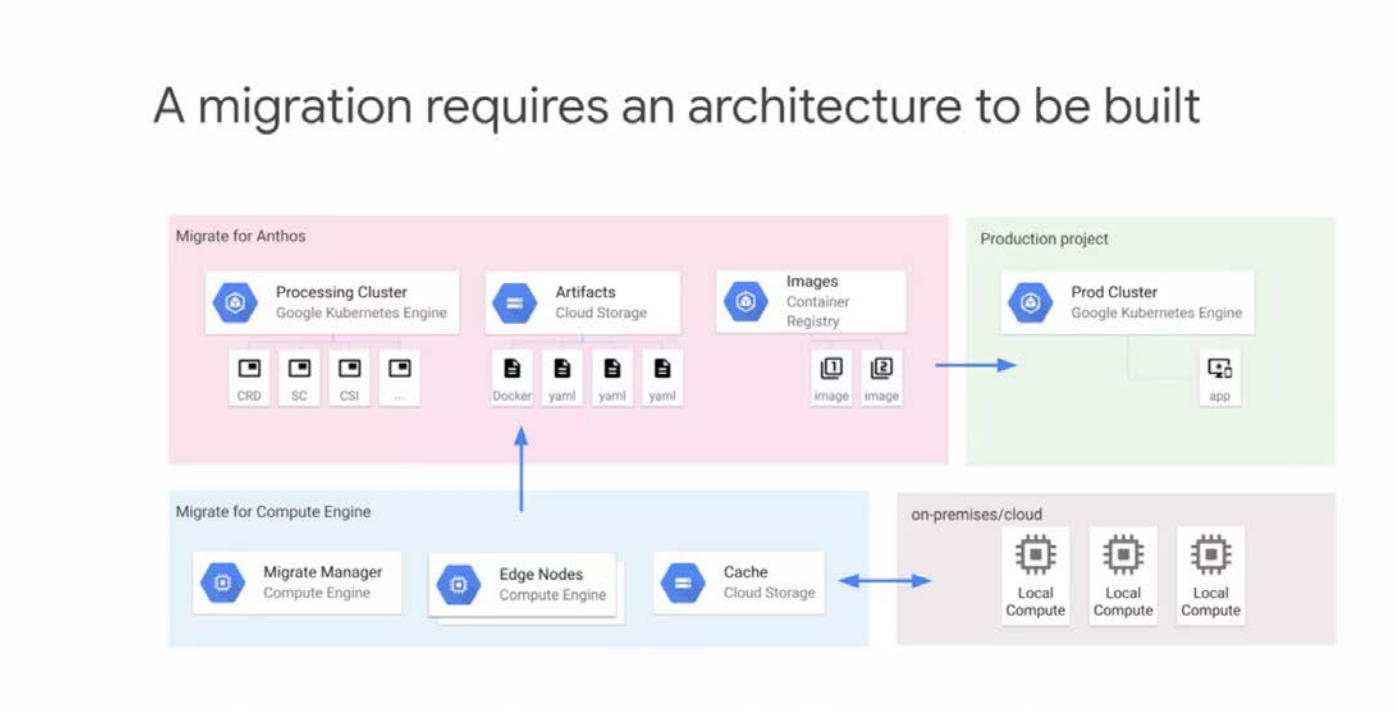
The **Job** controller creates one or more Pods required to run a task. When the task is completed, Job will then terminate all those Pods. A related controller is **CronJob**, which runs Pods on a time-based schedule.

ANTHOS

Migrate for Anthos, is a tool for getting workloads (non-containerized, non-cloud) into containerized deployments on Google Cloud.

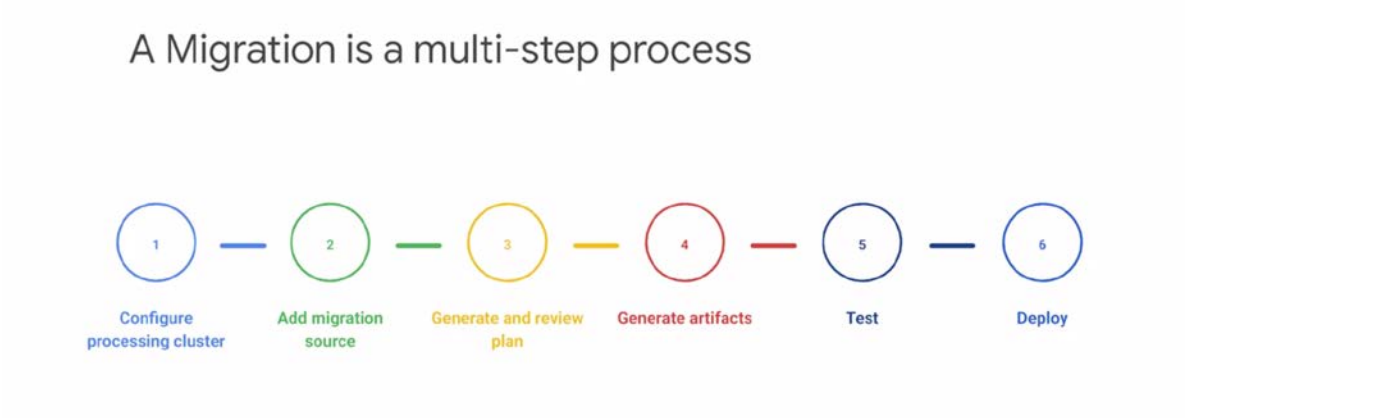


MIGRATE TO ANTHOS – ARCHITECTURE



- The first step is to allow **Migrate for Compute Engine** to create a pipeline for Streaming or Migration the data from on-premises or another cloud provider into Google Cloud.
- **Migrate for Compute Engine** is a tool that allows you to bring your existing applications into VMs on Google Cloud.
- **Migrate from Anthos** is then installed on a **GKE processing cluster** and is composed of many Kubernetes resources.
- Migrate from Anthos is used to generate **Deployment Artifacts**.
- With these artifacts, like your Kubernetes configurations in the Docker File are used
- to create the VM Wrapping Container.
- This Container goes into Cloud Storage.
- The Container Images themselves are stored in the Container Registry.
- After the deployment assets are created, they can be used to deploy your application into a target cluster. You simply apply the generator configuration and it creates all the necessary Kubernetes elements on the target cluster.

MIGRATION PROCESS



- First you need to create the processing cluster.

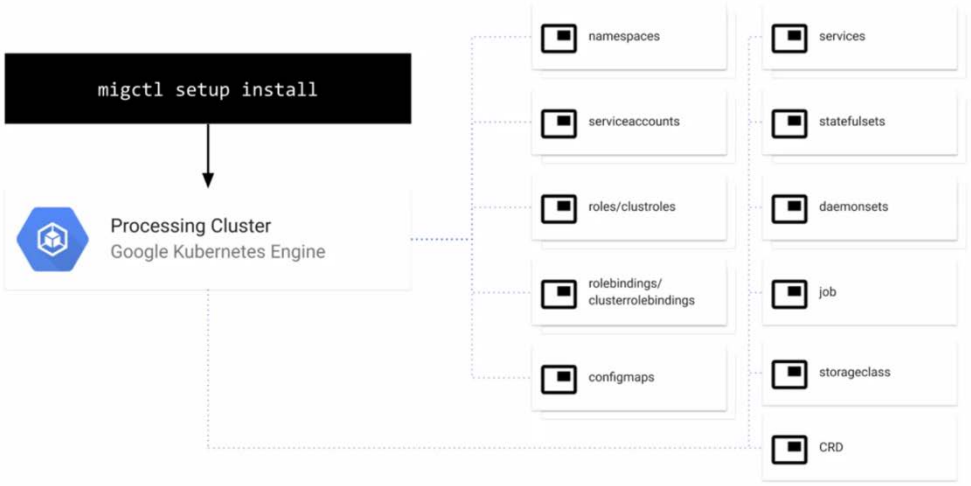
- After that you install the Migrate For Anthos components onto that cluster.Next you need to add a migration source. You can migrate from VMware, AWS, Azure or Google cloud.
- You will need to create a migration object with the details of the migration that you're performing.
- This will generate a plan template for you in a YAML file. You may need to alter this configuration file to create the level of customization that you desire.
- When the plan is ready, you will need to generate the artifacts for the migration. This means generation that container images of your applications on the YAML files required for the deployment.
- After your migration artifacts have been generated, they need to be tested. Both the container images and the deployments will be tested at this stage.
- Finally, if the tests are successful, you can use the generative artifacts to deploy your application in to your production clusters.

MIGRATE FROM ANTHOS

Migrate for Anthos requires a processing cluster

```
gcloud container --project $PROJECT_ID \
clusters create $CLUSTER_NAME \
--zone $CLUSTER_ZONE \
--username "admin" \
--cluster-version 1.14 \
--machine-type "n1-standard-4" \
--image-type "UBUNTU" \
--num-nodes 1 \
--enable-stackdriver-kubernetes \
--scopes "cloud-platform" \
--enable-ip-alias \
--tags="http-server"
```

Installing Migrate for Anthos uses `migctl`

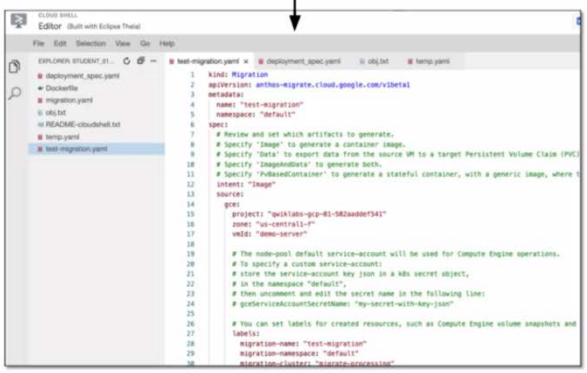


Adding a source enables migrations from a specific environment

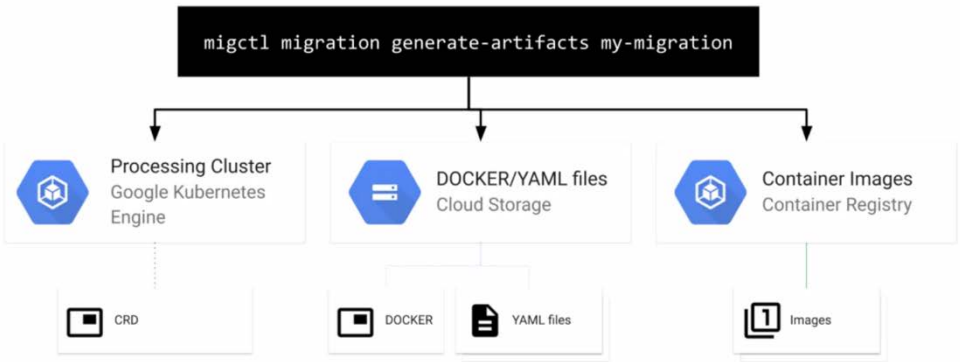
```
migctl source create ce my-ce-src --project my-project --zone zone
```

Creating a migration generates a migration plan

```
migctl migration create test-migration --source my-ce-src --vm-id my-id --intent
                                        Image
```

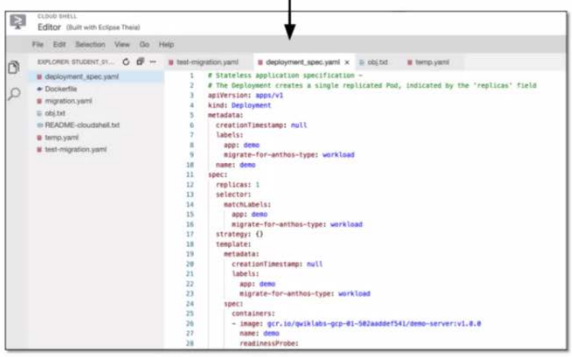


Executing a migration generates resources and artifacts



Deployment files typically need modification

```
migctl migration get-artifacts test-migration
```



Apply the configuration to deploy the workload

```
kubectl apply -f deployment_spec.yaml
```

GLOSSARY

TERM	DESCRIPTION
Tomcat server	Apache Tomcat is a long-lived, open source Java servlet container that implements several core Java enterprise specs, namely the Java Servlet, JavaServer Pages (JSP), and WebSockets APIs
Binary Authorization	• Binary Authorization is a deploy-time security control that ensures only trusted container images are deployed on Google Kubernetes Engine (GKE). • With Binary Authorization, you can require images to be signed by trusted authorities during the development process and then enforce signature validation when deploying. • By enforcing validation, you can gain tighter control over your container environment by ensuring only verified images are integrated into the build-and-release process.
Container Analysis API	An implementation of the Grafeas API, which stores, and enables querying and retrieval of critical metadata about all of your software artifacts.
gVisor	• gVisor is more lightweight than a VM while maintaining a similar level of isolation. • gVisor is an application kernel for containers that provides efficient defense-in-depth anywhere. • By providing each container with its own application kernel, gVisor limits the attack surface of the host.
Containerd	• industry-standard container runtime that's supported by Kubernetes, and used by many other projects. • Containerd provides the layering abstraction that allows for the implementation of a rich set of features like gVisor to extend Kubernetes functionality. • Containerd is considered more resource efficient and secure when compared to the Docker runtime.
Squid Proxy Server	• Squid is a full-featured web proxy cache server application which provides proxy and cache services for HTTP, File Transfer Protocol (FTP), and other popular network protocols. • Squid can implement caching and proxying of SSL requests and caching of DNS lookups, and perform transparent caching.
LDAP	• LDAP (Lightweight Directory Access Protocol) is an open and cross platform protocol used for directory services authentication. • Directory services store the users, passwords, and computer accounts, and share that information with other entities on the network.
Passing a startup script through command line	• You can only pass a local startup script file by using the gcloud command-line tool. Include the --metadata-from-file flag, followed by a metadata key pair, startup-script=PATH_TO_FILE, replacing PATH_TO_FILE with a relative path to the startup script: • gcloud compute instances create example-instance \ --metadata-from-file startup-script=examples/scripts/install.sh

gcloud topic filters	• gcloud topic filters - resource filters supplementary help • The --format=NAME[ATTRIBUTES](PROJECTION) and --filter=EXPRESSION flags along with projections can be used to format and change the default output to a more meaningful result.
SDK Properties	• Properties are settings that govern the behavior of the gcloud CLI and other SDK tools. • Properties can be used to define a per-product or per-service setting such as the account used by the gcloud CLI and other Cloud SDK tools for authorization, the default region to be used when working with Google Compute Engine resources, or even the option to turn off automatic Cloud SDK component updates. • Properties can also be used to define gcloud command-line tool preferences like verbosity level and prompt configuration for gcloud CLI commands.
RFC1918	• An RFC1918 address is an IP address that is assigned by an enterprise organization to an internal host. These IP addresses are used in private networks, which are not available, or reachable, from the Internet. • 10.0.0.0 – 10.255.255.255 (10/8 prefix) • 172.16.0.0 – 172.31.255.255 (172.16/12 prefix) • 192.168.0.0 – 192.168.255.255 (192.168/16 prefix)
gcloud services enable	• gcloud services enable - enables a service for consumption for a project • gcloud services list --available • gcloud services enable my-consumed-service • gcloud services enable my-consumed-service --async • gcloud services enable service1 service2 service3
SPF - Sender Policy Framework	An SPF record is a TXT record that is part of a domain's DNS (Domain Name Service). An SPF record lists all authorized hostnames / IP addresses that are permitted to send email on behalf of your domain.
DKIM - DomainKeys Identified Mail	DKIM is a process to validate sending domain names associated to email messages through cryptographic authentication. It achieves this by inserting a digital signature into the message header which is later verified by the receiving host to validate the authenticity of the sending domain.
DNSSEC	The Domain Name System Security Extensions (DNSSEC) is a suite of Internet Engineering Task Force (IETF) specifications for securing certain kinds of information provided by the Domain Name System (DNS) as used on Internet Protocol (IP) networks.
gRPC RPC - remote procedure call	• connect services in and across data centers with pluggable support for load balancing, tracing, health checking and authentication. • It is also applicable in last mile of distributed computing to connect devices, mobile applications and browsers to backend services.
SDK Configurations	A configuration is a named set of SDK properties. These properties are key-value pairs, organized in sections, that govern the behavior of the gcloud command-line tool and other SDK tools.
Cloud Auto ML	Cloud AutoML enables developers with limited machine learning expertise to train high-quality models specific to their business needs.

Cloud Tasks	• Cloud Tasks is a fully managed service that allows you to manage the execution, dispatch, and delivery of a large number of distributed tasks. • Using Cloud Tasks, you can perform work asynchronously outside of a user or service-to-service request.
Shielded VMs	• Shielded VMs are virtual machines (VMs) on Google Cloud hardened by a set of security controls that help defend against rootkits and bootkits. • Using Shielded VMs helps protect enterprise workloads from threats like remote attacks, privilege escalation, and malicious insiders. • Shielded VMs leverage advanced platform security capabilities such as secure and measured boot, a virtual trusted platform module (vTPM), UEFI firmware, and integrity monitoring.
gcloud projects	• The gcloud projects group lets you create and manage IAM policies for projects
query dry run	When you run a query in the `bq` command-line tool, you can use the `--dry_run` flag to estimate the number of bytes read by the query. You can also use the `dryRun` parameter when submitting a query job using the API or client libraries. You can use the estimate returned by the dry run to calculate query costs in the pricing calculator. You are not charged for performing the dry run.
Signed URL	• A signed URL is a URL that provides limited permission and time to make a request. Signed URLs contain authentication information in their query string, allowing users without credentials to perform specific actions on a resource. • When you generate a signed URL, you specify a user or service account which must have sufficient permission to make the request that the signed URL will make. • After you generate a signed URL, anyone who possesses it can use the signed URL to perform specified actions, such as reading an object, within a specified period of time.
Sink-ing logs	• 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.
Live Migration	Live migration migrates your running instances to another host in the same zone so that Google can perform maintenance such as a software or hardware update. It can not be used for changing machine type.
MaxSurge	maxSurge specifies the maximum number of instances that can be created over the desired number of instances. If maxSurge is set to 0, the rolling update can not create additional instances and is forced to update existing instances.
Traffic Splitting	• You can use traffic splitting to specify a percentage distribution of traffic across two or more of the versions within a service. Splitting traffic allows you to conduct A/B testing between your versions and provides control over the pace when rolling out features. • Each application in the app engine is different and it is not possible to split traffic between applications in App Engine. You can use traffic splitting to specify a percentage distribution of traffic across two or more of the versions within a service but not across applications.
App Engine flexible	• Not serverless

Activity Logs	• Activity logs display a list of all actions and you can restrict this down to a user and further filter by specifying Data access as the Activity types and GCS Bucket as the Resource type. • But that is the extent of the filter functionality in Activity logs. It is not possible to restrict the activity logs to just some specific buckets that we are interested in. Secondly, it is not possible to restrict the activity logs to just the gets and updates.
Cloud Run	• Cloud Run implements the Knative serving API, an open-source project to run serverless workloads on top of Kubernetes. • That means you can deploy Cloud Run services anywhere Kubernetes runs. • And if you need more control over your services (like access to GPU or more memory), you can also deploy these serverless containers in your own GKE cluster instead of using the fully managed environment. When using the fully managed environment, Cloud Run on GKE is serverless.
cross region SSH IAM permissions.	• No such thing
RPS	Requests per sec
to modify the gcloud configuration such that you are prompted for a zone when you execute the create instance commands above.	gcloud config unset compute/zone
IPSec VPN	• IPsec VPN is one of two common VPN protocols, or set of standards used to establish a VPN connection. • IPsec is set at the IP layer, and it is often used to allow secure, remote access to an entire network (rather than just a single device). IPsec VPNs come in two types: tunnel mode and transport mode.
Ad hoc SQL query (done in bq)	In SQL, an ad hoc query is a loosely typed command/query whose value depends upon some variable. Each time the command is executed, the result is different, depending on the value of the variable. ... An ad hoc query is short lived and is created at runtime
Expand Subnets	gcloud compute networks subnets expand-ip-range –region= –prefix-length=27
Parallel uploading files to GCS	• It splits a large file into component pieces, uploads them in parallel and then recomposes them once they're in the cloud (and deletes the temporary components it created locally). • gsutil -o GSUtil:parallel_composite_upload_threshold=150M cp ./localbigfile gs://your-bucket Where "localbigfile" is a file larger than 150MB. This divides up your data into chunks ~150MB and uploads them in parallel, increasing upload performance. • Faster than multi threading
.boto file	The boto configuration file contains values that control how gsutil behaves. For example, the prefer_api variable determines which API gsutil preferentially uses. Boto configuration file variables can be changed by editing the configuration file directly. While most users won't need to edit these variables, those do typically do so for one of the following reasons: • Setting up gsutil to work through a proxy. • Using customer-managed or customer-supplied encryption keys.

	• Performing specialized customization of global gsutil behavior.
TCP port 22	For SSH (Ubuntu, Linux)
TCP port 3389	For RDP
Cloud Audit Logs	Cloud Audit Logs provides the following audit logs for each Cloud project, folder, and organization: • Admin Activity audit logs • Policy Denied audit logs • Data Access audit logs • System Event audit logs
BigQuery data	Can't be accessed by members outside the organization
VPC Flow Logs	VPC Flow Logs record a sample of network flows sent from and received by VM instances. These logs can be used for network monitoring, forensics, real-time security analysis, and expense optimization. Flow logs are aggregated by connection, at 5-second intervals, from Compute Engine VMs and exported in real time. By subscribing to Cloud Pub/Sub, you can analyze flow logs using real-time streaming APIs.
VPC Network Logs	No such thing
Deploying a Cloud Function	deploy your function from the directory containing your function code with thegcloud functions deploy command: gcloud functions deploy NAME –runtime RUNTIME TRIGGER [FLAGS…]
Bastion Hosts	A **bastion host** is a special-purpose computer on a network specifically designed and configured to withstand attacks. The computer generally **hosts** a single application, for example a proxy **server**, and all other services are removed or limited to reduce the threat to the computer.
NAT – Network Address Translation	**Network address translation** (NAT) is a method of remapping an IP address space into another by modifying network address information in the IP header of packets while they are in transit across a traffic routing device.

CNAME	• **.A CNAME record is a type of DNS record. It directs traffic that requests a URL from your domain to the resources you want to serve, eg, objects in your Cloud Storage buckets.** • **For www.example.com, the CNAME record might contain the following information: NAME TYPE DATA www.example.com CNAME c.storage.googleapis.com.**
Data Catalog	• fully managed and scalable metadata management service that empowers organizations to quickly discover, understand, and manage all their data. • It offers a simple and easy-to-use search interface for data discovery, a flexible and powerful cataloging system for capturing both technical and

	business metadata, and a strong security and compliance foundation with Cloud Data Loss Prevention (DLP) and Cloud Identity and Access Management (IAM) integrations.
Binary Logging	• The binary log is a set of log files that contain information about data modifications made to a MySQL server instance. The log is enabled by starting the server with the --log-bin option. The binary log was introduced in MySQL 3.23. ... It contains all statements that update data.
Cloud Composer	•
signurl - Create a signed url	• gsutil signurl [-c <content_type>] [-d <duration>] [-m <http_method>] \ • [-p <password>] [-r <region>] [-b <project>] (-u | <private-key-file>) \ • (gs://<bucket_name> | gs://<bucket_name>/<object_name>)...
Add memory	• Stop instance and change it's machine type
Cloud Composer	• A fully managed workflow orchestration service built on Apache Airflow. • Author, schedule, and monitor pipelines that span across hybrid and multi-cloud environments • Cloud Composer's managed nature and Apache Airflow compatibility allows you to focus on authoring, scheduling, and monitoring your workflows as opposed to provisioning resources.

BILLING ACCOUNT ROLES

Role	Purpose	Level	Use Case

Billing Account Creator (roles/billing.creator)	Create new self-serve (online) billing accounts.	Organization	Use this role for initial billing setup or to allow creation of additional billing accounts. Users must have this role to sign up for Google Cloud with a credit card using their corporate identity. Tip: Minimize the number of users who have this role to help prevent proliferation of untracked cloud spend in your organization.
Billing Account Administrator (roles/billing.admin)	Manage billing accounts (but not create them).	Organization or billing account.	This role is an owner role for a billing account. Use it to manage payment instruments, configure billing exports, view cost information, link and unlink projects and manage other user roles on the billing account.
Billing Account User (roles/billing.user)	Link projects to billing accounts.	Organization or billing account.	This role has very restricted permissions, so you can grant it broadly, typically in combination with Project Creator. These two roles allow a user to create new projects linked to the billing account on which the role is granted.
Billing Account Viewer (roles/billing.viewer)	View billing account cost information and transactions.	Organization or billing account.	Billing Account Viewer access would usually be granted to finance teams, it provides access to spend information, but does not confer the right to link or unlink projects or otherwise manage the properties of the billing account.

Project Billing Manager (roles/billing.projectManager)	Link/unlink the project to/from a billing account.	Organization, folder, or project.	This role allows a user to attach the project to the billing account, but does not grant any rights over resources. Project Owners can use this role to allow someone else to manage the billing for the project without granting them resource access.

GCS ROLES

Role	Description
Storage Object Creator (roles/storage.objectCreator)	Allows users to create objects. Does not give permission to view, delete, or replace objects.
Storage Object Viewer (roles/storage.objectViewer)	Grants access to view objects and their metadata, excluding ACLs. Can also list the objects in a bucket.
Storage Object Admin (roles/storage.objectAdmin)	Grants full control over objects, including listing, creating, viewing, and deleting objects.
Storage HMAC Key Admin (roles/storage.hmacKeyAdmin)	Full control over HMAC keys in a project. This role can only be applied to a project.
Storage Admin (roles/storage.admin)	Grants full control of buckets and objects. When applied to an individual bucket, control applies only to the specified bucket and objects within the bucket.

APP ENGINE ROLES

Role	Title	Description
roles/appengine.appAdmin	App Engine Admin	Read/Write/Modify access to all application configuration and settings. To deploy new versions, you must also grant the Service Account User (roles/iam.serviceAccountUser) role. To use the gcloud tool to deploy, you must add the Storage Admin (roles/compute.storageAdmin) and Cloud Build Editor (roles/cloudbuild.builds.editor) roles.
roles/appengine.appCreator	App Engine Creator	Ability to create the App Engine resource for the project.
roles/appengine.appViewer	App Engine Viewer	Read-only access to all application configuration and settings.
roles/appengine.codeViewer	App Engine Code Viewer	Read-only access to all application configuration, settings, and deployed source code.
roles/appengine.deployer	App Engine Deployer	Read-only access to all application configuration and settings. To deploy new versions, you must also grant the Service Account User (roles/iam.serviceAccountUser) role. To use the gcloud tool to deploy, you must add the Storage Admin (roles/compute.storageAdmin) and Cloud Build Editor (roles/cloudbuild.builds.editor) roles. Cannot modify existing versions other than deleting versions that are not receiving traffic.
roles/appengine.serviceAdmin	App Engine Service Admin	Read-only access to all application configuration and settings. Write access to module-level and version-level settings. Cannot deploy a new version.

GCE ROLES

Just read em from google

SUPPORT LEVEL FOR PERMISSIONS

Support level	Description
SUPPORTED	The permission is fully supported in custom roles.
TESTING	The permission is being tested to check its compatibility with custom roles. You can include the permission in custom roles, but you might see unexpected behavior. Not recommended for production use.
NOT_SUPPORTED	The permission is not supported in custom roles.

CLOUD STORAGE STATIC WEBSITE

To host a static site in Cloud Storage, you need to create a Cloud Storage bucket, upload the content, and test your new site. You can serve your data directly from storage.googleapis.com, or you can verify that you own your domain and use your domain name. Either way, you'll get consistent, fast delivery from global edge caches.
You can create your static web pages however you choose. For example, you could hand-author pages by using HTML and CSS. You can use a static-site generator, such as Jekyll, Ghost, or Hugo, to create the content. Static-site generators make it easier for you to create a static website by letting you author in markdown, and providing templates and tools. Site generators generally provide a local web server that you can use to preview your content.
After your static site is working, you can update the static pages by using any process you like. That process could be as straightforward as hand-copying an updated page to the bucket. You might choose to use a more automated approach, such as storing your content on GitHub and then using a webhook to run a script that updates the bucket. An even more advanced system might use a continuous-integration /continuous-delivery (CI/CD) tool, such as Jenkins, to update the content in the bucket. Jenkins has a Cloud Storage plugin that provides a Google Cloud Storage Uploader post-build step to publish build artifacts to Cloud Storage.
If you have a web application that needs to serve static content or user-uploaded static media, using Cloud Storage can be a cost-effective and efficient way to host and serve this content, while reducing the amount of dynamic requests to your web application.

APP ENGINE ENVIRONMENTS

The App Engine environments

App Engine is well suited to applications that are designed using a microservice architecture, especially if you decide to utilize both environments. Use the following sections to learn and understand which environment best meets your application's needs.

When to choose the standard environment

Application instances run in a sandbox, using the runtime environment of a supported language listed below.

Applications that need to deal with rapid scaling.

The standard environment is optimal for applications with the following characteristics:

- Source code is written in specific versions of the supported programming languages:
- Python 2.7, Python 3.7, Python 3.8
- Java 8, Java 11
- Node.js 8, Node.js 10, and Node.js 12
- PHP 5.5, PHP 7.2, PHP 7.3, and PHP 7.4
- Ruby 2.5, Ruby 2.6, and Ruby 2.7
- Go 1.11, Go 1.12, Go 1.13, and Go 1.14
- Intended to run for free or at very low cost, where you pay only for what you need and when you need it. For example, your application can scale to 0 instances when there is no traffic.
- Experiences sudden and extreme spikes of traffic which require immediate scaling.

When to choose the flexible environment

Application instances run within Docker containers on Compute Engine virtual machines (VM).

Applications that receive consistent traffic, experience regular traffic fluctuations, or meet the parameters for scaling up and down gradually.

The flexible environment is optimal for applications with the following characteristics:

- Source code that is written in a version of any of the supported programming languages: Python, Java, Node.js, Go, Ruby, PHP, or .NET
- Runs in a Docker container that includes a custom runtime or source code written in other programming languages.
- Uses or depends on frameworks that include native code.
- Accesses the resources or services of your Google Cloud project that reside in the Compute Engine network.

Comparing high-level features

The following table summarizes the differences between the two environments:

Feature	Standard environment	Flexible environment
Instance startup time	Seconds	Minutes
Maximum request timeout	Depends on the runtime and type of scaling.	60 minutes
Background threads	Yes, with restrictions	Yes

Feature	Standard environment	Flexible environment
Background processes	No	Yes
SSH debugging	No	Yes
Scaling	Manual, Basic, Automatic	Manual, Automatic
Scale to zero	Yes	No, minimum 1 instance
Writing to local disk	Java 8, Java 11, Node.js, Python 3, PHP 7, Ruby, Go 1.11, and Go 1.12+ have read and write access to the /tmp directory. Python 2.7 and PHP 5.5 don't have write access to the disk.	Yes, ephemeral (disk initialized on each VM startup)
Modifying the runtime	No	Yes (through Dockerfile)
Deployment time	Seconds	Minutes
Automatic in-place security patches	Yes	Yes (excludes container image runtime)
Access to Google Cloud APIs & Services such as Cloud Storage, Cloud SQL, Memorystore, Tasks and others.	Yes	Yes
WebSockets	No Java 8, Python 2, and PHP 5 provide a proprietary Sockets API (beta), but the API is not available in newer standard runtimes.	Yes
Supports installing third-party binaries	Yes for Java 8, Java 11, Node.js, Python 3, PHP 7, Ruby, Go 1.11, and Go 1.12+.	Yes

Feature	Standard environment	Flexible environment
	No for Python 2.7 and PHP 5.5.	
Location	North America, Asia Pacific, or Europe	North America, Asia Pacific, or Europe
Pricing	Based on instance hours	Based on usage of vCPU, memory, and persistent disks

For an in-depth comparison of the environments, see the guide for your language: Python, Java, Go, or PHP.

Comparing the flexible environment to Compute Engine

The App Engine flexible environment has the following differences to Compute Engine:

- Flexible environment VM instances are restarted on a weekly basis. During restarts, Google's management services apply any necessary operating system and security updates.
- You always have root access to Compute Engine VM instances. By default, SSH access to the VM instances in the flexible environment is disabled. If you choose, you can enable root access to your app's VM instances.
- Code deployments can take longer as container images are built by using the Cloud Build service.
- The geographical region of a flexible environment VM instance is determined by the location that you specify for the App Engine application of your Cloud project. Google's management services ensures that the VM instances are co-located for optimal performance.

Testing on App Engine

Before configuring a new version to receive traffic, you can test it on App Engine. For example, to test a new version of your `default` service:

1. Deploy your new version and include the `--no-promote` flag:

   ```
   gcloud app deploy --no-promote
   ```
2. Access your new version by navigating to the following URL:

   `https://VERSION_ID-dot-default-dot-PROJECT_ID.REGION_ID.r.appspot.com`

   Note: You can find the version ID in the Cloud Console, or specify one when you deploy with the `--version` flag. The `gcloud` tool also outputs the version ID when you deploy.

Now you can test your new version in the App Engine runtime environment. You can debug your application by viewing its logs in the Google Cloud Console Logs Viewer. For more information, see Writing Application Logs.

Requests sent to https://PROJECT_ID.REGION_ID.r.appspot.com will still be routed to the version previously configured to receive traffic.

3. When you want to send traffic to the new version, use the Cloud Console to migrate traffic: Select the version you just deployed and click Migrate traffic.

TEST MISTAKES

QUESTION	ANSWER/MISTAKE
You have a project for your App Engine application that serves a development environment. The required testing has succeeded and you want to create a new project to serve as your production environment.	• Use gcloud to create the new project, and then deploy your application to the new project • The deployment manager configuration file contains configuration about the resources that need to be created in Google cloud, however, it does not offer the feature to copy app engine deployment into a new project
You have a website hosted on App Engine standard environment. You want 1% of your users to see a new test version of the website. You want to minimize complexity.	Deploy the new version in the same application and use the --splits option to give a weight of 99 to the current version and a weight of 1 to the new version.
You have an application deployed in a GKE Cluster as a Kubernetes workload with Daemon Sets. Your application has become very popular and is now struggling to cope up with increased traffic. You want to add more pods to your workload and want to ensure your cluster scales up and scales down automatically based on volume.	Enable autoscaling on Kubernetes Engine Enable Horizontal Pod Autoscaling for the Kubernetes deployment. is not right. Horizontal Pod Autoscaling can not be enabled for Daemon Sets, this is because there is only one instance of a pod per node in the cluster.
You have an application running in App Engine standard environment. You want to add a custom C# library to enhance the functionality of this application. However, C# isn't supported by App Engine standard. You want to maintain the serverless aspect of your application.	• Containerize your new application and deploy it to a Cloud Run environment. • Containerize your new application and deploy it to a Cloud Run on GKE environment.
You have an application running in Google Kubernetes Engine (GKE) with cluster autoscaling enabled. This application exposes a TCP endpoint. There are several replicas of the application. You have a Compute Engine instance in the same region but in another Virtual Private Cloud (VPC) called pt-network that has no overlapping CIDR range with the other VPC. The instance needs to connect to the application on GKE. You want to minimize effort.	In GKE, create a Service of type LoadBalancer that uses the application's Pods as backend. 2. Add an annotation to this service cloud.google.com/load-balancer-type: Internal 3. Peer the two VPCs together 4. Configure the Compute Engine instance to use the address of the load balancer that has been created.
You have an application that receives SSL-encrypted TCP traffic on port 443. Clients for this application are located all over the world. You want to minimize latency for the clients.	Network load balancer is used
You have annual audits every year and you need to provide external auditors access to the last 10 years of audit logs. You want to minimize the cost and operational overhead while following Google recommended practices.	Grant external auditors Storage Object Viewer role on the logs storage bucket Configure a lifecycle management policy on the logs bucket to delete objects older than 10 years Export audit logs to Cloud Storage via an audit log export sink.
You have asked your supplier to send you a purchase order and you want to enable them to upload the file to a cloud	Create a service account with just the permissions to upload files to the bucket. Create a JSON key for the

TEST MISTAKES

storage bucket within the next 4 hours. Your supplier does not have a Google account. You want to follow Google recommended practices.	service account. Execute the command gsutil signurl -m PUT -d 4h gs:///po.pdf.
You have one project called ptech-sa where you manage all your service accounts. You want to be able to use a service account from this project to take snapshots of VMs running in another project called ptech-vm.	Grant the service account the IAM Role of Compute Storage Admin in the project called ptech-vm.
You have sensitive data stored in three Cloud Storage buckets and have enabled data access logging. You want to verify activities for a particular user for these buckets, using the fewest possible steps. You need to verify the addition of metadata labels and which files have been viewed from those buckets	Using the GCP Console, filter the Stackdriver log to view the information. is the right answer. Data access logs is already enabled, so we already record all API calls that read the configuration or metadata of resources, as well as user-driven API calls that create, modify, or read user-provided resource data.
You have two compute instances in the same VPC but in different regions. You can SSH from one instance to another instance using their external IP address but not their internal IP address. What could be the reason for SSH failing on the internal IP address?	The firewall rule can be configured to allow SSH traffic from 0.0.0.0/0 but deny traffic from the VPC range e.g. 10.0.0.0/8. In this case, all SSH traffic from within the VPC is denied but external SSH traffic (i.e. on external IP) is allowed.
You have two compute instances in the same VPC but in different regions. You can SSH from one instance to another instance using their internal IP address but not their external IP address. What could be the reason for SSH failing on external IP address?	• The combination of compute instance network tags and VPC firewall rules only allow SSH from the subnets IP range. is the right answer. • The combination of compute instance network tags and VPC firewall rules can certainly result in SSH traffic being allowed from only subnets IP range. • The firewall rule can be configured to allow SSH traffic from just the VPC range e.g. 10.0.0.0/8.
You want to configure auto-healing for network load balancer for a group of Compute Engine instances that run in multiple zones using the fewest possible steps. You need to configure the recreation of VMs if they are unresponsive after 3 attempts of 10 seconds each	• Create a managed instance group. Set the Autohealing health check to healthy (HTTP) • Create a HTTP load balancer with a backend configuration that references an existing instance group. Define a balancing mode and set the maximum RPS to 10 is not right. You set RPS (Requests per Second) on load balancer when using RATE balancing mode. This has no effect on auto-healing.
You need to set up permissions for a set of Compute Engine instances to enable them to write data into a particular Cloud Storage bucket. You want to follow Google-recommended practices.	Create a service account with an access scope. Use the access scope 'https://www.googleapis.com/auth/cloud-platform'.

TEST MISTAKES

You need to set up a policy so that videos stored in a specific Cloud Storage Regional bucket are moved to Coldline after 90 days and then deleted after one year from their creation.	• Use Cloud Storage Object Lifecycle Management using Age conditions with SetStorageClass and Delete actions. Set the SetStorageClass action to 90 days and the Delete action to 365 days. is the right answer. • Object Lifecycle Management does not rewrite an object when changing its storage class. • This means that when an object is transitioned to Nearline Storage, Coldline Storage, or Archive Storage using the SetStorageClass feature, any subsequent early deletion and associated charges are based on the original creation time of the object, regardless of when the storage class changed.
You need to reduce GCP service costs for a division of your company using the fewest possible steps. You need to turn off all configured services in an existing GCP project	1. Verify that you are assigned the Project Owners IAM role for this project. 2. Locate the project in the GCP console, click Shut down and then enter the project ID.
You need to produce a list of the enabled Google Cloud Platform APIs for a GCP project using the gcloud command line in the Cloud Shell. The project name is my-project.	• Run gcloud projects list to get the project ID, and then run gcloud services list --project . is the right answer. For the gcloud services list command, --enabled is the default. So running gcloud services list --project is the same as running gcloud services list --project --enabled which would get all the enabled services for the project. • --available. is not right. --available return the services available to the project to enable and not the services that are enabled.
You have two workloads on GKE (Google Kubernetes Engine) – create-order and dispatch-order. create-order handles the creation of customer orders, and dispatch-order handles dispatching orders to your shipping partner. Both create-order and dispatch-order workloads have cluster autoscaling enabled. The create-order deployment needs to access (i.e. invoke web service of) dispatch-order deployment. dispatch-order deployment cannot be exposed publicly.	• Create a Service of type NodePort for dispatch-order and an Ingress Resource for that Service. Have create-order use the Ingress IP address. is not right. Exposes the Service on each Node's IP at a static port (the NodePort). If the compute instance has public connectivity, the dispatch-order can be accessed publicly which is undesirable. Secondly, dispatch-order has auto-scaling enabled so we shouldn't create a service of NodePort. If autoscaler spins up another pod on the node, it fails to initialize as the port on the node is already taken by an existing pod on the same node. • Create a Service of type ClusterIP for dispatch-order. Have create-order use the Service IP address. is the right answer. ClusterIP exposes the Service on a cluster-internal IP that is only reachable within the cluster. This satisfies our requirement that dispatch-order

TEST MISTAKES

	shouldn't be publicly accessible. create-order which is also located in the same GKE cluster can now access the ClusterIP of the service to reach dispatch-order
You host a static website in Cloud Storage. Recently you began to include links to PDF files on this site. Currently, when users click on links to these PDF files, their browser prompts them to save the file to their machine locally. However, you want the clicked PDF files to be displayed within the browser window directly without prompting the user to save the files locally.	• Set Content-Type metadata to application/pdf on the PDF file objects is the right answer. Content-Type allows browsers to render the object properly. • If the browser prompts users to save files to their machine, it is likely the browser does not see the Content-Type as application/pdf. Setting this would ensure the browser displays PDF files within the browser instead of popping up a download dialog.
You installed Stackdriver Logging agent on all compute instances. You now need to forward logs from all Compute Engine instances to a BigQuery dataset called pt-logs. You want to minimize cost. What should you do?	• Give the BigQuery Data Editor role on the pt-logs dataset to the service accounts used by your instances. Update your instances' metadata to add the following value: logs-destination: bq://pt-logs. is not right. • Among other things, roles/bigquery.dataEditor lets you Create, update, get, and delete the dataset's tables. However, setting a metadata tag logs-destination to bq://pt-logs has no effect on how the logs are generated or forwarded. • The stack driver agent is already installed so the logs are forwarded to stack driver logging and not to the BigQuery dataset • In Stack driver Logging, create a filter to view only Compute Engine logs. 2. Click Create Export. 3. Choose BigQuery as Sink Service, and the pt-logs dataset as Sink Destination. is the right answer. In stack driver logging, it is possible to create a filter to just query the compute engine logs which is what we are interested in.
You need to create a custom IAM role for use with a GCP service. All permissions in the role must be suitable for use. You also want to clearly share with your organization the status of the custom role. This will be the first version of the custom role.	• Use permissions in your role that use the SUPPORTED support level for role permissions. Set the role stage to ALPHA while testing the role permissions. • TESTING - The permission is being tested to check its compatibility with custom roles. You can include the permission in custom roles, but you might see unexpected behavior. Not recommended for production use. Since we want

TEST MISTAKES

	the role to be suitable for production use, we need "SUPPORTED" and not "TESTING".
You need to create a custom VPC with a single subnet. The subnet's range must be as large as possible.	Only one of three IPA CIDR blocks and out of them, whichev one has lowest no. of occupied bits 10.0.0.0/ 192.168.0.0/ 172.16.0.0/
You need to create a new billing account and then link it with an existing Google Cloud Platform project.	Project Billing Manager has power not Billing Admin
You need to host an application on a Compute Engine instance in a project shared with other teams. You want to prevent the other teams from accidentally causing downtime on that application.	• shielded VMs don't offer protection for accidental termination of the instance. • Enable deletion protection on the instance. is the right answer.
You need to monitor resources that are distributed over different projects in Google Cloud Platform. You want to consolidate reporting under the same Stackdriver Monitoring dashboard.	• Configure a single Stackdriver account, and link all projects to the same account. is the right answer. You can monitor resources of different projects in a single Stackdriver account by creating a Stackdriver workspace. • Use Shared VPC to connect all projects, and link Stackdriver to one of the projects. is not right. Linking Stackdriver to one project brings metrics from that project alone. A Shared VPC allows an organization to connect resources from multiple projects to a common Virtual Private Cloud (VPC) network so that they can communicate with each other securely and efficiently using internal IPs from that network. But it does not help in linking all projects to a single Stackdriver workspace/account.
You are designing a large distributed application with 30 microservices. Each of your distributed microservices needs to connect to a database back-end. You want to store the credentials securely.	recommended practice to store the credentials in a secret management system such as KMS. Applications often require access to small pieces of sensitive data at build or run time. These pieces of data are often referred to as secrets.
Your company's test suite is a custom C++ application that runs tests throughout each day on Linux virtual machines. The full test suite takes several hours to complete, running on a limited number of on-premises servers reserved for testing. Your company wants to move the testing infrastructure to the cloud, to reduce the amount of time it takes to fully test a change to the system, while changing the tests as little as possible.	Google Compute Engine managed instance group can help the testing application to scale to reduce the amount of time to run tests.

TEST MISTAKES

A Company is planning the migration of their web application to Google App Engine. However, they would still continue to use their on-premises database.	Setup the application using App Engine Flexible environment with Cloud VPN to connect to database App Engine standard cannot use Cloud VPN
A lead software engineer tells you that his new application design uses websockets and HTTP sessions that are not distributed across the web servers. You want to help him ensure his application will run properly on Google Cloud Platform.	• Meet with the cloud operations team and the engineer to discuss load balancer options. • We ain't gonna redesign system without websockets just to migrate to GCP
You have an application server running on Compute Engine in the europe-west1-d zone. You need to ensure high availability and replicate the server to the europe-west2-c zone using the fewest steps possible.	• Create a snapshot from the disk. Create a disk from the snapshot in the europe-west2-c zone. Create a new VM with that disk. • Disks are zonal resources, so they reside in a particular zone for their entire lifetime. The contents of a disk can be moved to a different zone by snapshotting the disk (using gcloud compute disks snapshot) and creating a new disk using –source-snapshot in the desired zone. The contents of a disk can also be moved across project or zone by creating an image (using gcloud compute images create) and creating a new disk using –image in the desired project and/or zone.
Your developers are trying to select the best compute service to run a static website. They have a dozen HTML pages, a few JavaScript files, and some CSS. They need the site to be highly available for the few weeks it is running. They also have a limited budget. What is the best service to use to run the site?	the website is static and needs to be hosted with high availability and limited budget, Cloud Storage would be an ideal choice.
Your developers have been thoroughly logging everything that happens in the API. The API allows end users to request the data as JSON, XML, CSV, and XLS. Supporting all of these formats is taking a lot of developer effort. Management would like to start tracking which options are used over the next month. Without modifying the code, what's the fastest way to be able to report on this data at the end of the month?	User-defined (logs-based) metrics are created by a user on a project. They count the number of log entries that match a given filter, or keep track of particular values within the matching log entries. Faster than sinking to bigQuery
You are a project owner and need your co-worker to deploy a new version of your application to App Engine. You want to follow Google's recommended practices. Which IAM roles should you grant your co-worker?	App Engine Deployer
You're writing a Python application and want your application to run in a sandboxed managed environment with the ability to scale up in seconds to account for huge spikes in demand. Which service should you host your application on?	App Engine Standard Environment provides rapid scaling as compared to App Engine Flexible Environment and is ideal for applications requiring quick start times and handle sudden and extreme spikes.

TEST MISTAKES

You are creating a solution to remove backup files older than 90 days from your backup Cloud Storage bucket. You want to optimize ongoing Cloud Storage spend. What should you do?	Write a lifecycle management rule in JSON and push it to the bucket with gsutil XML ain't supported by gsutil
You need to create a new Kubernetes Cluster on Google Cloud Platform that can autoscale the number of worker nodes. What should you do?	Create a cluster on Kubernetes Engine and enable autoscaling on Kubernetes Engine.
Your company has appointed external auditors for auditing the security of your setup. They want to check all the users and roles configured.	Ask Auditors to navigate to the IAM page and check member and roles section
Your company has created a new billing account and needs to move the projects to the billing account. What roles are needed to change the billing account?	To change the billing account for an existing project, you must be an owner on the project and a billing administrator on the destination billing account. (Project Billing Manager not required)
You're migrating an on-premises application to Google Cloud. The application uses a component that requires a licensing server. The license server has the IP address 10.28.0.10. You want to deploy the application without making any changes to the code or configuration. How should you go about deploying the application?	• Create a subnet with a CIDR range of 10.28.0.0/28. Reserve a static internal IP address of 10.28.0.10. Assign the static address to the license server instance • the IP is internal it can be reserved using the static internal IP address, which blocks it and prevents it from getting allocated to other resource.
You have a project using BigQuery. You want to list all BigQuery jobs for that project. You want to set this project as the default for the bq command-line tool. What should you do?	Use "gcloud config set project" to set the default project
You are creating a Kubernetes Engine cluster to deploy multiple pods inside the cluster. All container logs must be stored in BigQuery for later analysis. You want to follow Google-recommended practices. Which two approaches can you take?	Turn on Stackdriver Logging during the Kubernetes Engine cluster creation. Use the Stackdriver Logging export feature to create a sink to BigQuery. Specify a filter expression to export log records related to your Kubernetes Engine cluster only.
You have created a Kubernetes deployment, called Deployment-A, with 3 replicas on your cluster. Another deployment, called Deployment-B, needs access to Deployment-A. You cannot expose Deployment-A outside of the cluster. What should you do?	Create a Service of type ClusterIP for Deployment A. Have Deployment B use the Service IP address.
You have an App Engine application serving as your front-end. It's going to publish messages to Pub/Sub. The Pub/Sub API hasn't been enabled yet. What is the fastest way to enable the API?	• simplest way to enable an API for the project is using the GCP console. • providing the Pub/Sub Admin role does not provide the access to enable API. Enabling an API requires the following two Cloud Identity and Access Management permissions: 1.

TEST MISTAKES

	The servicemanagement.services.bind permission on the service to enable. This permission is present for all users for public services. For private services, you must share the service with the user who needs to enable it. 2. The serviceusage.services.enable permission on the project to enable the service on. This permission is present in the Editor role as well as in the Service Usage Admin role.
Your manager needs you to test out the latest version of MS-SQL on a Windows instance. You've created the VM and need to connect into the instance. What steps should you follow to connect to the instance?	• connecting to Windows instance involves installation of the RDP client. GCP does not provide RDP client and it needs to be installed. Generate Windows instance password to connect to the instance. • GCP Console does not have a direct RDP connectivity.
Your security team wants to be able to audit network traffic inside of your network. What's the best way to ensure they have access to the data they need?	VPC Flow Logs
You've been tasked with getting all of your team's public SSH keys onto to a specific Bastion host instance of a particular project. You've collected them all. With the fewest steps possible, what is the simplest way to get the keys deployed?	Add all of the keys into a file that's formatted according to the requirements. Use the gcloud compute instances add-metadatacommand to upload the keys to each instance Not added over project as only specific instance is granted the privilege
Your billing department has asked you to help them track spending against a specific billing account. They've indicated that they prefer to use Excel to create their reports so that they don't need to learn new tools. Which export option would work best for them?	File Export with CSV
Your project manager wants to delegate the responsibility to upload objects to Cloud Storage buckets to his team members. Considering the principle of least privilege, which role should you assign to the team members?	roles/storage.objectCreator Not the "object admin" role as only uploading of objects is required
Your application has a large international audience and runs stateless virtual machines within a managed instance group across multiple locations. One feature of the application lets users upload files and share them with other users. Files must be available for 30 days; after that, they are removed from the system entirely. Which storage solution should you choose?	Multi region GCS bucket Not Filestore as disks are regional and not ideal for sharable content
Your company wants to track whether someone is present in a meeting room reserved for a scheduled meeting. There are 1000 meeting rooms across 5 offices on 3 continents. Each room is equipped with a motion sensor that reports its status	NoSQL like Bigtable and Datastore solution is an ideal solution to store sensor ID and several different discrete items of information. It also provides an ability to join with

TEST MISTAKES

every second. The data from the motion detector includes only a sensor ID and several different discrete items of information. Analysts will use this data, together with information about account owners and office locations. Which database type should you use?	other data. Datastore can also be configured to store data in multi-region locations.
You're attempting to deploy a new instance that uses the centos 7 family. You can't recall the exact name of the family. Which command could you use to determine the family names?	gcloud compute images list
You've been tasked with getting all of only the operations team's public SSH keys onto to a specific Bastion host instance of a particular project. Currently Project wide access has already been granted to all the instances within the projects. With the fewest steps possible, how do you block or override the project level access on the Bastion host?	Use the gcloud compute instances add-metadata [INSTANCE_NAME] --metadata block-project-ssh-keys=TRUEcommand to block the access This will only allow users whose public SSH key is stored in instance-level metadata to access the instance. If you want your instance to use both project-wide and instance-level public SSH keys, set the instance metadata to allow project-wide SSH keys.
You have created an App engine application in the development environment. The testing for the application has been successful. You want to move the application to production environment. How can you deploy the application with minimal steps?	Perform app engine deploy using the --project parameter gcloud app deploy allows the –project parameter to be passed to override the project that the app engine application needs to be deployed to. Refer GCP documentation – Cloud SDK
Your company has deployed their application on managed instance groups, which is served through a network load balancer. They want to enable health checks for the instances. How do you configure the health checks?	Perform the health check using HTTP by hosting a basic web server Network Load Balancer does not support TCP health checks and hence HTTP health checks need to be performed. You can run a basic web server on each instance for health checks.
An engineer from your team accidentally deployed several new versions of NodeJS application on Google App Engine Standard. You are concerned the new versions are serving traffic. You have been asked to produce a list of all the versions of the application that are receiving traffic as well the percent traffic split between them. What should you do?	• gcloud app versions list --hide-no-traffic
Every employee of your company has a Google account. Your operational team needs to manage a large number of instances on the Compute Engine. Each member of this team needs only administrative access to the servers. Your security team wants to ensure that the deployment of credentials is operationally efficient and must be able to determine who accessed a given instance. What should you do?	• Ask each member of the team to generate a new SSH key pair and to add the public key to their Google account. • Grant the "compute.osAdminLogin" role to the Google group corresponding to this team.
For service discovery, you need to associate each of the Compute Engine instances of your VPC with an internal (DNS)	• Create a Cloud DNS zone, set its visibility to private and associate it with your VPC. Create

TEST MISTAKES

record in a custom zone. You want to follow Google recommended practices. What should you do?	records for each instance in that zone. is the right answer. • You should absolutely do this when you want internal DNS records in a custom zone. Cloud DNS gives you the option of private zones and internal DNS names.
You are building an application that will run in your data center. The application will use Google Cloud Platform (GCP) services like AutoML. You created a service account that has appropriate access to AutoML. You need to enable authentication to the APIs from your on-premises environment. What should you do?	• Use gcloud to create a key file for the service account that has appropriate permissions. is the right answer. • To use a service account outside of Google Cloud, such as on other platforms or on-premises, you must first establish the identity of the service account. Public/private key pairs provide a secure way of accomplishing this goal. You can create a service account key using the Cloud Console, the gcloud tool, the serviceAccounts.keys.create() method, or one of the client libraries.
You are hosting an application on bare metal servers in your data center. The application needs access to Cloud Storage. However, security policies prevent the servers hosting the application from having public IP addresses or access to the internet. You want to follow Google recommended practices to provide the application with access to Cloud Storage. What should you do?	• Using Cloud VPN or Interconnect, create a tunnel to a VPC in GCP Using Cloud Router to create a custom route advertisement for 199.36.153.4/30. Announce that network to your on-premises network through the VPN tunnel. In your on-premises network, configure your DNS server to resolve *.googleapis.com as a CNAME to restricted.googleapis.com is the right answer right, and it is what Google recommends. • You must configure routes so that Google API traffic is forwarded through your Cloud VPN or Cloud Interconnect connection, firewall rules on your on-premises firewall to allow the outgoing traffic, and DNS so that traffic to Google APIs resolves to the IP range you've added to your routes." "You can use Cloud Router Custom Route Advertisement to announce the Restricted Google APIs IP addresses through Cloud Router to your on-premises network. The Restricted Google APIs IP range is 199.36.153.4/30. While this is technically a public IP range, Google does not announce it publicly. This IP range is only accessible to hosts that can reach your Google Cloud projects through internal IP ranges, such as through a Cloud VPN or Cloud Interconnect connection." Without having a public IP address or access to the internet, the only way you could connect to cloud storage is if you have an internal route to it. So Negotiate with the

TEST MISTAKES

	security team to be able to give public IP addresses to the servers is not right. Following “Google recommended practices” is synonymous with “using Google’s services”
You have a number of compute instances belonging to an unmanaged instances group. You need to SSH to one of the Compute Engine instances to run an ad hoc script. You’ve already authenticated gcloud, however, you don’t have an SSH key deployed yet. In the fewest steps possible, what’s the easiest way to SSH to the instance?	Use the gcloud compute ssh command.
You have a Linux VM that must connect to Cloud SQL. You created a service account with the appropriate access rights. You want to make sure that the VM uses this service account instead of the default Compute Engine service account. What should you do?	• When creating the VM via the web console, specify the service account under the ‘Identity and API Access’ section. is the right answer. You can set the service account at the time of creating the compute instance. • You can also update the service account used by the instance - this requires that you stop the instance first and then update the service account. Setting/Updating the service account can be done either via the web console or by executing gcloud command or by the REST API.
You have a development project with appropriate IAM roles defined. You are creating a production project and want to have the same IAM roles on the new project, using the fewest possible steps. What should you do?	• Use gcloud iam roles copy and specify the production project as the destination project. • In the Google Cloud Platform Console, use the 'create role from role' functionality. is not right. This creates a role in the same (development) project, not in the production project.
You have a developer laptop with Cloud SDK installed on Ubuntu. The cloud SDK was installed from Google Cloud Ubuntu package repository. You want to test your application locally on your laptop with Cloud Datastore. What should you do?	• Install the cloud-datastore-emulator component using the gcloud components install command. is the right answer. • The Datastore emulator provides local emulation of the production Datastore environment. You can use the emulator to develop and test your application locally
You deployed an App Engine application using gcloud app deploy, but it did not deploy to the intended project. You want to find out why this happened and where the application deployed. What should you do?	Go to Cloud Shell and run gcloud config list to review the Google Cloud configurations used for deployment.
You deployed a number of services to Google App Engine Standard. The services are designed as microservices with several interdependencies between them. Most services have few version upgrades but some key services have over 20 version upgrades. You identified an issue with the service pt-	Execute gcloud app versions migrate v3 --service="pt-createOrder"

TEST MISTAKES

createOrder and deployed a new version v3 for this service. You are confident this works and want this new version to receive all traffic for the service. You want to minimize effort and ensure the availability of service. What should you do?	
You built an application on Google Cloud Platform that uses Cloud Spanner. The support team needs to monitor the environment but should not have access to the data. You need a streamlined solution to grant the correct permissions to your support team, and you want to follow Google recommended practices. What should you do?	Add the support team group to the roles/monitoring.viewer role
You are using multiple configurations for gcloud. You want to review the configured Kubernetes Engine cluster of an inactive configuration using the fewest possible steps. What should you do?	Use kubectl config get-contexts to review the output.
You are using Google Kubernetes Engine with autoscaling enabled to host a new application. You want to expose this new application to the public, using HTTPS on a public IP address. What should you do?	• Create a Kubernetes Service of type NodePort for your application, and a Kubernetes Ingress to expose this Service via a Cloud Load Balancer. is the right answer. • This meets all our requirements. With (Global) Cloud Load Balancing, a single anycast IP front-ends all your backend instances in regions around the world. It provides cross-region load balancing, including automatic multi-region failover, which gently moves traffic in fractions if backends become unhealthy. • The ingress accepts traffic from the cloud load balancer and can distribute the traffic across the pods in the cluster.
You are using Deployment Manager to create a Google Kubernetes Engine cluster. Using the same Deployment Manager deployment, you also want to create a DaemonSet in the kube-system namespace of the cluster. You want a solution that uses the fewest possible services. What should you do?	• Add the cluster's API as a new Type Provider in Deployment Manager, and use the new type to create the DaemonSet. • A type provider exposes all resources of a third-party API to Deployment Manager as base types that you can use in your configurations. • If you have a cluster running on Google Kubernetes Engine, you could add the cluster as a type provider and access the Kubernetes API using Deployment Manager. Using these inherited API, you can create a DaemonSet.
You are setting up a Windows VM on Compute Engine and want to make sure you can log in to the VM via RDP. What should you do?	• After the VM has been created, use gcloud compute reset-windows-password to retrieve the login credentials for the VM. is the right answer. • You can generate Windows passwords using either the Google Cloud Console or the gcloud

TEST MISTAKES

	command-line tool. This option uses the right syntax to reset the windows password. gcloud compute reset-windows-password windows-instance
You are migrating a mission critical on-premises application to cloud. The application requires 96 vCPUs to perform its task. You want to make sure the application runs in a similar environment on GCP. What should you do?	When creating the VM, use machine type n1-standard-96. Create the VM using Compute Engine default settings. Use gcloud to modify the running instance to have 96 vCPUs. is not right. You can't increase the vCPUs to 96 without changing the machine type. While it is possible to set machine type using gcloud, this would mean downtime for the mission-critical application
You have a managed instance group comprised of preemptible VM's. All of the VM's keep deleting and recreating themselves every minute. What is a possible cause of this behavior?	• Your managed instance group's VM's are toggled to only last 1 minute in preemptible settings.
You write a Python script to connect to Google BigQuery from a Google Compute Engine virtual machine. The script is printing errors that it cannot connect to BigQuery. What should you do to fix the script?	• Run your script on a new virtual machine with the BigQuery access scope enabled • by default an instance is associated with default service account and default access scope, neither of which provides an access to BigQuery. While Service account is the recommended approach and Access scope are legacy, access scope still need to granted to the instance for applications to access the services. So enabling only the Service Account with role would not enable the script to access BigQuery.
You have created a Kubernetes engine cluster named 'project-1'. You've realized that you need to change the machine type for the cluster from n1-standard-1 to n1-standard-4. What is the command to make this change?	• When you need to change the machine profile of your Compute Engine cluster, you can create a new node pool and then migrate your workloads over to the new node pool. To migrate your workloads without incurring downtime, you need to: Mark the existing node pool as unschedulable. Drain the workloads running on the existing node pool. Delete the existing node pool.
You need to have a backup/rollback plan in place for your application that is distributed across a large managed instance group. What is the preferred method for doing so?	• Use the Rolling Update feature to deploy/roll back versions with different managed instance group templates.
You're deploying an application to a Compute Engine instance, and it's going to need to make calls to read from Cloud Storage and Bigtable. You want to make sure you're following the principle of least privilege. What's the easiest way to	• Create a new service account and key with the required limited permissions. Set the instance to use the new service account. Edit the code to use the service account key.

TEST MISTAKES

ensure the code can authenticate to the required Google Cloud APIs?	• Default GCE does not have permission only to APIs so no point in even thinking about access scopes for it
A SysOps admin has configured a lifecycle rule on an object versioning disabled multi-regional bucket. Which of the following statement effect reflects the following lifecycle config?	• Move objects to Coldline Storage after 365 days if the storage class in Multi-regional First rule has no effect on the bucket. • Since object versioning is disabled, object will never be live i.e. never have a latest version
The development team has provided you with a Kubernetes Deployment file. You have no infrastructure yet and need to deploy the application. What should you do?	• Use gcloud to create a Kubernetes cluster. Use kubectl to create the deployment.
One of the microservices in your application has an intermittent performance problem. You have not observed the problem when it occurs but when it does, it triggers a particular burst of log lines. You want to debug a machine while the problem is occurring. What should you do?	• Set up a log metric in Stackdriver Logging, and then set up an alert to notify you when the number of log lines increases past a threshold.
You're writing a Java application with lot a threading and concurrency. You want your application to run in a sandboxed managed environment with the ability to perform SSH debugging to check on any thread dump for troubleshooting. Which service should you host your application on?	• App Engine provides the managed service and Flexible environment supports the ability to perform SSH debugging.
A company is hosting their Echo application on Google Cloud using Google Kubernetes Engine. The application is deployed with deployment echo-deployment exposed with echo-service. They have a new image that needs to be deployed for the application. How can the change be deployed with minimal downtime?	• Update image using kubectl set image deployment • the image can be directly updated using the kubectl command and Kubernetes Engine performs a rolling update.
A company uses Cloud Storage for storing their critical data. As a part of compliance, the objects need to be encrypted using customer-supplied encryption keys. How should the object be handled to support customer-supplied encryption?	• Use gsutil with GSUtil:encryption_key=[YOUR_ENCRYPTION_KEY] to pass the encryption key
The development team needs a regional MySQL database with point-in-time recovery for a new proof-of-concept application. What's the most inexpensive way to enable point-in-time recovery?	• Binary Logging
Your application deployed on a Google Compute Engine virtual machine instance needs to connect to Google Cloud Pub/Sub. What is the best way to provision the access to the application?	• VM needs to be granted permissions using the service account to be able to communicate with Cloud Pub/Sub. • Without permission, there is no point of having an access scope

TEST MISTAKES

Your company pushes batches of sensitive transaction data from its application server VMs to Cloud Pub/Sub for processing and storage. What is the Google-recommended way for your application to authenticate to the required Google Cloud services?	• Ensure that VM service accounts are granted the appropriate Cloud Pub/Sub IAM roles.
You have an application deployed on Kubernetes Engine using a Deployment named echo-deployment. The deployment is exposed using a Service called echo-service. You need to perform an update to the application with minimal downtime to the application. What should you do?	• Use kubectl set image deployment/echo-deployment • image can be directly updated using the kubectl command and Kubernetes Engine performs a rolling update.
You have a Linux VM that must connect to Cloud SQL. You created a service account with the appropriate access rights. You want to make sure that the VM uses this service account instead of the default Compute Engine service account. What should you do?	• When creating the VM via the web console, specify the service account under the 'Identity and API Access' section.
You have a project for your App Engine application that serves a development environment. The required testing has succeeded and you want to create a new project to serve as your production environment. What should you do?	• Use gcloud to create the new project, and then deploy your application to the new project. • You can't copy an application to another project
You created an instance of SQL Server 2017 on Compute Engine to test features in the new version. You want to connect to this instance using the fewest number of steps. What should you do?	• Install a RDP client on your desktop. Verify that a firewall rule for port 3389 exists
You're working on setting up a cluster of virtual machines with GPUs to perform some 3D rendering for a customer. They're on a limited budget and are looking for ways to save money. What is the best solution for implementing this?	• Use an autoscaled managed instance group containing some preemptible instances. • No GPUs supported for app engine
You need to connect to one of your Compute Engine instances using SSH. You've already authenticated gcloud, however, you don't have an SSH key deployed yet. In the fewest steps possible, what's the easiest way to connect to the app?	• Use gcloud compute ssh • gcloud compute ssh is a thin wrapper around the ssh(1) command that takes care of authentication and the translation of the instance name into an IP address. gcloud compute ssh ensures that the user's public SSH key is present in the project's metadata. If the user does not have a public SSH key, one is generated using ssh-keygen(1) (if the –quiet flag is given, the generated key will have an empty passphrase).
You want to migrate an application from Google App Engine Standard to Google App Engine Flex. Your application is currently serving live traffic and you want to ensure everything is working in Google App Engine Flex before	• Set env: flex in app.yaml gcloud app deploy --no-promote --version=[NEW_VERSION] • Validate [NEW_VERSION] in App Engine Flex

TEST MISTAKES

migrating all traffic. You want to minimize effort and ensure the availability of service. What should you do?	• gcloud app versions migrate [NEW_VERSION]
You want to use Google Cloud Storage to host a static website on http://www.example.com for your staff. You created a bucket example-static-website and uploaded index.html and css files to it. You turned on static website hosting on the bucket and set up a CNAME record on http://www.example.com to point to c.storage.googleapis.com. You access the static website by navigating to http://www.example.com in the browser but your index page is not displayed. What should you do?	• Delete the existing bucket, create a new bucket with the name www.example.com and upload the html/css files. is the right answer. We need to create a bucket whose name matches the CNAME you created for your domain. For example, if you added a CNAME record pointing www.example.com to c.storage.googleapis.com., then create a bucket with the name "www.example.com".
Your company has an existing GCP organization with hundreds of projects and a billing account. Your company recently acquired another company that also has hundreds of projects and its own billing account. You would like to consolidate all GCP costs of both GCP organizations onto a single invoice and you would like to do this as soon as possible. What should you do?	• Link the acquired company's projects to your company's billing account. • Migrating em to existing org takes a lotta time
Your company has migrated most of the data center VMs to Google Compute Engine. The remaining VMs in the data center host legacy applications that are due to be decommissioned soon and your company has decided to retain them in the datacenter. Due to a change in the business operational model, you need to introduce changes to one of the legacy applications to read files from Google Cloud Storage. However, your data center does not have access to the internet and your company doesn't want to invest in setting up internet access as the data center is due to be turned off soon. Your data center has a partner interconnect to GCP. You wish to route traffic from your datacenter to Google Storage through partner interconnect. What should you do?	• In on-premises DNS configuration, map *.googleapis.com to restricted.googleapis.com, which resolves to the 199.36.153.4/30. 2. Configure Cloud Router to advertise the 199.36.153.4/30 IP address range through the Cloud VPN tunnel. 3. Add a custom static route to the VPC network to direct traffic with the destination 199.36.153.4/30 to the default internet gateway. 4. Created a Cloud DNS managed private zone for *.googleapis.com that maps to 199.36.153.4/30 and authorize the zone for use by VPC network
Your Company is planning to migrate all Java web applications to Google App Engine. However, you still want to continue using your on-premise database. How can you set up the app engine to communicate with your on-premise database while minimizing effort?	• Setup the application using App Engine Standard environment with Cloud VPN to connect to an on-premise database. • Converting to a container model involves effort and we want to minimize effort
Your company procured a license for a third-party cloud-based document signing system for the procurement team. All members of the procurement team need to sign in with the same service account. Your security team prohibits sharing service account passwords. You have been asked to recommend a solution that lets the procurement team login as the service account in the document signing system but	• Register the application as a password vaulted app and set the credentials to the service account credentials. • As a G Suite or Cloud Identity administrator, the password vaulted apps service enables you to manage access to some of the apps that don't

TEST MISTAKES

without the team knowing the service account password. What should you do?	support federation and that are available to users on the User Dashboard.
Your company stores sensitive PII data in a cloud storage bucket. The objects are currently encrypted by Google-managed keys. Your compliance department has asked you to ensure all current and future objects in this bucket are encrypted by customer-managed encryption keys. You want to minimize effort. What should you do?	• In the bucket advanced settings, select the Customer-managed key and then select a Cloud KMS encryption key. 2. Rewrite all existing objects using gsutil rewrite to encrypt them with the new Customer-managed key.
Your company uses a legacy application that still relies on the legacy LDAP protocol to authenticate. Your company plans to migrate this application to cloud and is looking for a cost effective solution while minimizing any developer effort. What should you do?	• Use secure LDAP to authenticate the legacy application and ask users to sign in through Gmail • Using cloud directory sync takes developer effort
Your company uses BigQuery for data warehousing. Over time, many different business units in your company have created 1000+ datasets across hundreds of projects. Your CIO wants you to examine all datasets to find tables that contain an employee_ssn column. You want to minimize effort in performing this task. What should you do?	• Go to Data Catalog and search for employee_ssn in the search box.
Your company wants to move all documents from a secure internal NAS drive to a Google Cloud Storage (GCS) bucket. The data contains personally identifiable information (PII) and sensitive customer information. Your company tax auditors need access to some of these documents. What security strategy would you recommend on GCS?	• Grant no Google Cloud Identity and Access Management (Cloud IAM) roles to users, and use granular ACLs on the bucket.
Your managed instance group raised an alert stating that new instance creation has failed to create new instances. You need to maintain the number of running instances specified by the template to be able to process expected application traffic. What should you do?	• Verify that the instance template being used by the instance group contains valid syntax. Delete any persistent disks with the same name as instance names. Set the disks.autoDelete property to true in the instance template.
Your customer has implemented a solution that uses Cloud Spanner and notices some read latency-related performance issues on one table. This table is accessed only by their users using a primary key. The table schema is shown below.	• Change the primary key to not have monotonically increasing values. • You should be careful when choosing a primary key to not accidentally create hotspots in your database. One cause of hotspots is having a column whose value monotonically increases as the first key part because this results in all inserts occurring at the end of your keyspace. This pattern is undesirable because Cloud Spanner divides data among servers by key ranges, which means all your inserts will be directed at a single server that will end up doing all the work.