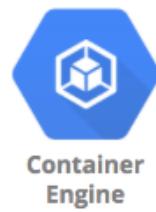
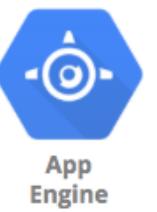
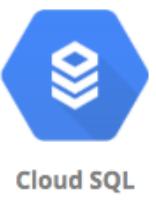
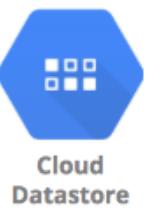
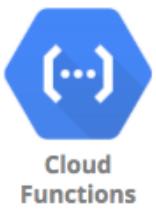


Google Certified Professional Cloud Architect

Getting Started

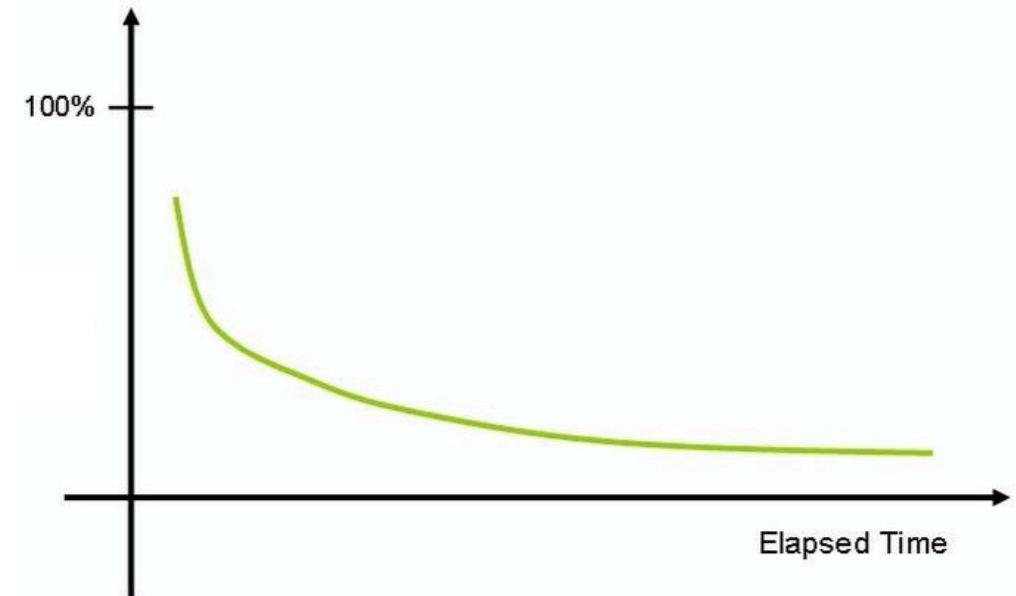
In 28
Minutes



- GCP has 200+ services. This exam expects knowledge of 40+ Services.
- Exam *tests your decision making abilities*:
 - Which service do you choose in which situation?
 - How do you **trade-off** between resilience, performance and cost while not compromising on security and operational excellence?
- This course is **designed** to help you *make these tough choices*
- **Our Goal** : Enable you to architect amazing solutions in GCP

How do you put your best foot forward?

- Challenging certification - Expects you to understand and **REMEMBER** a number of services
- As time passes, humans forget things.
- How do you improve your chances of remembering things?
 - Active learning - think and take notes
 - Review the presentation every once in a while



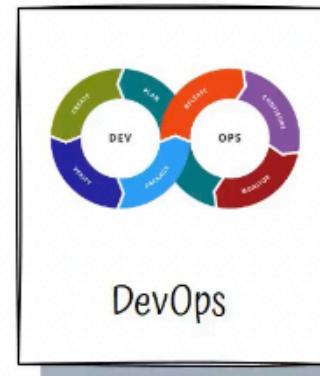
Our Approach

- Three-pronged approach to reinforce concepts:
 - Presentations (Video)
 - Demos (Video)
 - **Two kinds of quizzes:**
 - Text quizzes
 - Video quizzes
- (Recommended) Take your time. Do not hesitate to replay videos!
- (Recommended) Have Fun!



FASTEST ROADMAPS

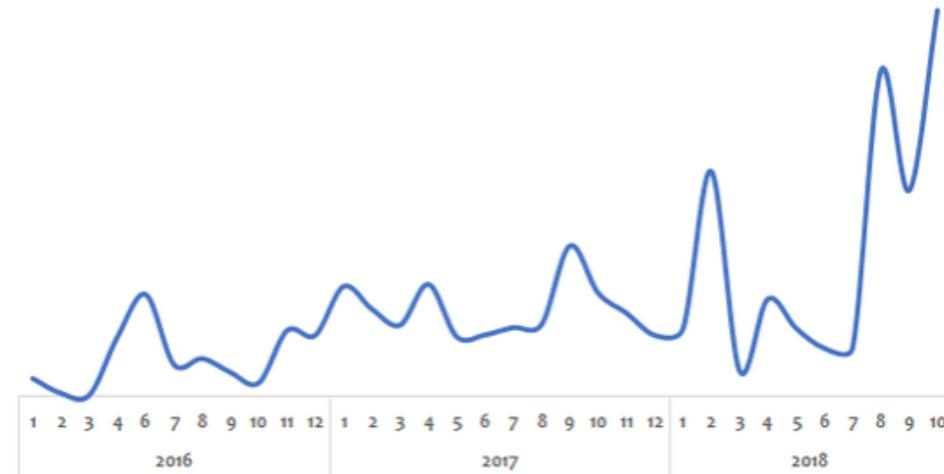
in28minutes.com



GCP - Getting started

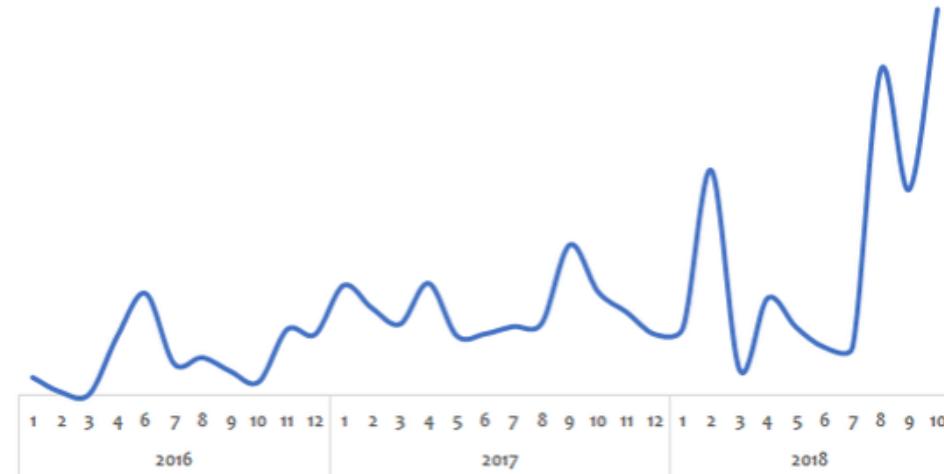
Before the Cloud - Example 1 - Online Shopping App

In 28
Minutes



- Challenge:
 - Peak usage during holidays and weekends
 - Less load during rest of the time
- Solution (before the Cloud):
 - **PEAK LOAD provisioning : Procure (Buy) infrastructure for peak load**
 - What would the infrastructure be doing during periods of low loads?

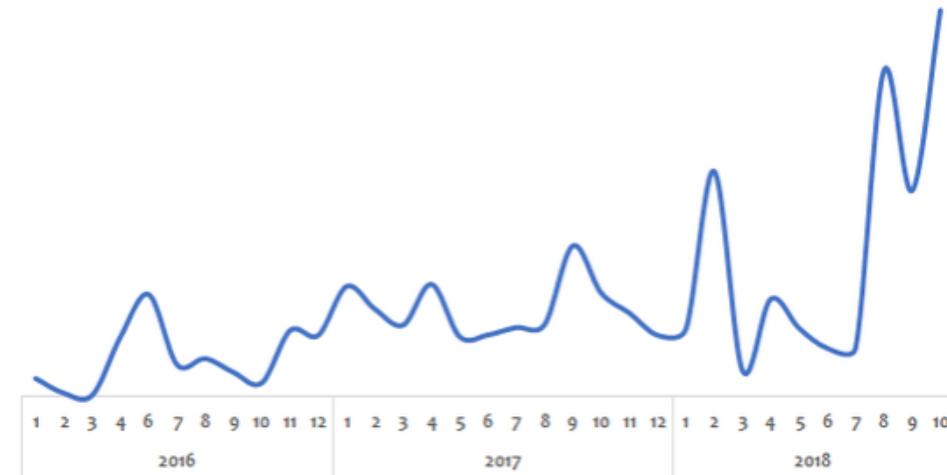
Before the Cloud - Example 2 - Startup



- Challenge:
 - Startup suddenly becomes popular
 - How to handle the **sudden increase** in load?
- Solution (before the Cloud):
 - **Procure** (Buy) infrastructure assuming they would be successful
 - What if they are not successful?

Before the Cloud - Challenges

In 28
Minutes



- High cost of procuring infrastructure
- Needs ahead of time planning (**Can you guess the future?**)
- Low infrastructure utilization (**PEAK LOAD** provisioning)
- Dedicated infrastructure maintenance team (**Can a startup afford it?**)

Silver Lining in the Cloud

- How about provisioning (renting) resources when you want them and releasing them back when you do not need them?
 - On-demand resource provisioning
 - Also called Elasticity



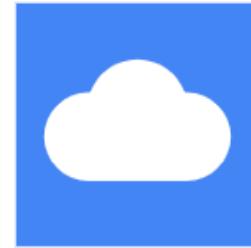
Cloud - Advantages

- Trade "capital expense" for "variable expense"
- Benefit from massive economies of scale
- Stop guessing capacity
- Stop spending money running and maintaining data centers
- "Go global" in minutes



Google Cloud Platform (GCP)

- One of the Top 3 cloud service providers
- Provides a number of services (200+)
- Reliable, secure and highly-performant:
 - Infrastructure that powers 8 services with over 1 Billion Users:
Gmail, Google Search, YouTube etc
- One thing I love : "**cleanest cloud**"
 - Net carbon-neutral cloud (electricity used matched 100% with renewable energy)
- The entire course is all about GCP. You will learn it as we go further.



Google Cloud

Best path to learn GCP!



Compute Engine



Cloud Functions



Cloud Datastore



Cloud SQL



App Engine



Container Engine

- Cloud applications make use of multiple GCP services
- There is no **single path** to learn these services independently
- **HOWEVER**, we've worked out a simple path!

Setting up GCP Account

- Create GCP Account

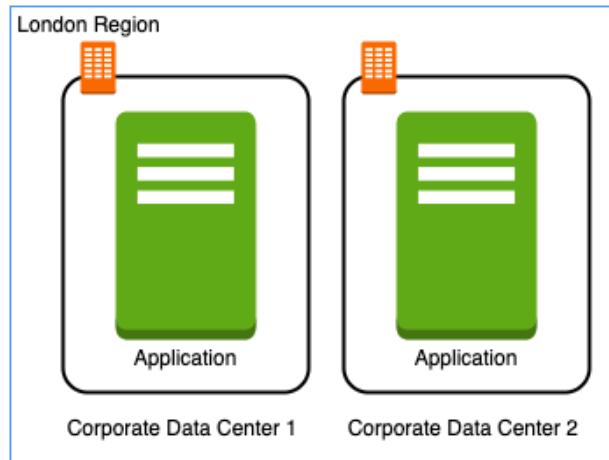
Regions and Zones

Regions and Zones



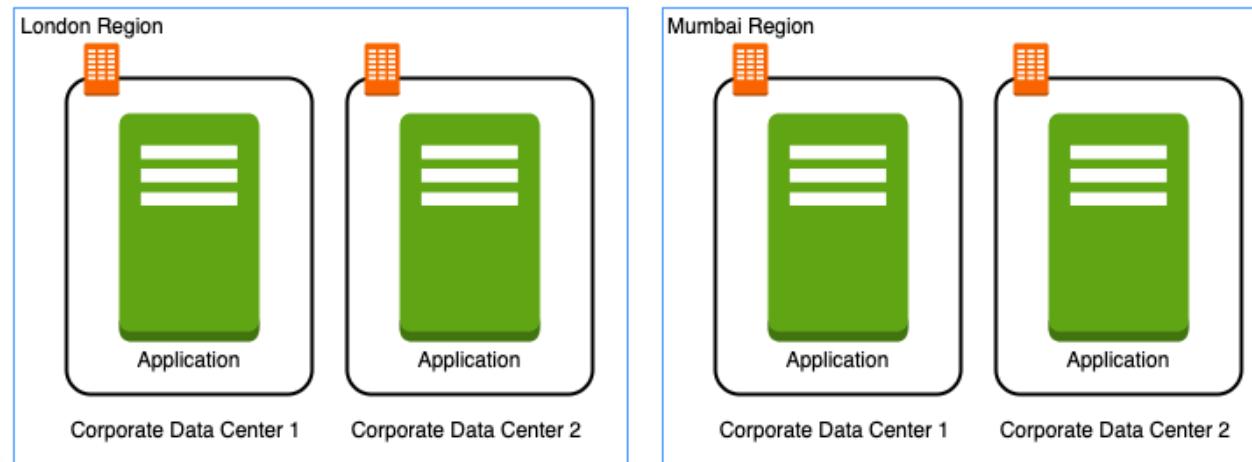
- Imagine that your application is deployed in a data center in London
- What would be the challenges?
 - Challenge 1 : Slow access for users from other parts of the world (**high latency**)
 - Challenge 2 : What if the data center crashes?
 - Your application goes down (**low availability**)

Multiple data centers



- Let's add in one more data center in London
- What would be the challenges?
 - Challenge 1 : Slow access for users from other parts of the world
 - Challenge 2 (**SOLVED**) : What if one data center crashes?
 - Your application is still available from the other data center
 - Challenge 3 : What if entire region of London is unavailable?
 - Your application goes down

Multiple regions



- Let's add a new region : Mumbai
- What would be the challenges?
 - Challenge 1 (**PARTLY SOLVED**) : Slow access for users from other parts of the world
 - You can solve this by adding deployments for your applications in other regions
 - Challenge 2 (**SOLVED**) : What if one data center crashes?
 - Your application is still live from the other data centers
 - Challenge 3 (**SOLVED**) : What if entire region of London is unavailable?
 - Your application is served from Mumbai

Regions

- Imagine setting up data centers in different regions around the world
 - Would that be easy?
- (Solution) Google provides **20+ regions** around the world
 - Expanding every year
- **Region** : Specific geographical location to host your resources
- **Advantages:**
 - High Availability
 - Low Latency
 - Global Footprint
 - Adhere to government regulations



Zones

- How to achieve high availability in the same region (or geographic location)?
 - Enter Zones
- Each Region has three or more **zones**
- (Advantage) **Increased availability and fault tolerance** within same region
- (Remember) Each Zone has **one or more discrete clusters**
 - **Cluster** : distinct physical infrastructure that is housed in a data center
- (Remember) Zones in a region are connected through **low-latency** links



Regions and Zones examples

New Regions and Zones are constantly added

Region Code	Region	Zones	Zones List
us-west1	The Dalles, Oregon, North America	3	us-west1-a us-west1-b us-west1-c
europe-north1	Hamina, Finland, Europe	3	europe-north1-a, europe-north1-b europe-north1-c
asia-south1	Mumbai, India APAC	3	asia-south1-a, asia-south1-b asia-south1-c

Compute

Compute Engine Fundamentals

Google Compute Engine (GCE)

- In corporate data centers, applications are deployed to physical servers
- Where do you deploy applications in the cloud?
 - Rent virtual servers
 - **Virtual Machines** - Virtual servers in GCP
 - **Google Compute Engine (GCE)** - Provision & Manage Virtual Machines



Compute Engine - Features

In 28
Minutes



- Create and manage lifecycle of Virtual Machine (VM) instances
- **Load balancing and auto scaling** for multiple VM instances
- **Attach storage** (& network storage) to your VM instances
- Manage **network connectivity and configuration** for your VM instances
- **Our Goal:**
 - Setup VM instances as HTTP (Web) Server
 - Distribute load with Load Balancers

Compute Engine Hands-on

- Let's create a few VM instances and play with them
- Let's check out the lifecycle of VM instances
- Let's use SSH to connect to VM instances



Compute Engine Machine Family

- What type of hardware do you want to run your workloads on?
- Different Machine Families for Different Workloads:
 - **General Purpose (E2, N2, N2D, N1)** : Best price-performance ratio
 - Web and application servers, Small-medium databases, Dev environments
 - **Memory Optimized (M2, M1)**: Ultra high memory workloads
 - Large in-memory databases and In-memory analytics
 - **Compute Optimized (C2)**: Compute intensive workloads
 - Gaming applications



Compute Engine Machine Types

Machine name	vCPUs ¹	Memory (GB)	Max number of persistent disks (PDs) ²	Max total PD size (TB)	Local SSD	Maximum egress bandwidth (Gbps) ³
e2-standard-2	2	8	128	257	No	4
e2-standard-4	4	16	128	257	No	8
e2-standard-8	8	32	128	257	No	16
e2-standard-16	16	64	128	257	No	16
e2-standard-32	32	128	128	257	No	16

- How much CPU, memory or disk do you want?
 - Variety of machine types are available for each machine family
 - Let's take an example : **e2-standard-2**:
 - **e2** - Machine Type Family
 - **standard** - Type of workload
 - **2** - Number of CPUs
- Memory, disk and networking capabilities increase along with vCPUs

Image



- What operating system and what software do you want on the instance?
- Type of Images:
 - **Public Images:** Provided & maintained by Google or Open source communities or third party vendors
 - **Custom Images:** Created by you for your projects

Compute Engine Hands-on : Setting up a HTTP server

```
#!/bin/bash
sudo su
apt update
apt -y install apache2
sudo service apache2 start
sudo update-rc.d apache2 enable
echo "Hello World" > /var/www/html/index.html
echo "Hello world from $(hostname) $(hostname -I)" > /var/www/html/index.html
```

- Commands:

- sudo su - execute commands as a root user
- apt update - Update package index - pull the latest changes from the APT repositories
- apt -y install apache2 - Install apache 2 web server
- sudo service apache2 start - Start apache 2 web server
- echo "Hello World" > /var/www/html/index.html - Write to index.html
- \$(hostname) - Get host name
- \$(hostname -I) - Get host internal IP address

Internal and External IP Addresses

- External (Public) IP addresses are **Internet addressable**.
- Internal (Private) IP addresses are **internal** to a corporate network
- You CANNOT have two resources with same public (External) IP address.
 - HOWEVER, two different corporate networks CAN have resources with same Internal (private) IP address
- All **VM instances** are assigned at least one Internal IP address
- Creation of External IP addresses can be enabled for VM instances
 - (Remember) When you stop an VM instance, External IP address is lost
- **DEMO:** VM instances - Internal and External IPs



Static IP Addresses

- Scenario : How do you get a constant External IP address for a VM instance?
 - Quick and dirty way is to assign an Static IP Address to the VM!
- DEMO: Using Static IP Address with an VM instance



Static IP Addresses - Remember

- Static IP can be switched to another VM instance in same project
- Static IP remains attached even if you stop the instance. You have to manually detach it.
- Remember : You are **billed for** an Static IP when **you are NOT using it!**
 - Make sure that you explicitly release an Static IP when you are not using it.



Simplify VM HTTP server setup

- How do we **reduce** the number of steps in creating an VM instance and setting up a HTTP Server?
- Let's explore a few options:
 - Startup script
 - Instance Template
 - Custom Image



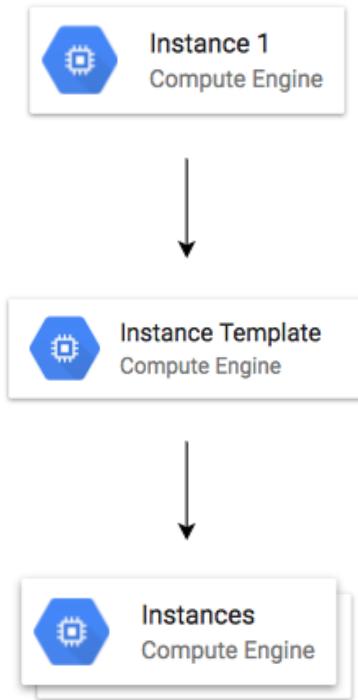
Bootstrapping with Startup script

```
#!/bin/bash
apt update
apt -y install apache2
echo "Hello world from $(hostname) $(hostname -I)" > /var/www/httr
```

- **Bootstrapping:** Install OS patches or software when an VM instance is launched.
- In VM, you can configure **Startup script** to bootstrap
- **DEMO** - Using Startup script

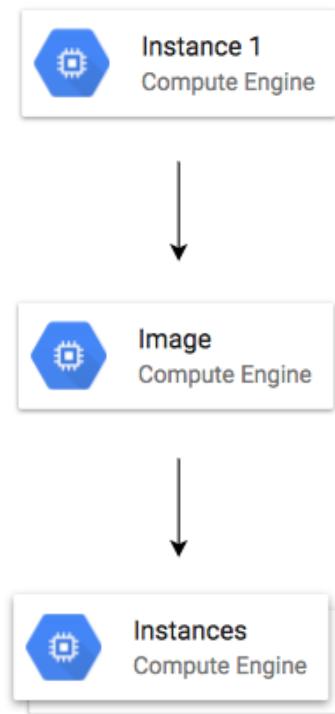
Instance templates

- Why do you need to specify all the VM instance details (Image, instance type etc) **every time** you launch an instance?
 - How about creating a Instance template?
 - Define machine type, image, labels, startup script and other properties
- Used to create **VM instances** and **managed instance groups**
 - Provides a convenient way to create similar instances
- **CANNOT** be updated
 - To make a change, copy an existing template and modify it
- (Optional) Image family can be specified (example - debian-9):
 - Latest non-deprecated version of the family is used
- **DEMO** - Launch VM instances using Instance templates



Reducing Launch Time with Custom Image

- Installing OS patches and software at launch of VM instances **increases boot up time**
- How about creating a custom image with OS patches and software **pre-installed?**
 - Can be created from an instance, a persistent disk, a snapshot, another image, or a file in Cloud Storage
 - Can be shared across projects
 - (Recommendation) Deprecate old images (& specify replacement image)
 - (Recommendation) **Hardening an Image** - Customize images to your corporate security standards
- **Prefer using Custom Image to Startup script**
- **DEMO** : Create a Custom Image and using it in an Instance Template



Compute Engine Scenarios

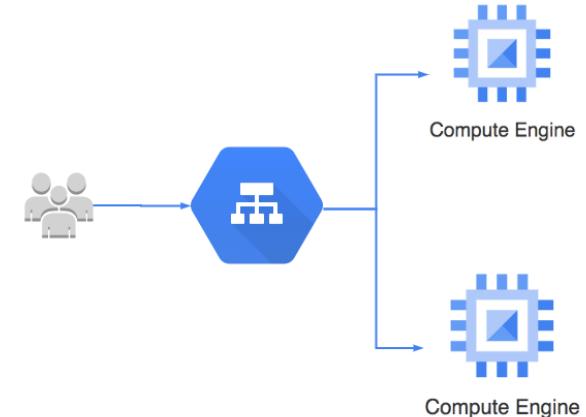
In 28
Minutes

Scenario	Solution
What are the pre-requisites to be able to create a VM instance?	<ol style="list-style-type: none">1. Project2. Billing Account3. Compute Engines APIs should be enabled
You want dedicated hardware for your compliance, licensing, and management needs	Sole-tenant nodes
I have 1000s of VM and I want to automate OS patch management, OS inventory management and OS configuration management (manage software installed)	Use "VM Manager"
You want to login to your VM instance to install software	You can SSH into it
You do not want to expose a VM to internet	Do NOT assign an external IP Address
You want to allow HTTP traffic to your VM	Configure Firewall Rules

Instance Groups

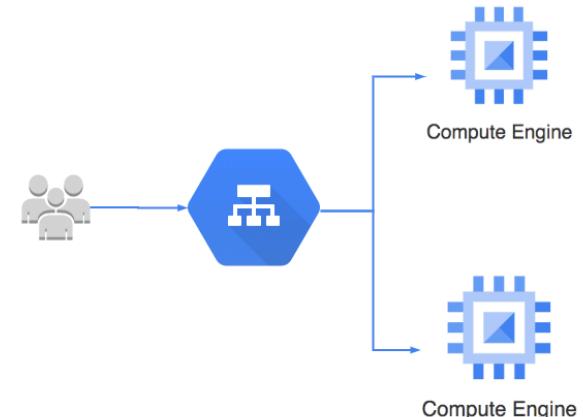
Instance Groups

- How do you create a group of VM instances?
 - **Instance Group** - Group of VM instances managed as a single entity
 - Manage group of similar VMs having similar lifecycle as **ONE UNIT**
- **Two Types of Instance Groups:**
 - **Managed** : Identical VMs created using a template:
 - Features: Auto scaling, auto healing and managed releases
 - **Unmanaged** : Different configuration for VMs in same group:
 - Does NOT offer auto scaling, auto healing & other services
 - NOT Recommended unless you need different kinds of VMs
- **Location** can be Zonal or Regional
 - Regional gives you higher availability (RECOMMENDED)



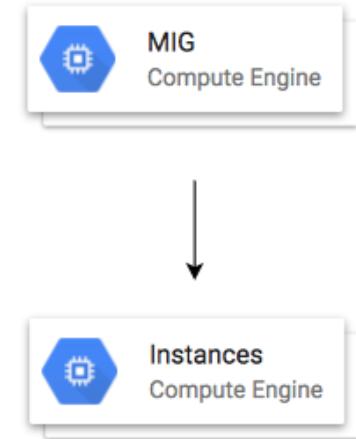
Managed Instance Groups (MIG)

- Managed Instance Group - Identical VMs created using an **instance template**
- Important Features:
 - **Maintain** certain number of instances
 - If an instance crashes, MIG launches another instance
 - **Detect application failures** using health checks (**Self Healing**)
 - Increase and decrease instances based on load (**Auto Scaling**)
 - Add **Load Balancer** to distribute load
 - Create instances in multiple zones (regional MIGs)
 - Regional MIGs provide higher availability compared to zonal MIGs
 - **Release new application versions without downtime**
 - **Rolling updates:** Release new version step by step (gradually). Update a percentage of instances to the new version at a time.
 - **Canary Deployment:** Test new version with a group of instances before releasing it across all instances.



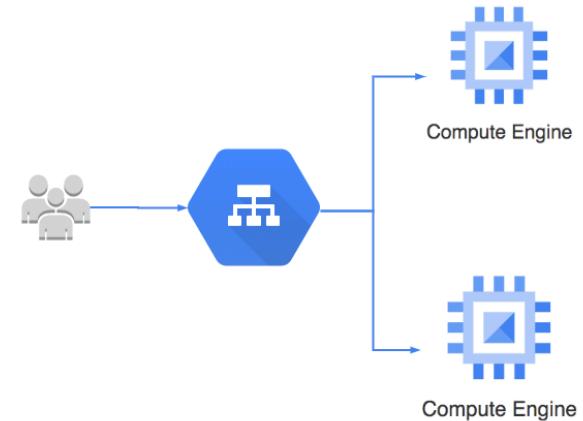
Creating Managed Instance Group (MIG)

- **Instance template** is mandatory
- Configure **auto-scaling** to automatically adjust number of instances based on load:
 - **Minimum** number of instances
 - **Maximum** number of instances
 - **Autoscaling metrics:** CPU Utilization target or Load Balancer Utilization target or Any other metric from Stack Driver
 - **Cool-down period:** How long to wait before looking at auto scaling metrics again?
 - **Scale In Controls:** Prevent a sudden drop in no of VM instances
 - **Example:** Don't scale in by more than 10% or 3 instances in 5 minutes
 - **Autohealing:** Configure a Health check with Initial delay (How long should you wait for your app to initialize before running a health check?)
- Time for a Demo



Updating a Managed Instance Group (MIG)

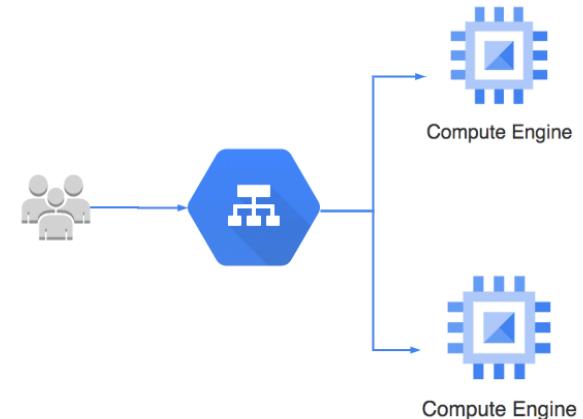
- **Rolling update** - Gradual update of instances in an instance group to the new instance template
 - Specify new template:
 - (OPTIONAL) Specify a template for canary testing
 - Specify how you want the update to be done:
 - When should the update happen?
 - Start the update immediately (Proactive) or when instance group is resized later(Opportunistic)
 - How should the update happen?
 - Maximum surge: How many instances are added at any point in time?
 - Maximum unavailable: How many instances can be offline during the update?
- **Rolling Restart/replace:** Gradual restart or replace of all instances in the group
 - No change in template BUT replace/restart existing VMs
 - Configure Maximum surge, Maximum unavailable and What you want to do? (Restart/Replace)



Cloud Load Balancing

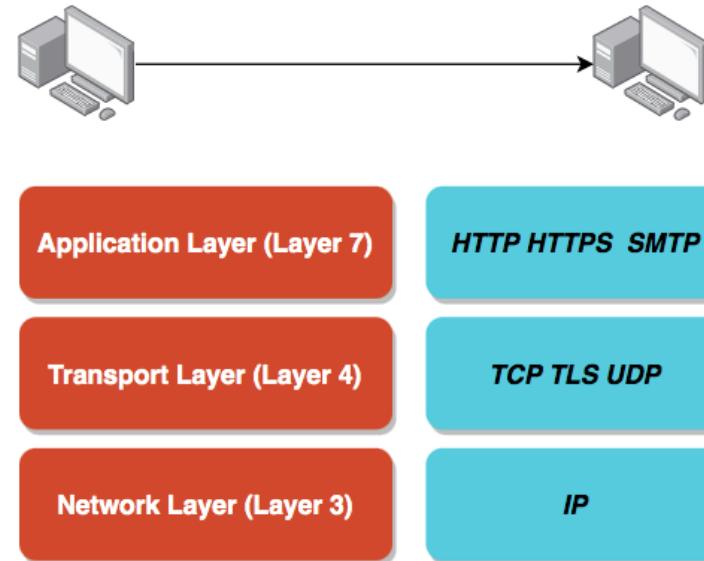
Cloud Load Balancing

- Distributes user traffic across instances of an application in single region or multiple regions
 - Fully distributed, software defined managed service
 - Important Features:
 - Health check - Route to healthy instances
 - Recover from failures
 - Auto Scaling
 - Global load balancing with single anycast IP
 - Also supports internal load balancing
- Enables:
 - High Availability
 - Auto Scaling
 - Resiliency



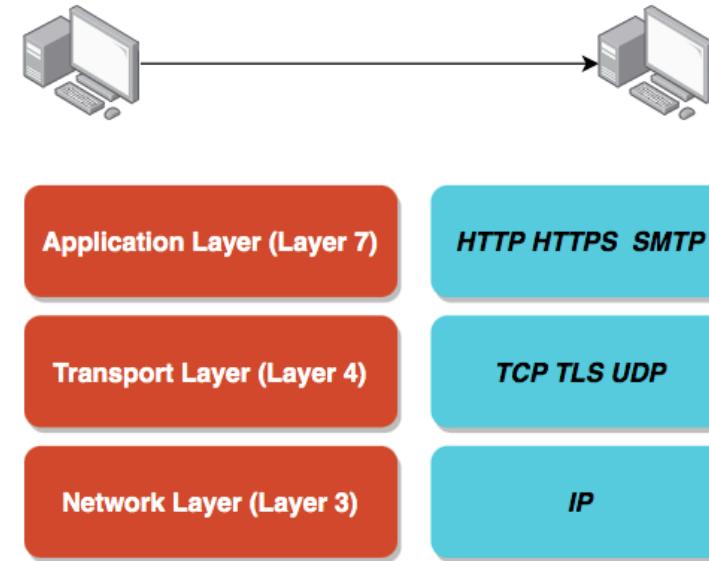
HTTP vs HTTPS vs TCP vs TLS vs UDP

- Computers use protocols to communicate
- Multiple layers and multiple protocols
- **Network Layer** - Transfer bits and bytes
- **Transport Layer** - Are the bits and bytes transferred properly?
- **Application Layer** - Make REST API calls and Send Emails
- (Remember) Each layer makes use of the layers beneath it
- (Remember) Most applications talk at application layer. BUT some applications talk at transport layer directly (high performance).



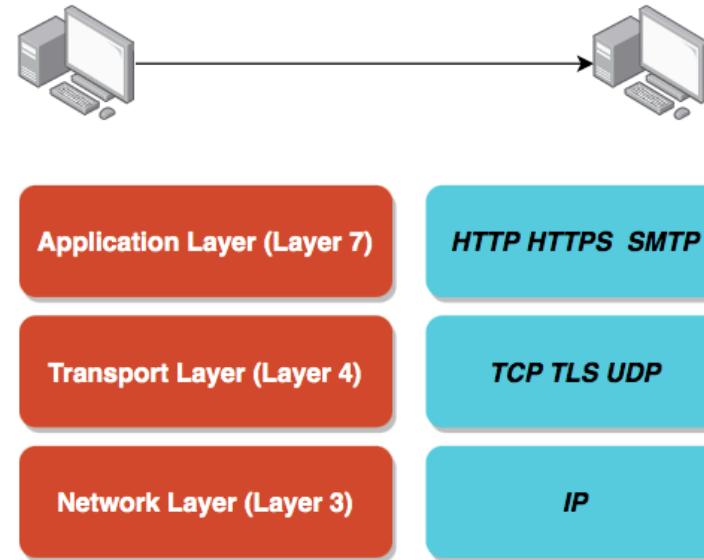
HTTP vs HTTPS vs TCP vs TLS vs UDP

- Network Layer:
 - IP (Internet Protocol): Transfer bytes. Unreliable.
- Transport Layer:
 - TCP (Transmission Control): Reliability > Performance
 - TLS (Transport Layer Security): Secure TCP
 - UDP (User Datagram Protocol): Performance > Reliability
- Application Layer:
 - HTTP(Hypertext Transfer Protocol): Stateless Request Response Cycle
 - HTTPS: Secure HTTP
 - SMTP: Email Transfer Protocol
 - and a lot of others...

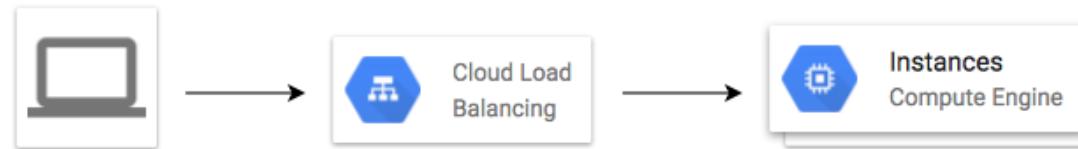


HTTP vs HTTPS vs TCP vs TLS vs UDP

- Most applications typically communicate at application layer
 - Web apps/REST API(HTTP/HTTPS), Email Servers(SMTP), File Transfers(FTP)
 - All these applications use TCP/TLS at network layer(for reliability)
- **HOWEVER** applications needing high performance directly communicate at transport layer:
 - Gaming applications and live video streaming use UDP (sacrifice reliability for performance)
- **Objective:** Understand Big Picture. Its OK if you do not understand all details.

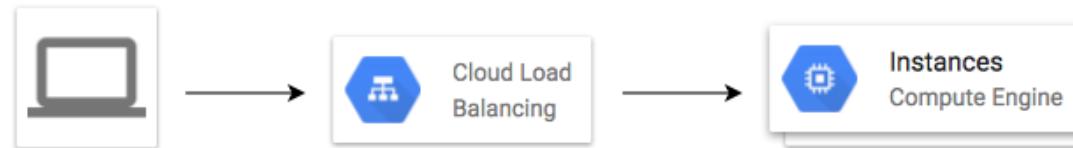


Cloud Load Balancing - Terminology



- **Backend** - Group of endpoints that receive traffic from a Google Cloud load balancer (example: instance groups)
- **Frontend** - Specify an IP address, port and protocol. This IP address is the frontend IP for your clients requests.
 - For SSL, a certificate must also be assigned.
- **Host and path rules (For HTTP(S) Load Balancing)** - Define rules redirecting the traffic to different backends:
 - Based on **path** - `in28minutes.com/a` vs `in28minutes.com/b`
 - Based on **Host** - `a.in28minutes.com` vs `b.in28minutes.com`
 - Based on **HTTP headers** (Authorization header) and methods (POST, GET, etc)

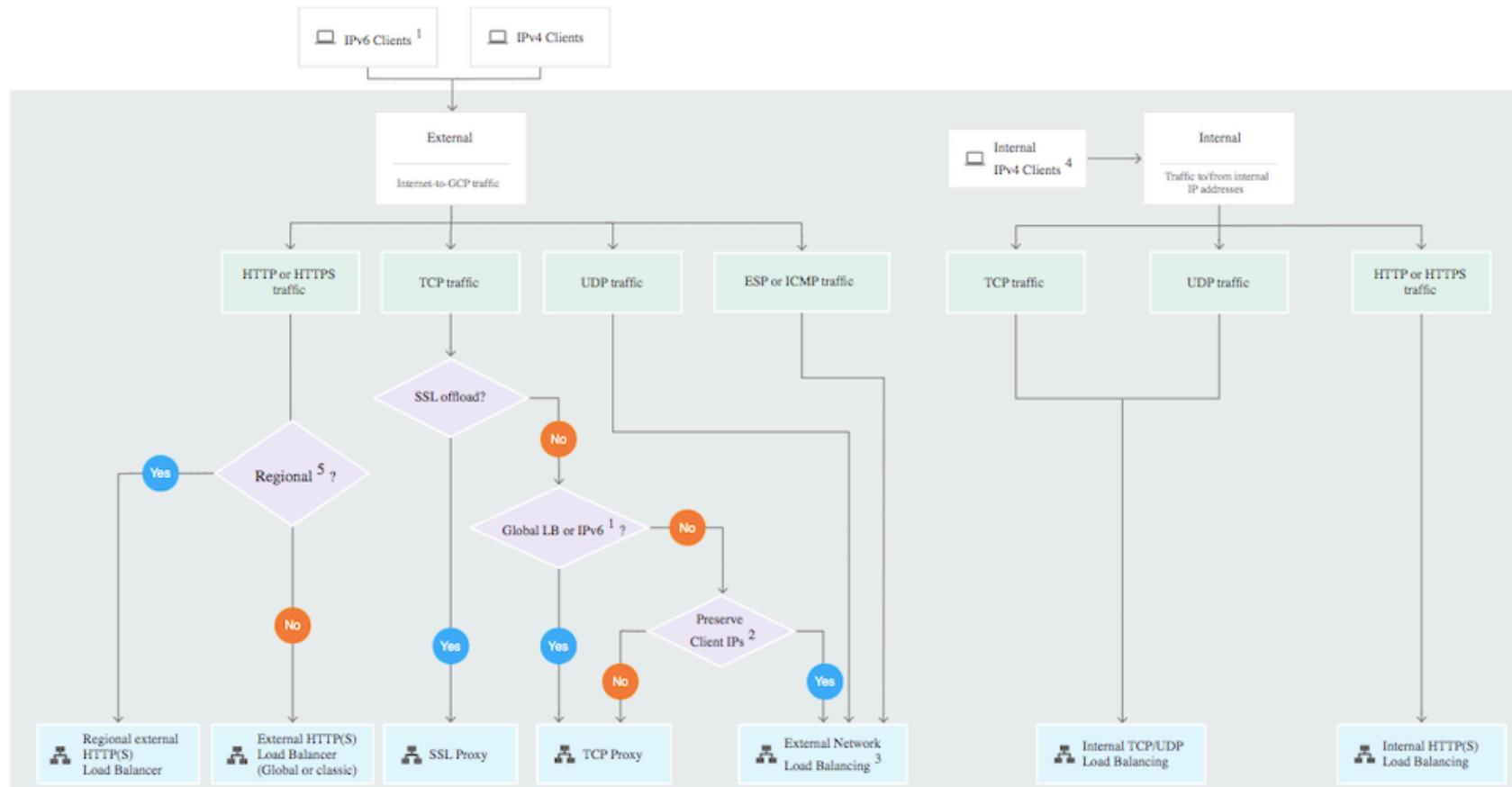
Load Balancing - SSL/TLS Termination/Offloading



- Client to Load Balancer: Over internet
 - HTTPS recommended
- Load Balancer to VM instance: Through Google internal network
 - HTTP is ok. HTTPS is preferred.
- SSL/TLS Termination/Offloading
 - Client to Load Balancer: HTTPS/TLS
 - Load Balancer to VM instance: HTTP/TCP

Cloud Load Balancing - Choosing Load Balancer

<https://cloud.google.com/load-balancing/images/choose-lb.svg>

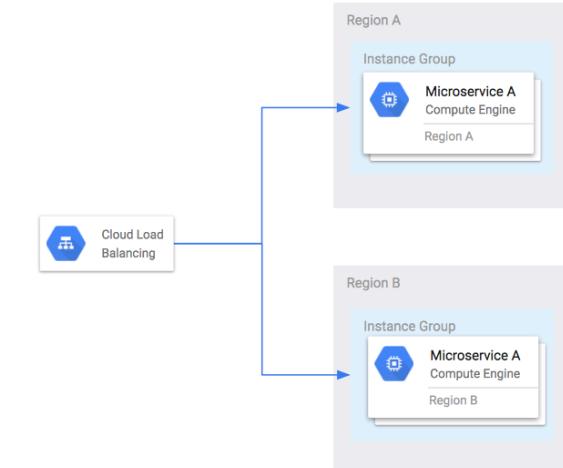


Cloud Load Balancing - Features

Load Balancer	Type of Traffic	Proxy or pass-through	Destination Ports
External HTTP(S)	Global, External, HTTP or HTTPS	Proxy	HTTP on 80 or 8080 HTTPS on 443
Internal HTTP(S)	Regional, Internal, HTTP or HTTPS	Proxy	HTTP on 80 or 8080 HTTPS on 443
SSL Proxy	Global, External, TCP with SSL offload	Proxy	A big list
TCP Proxy	Global, External, TCP without SSL offload	Proxy	A big list
External Network TCP/UDP	Regional, External, TCP or UDP	Pass-through	any

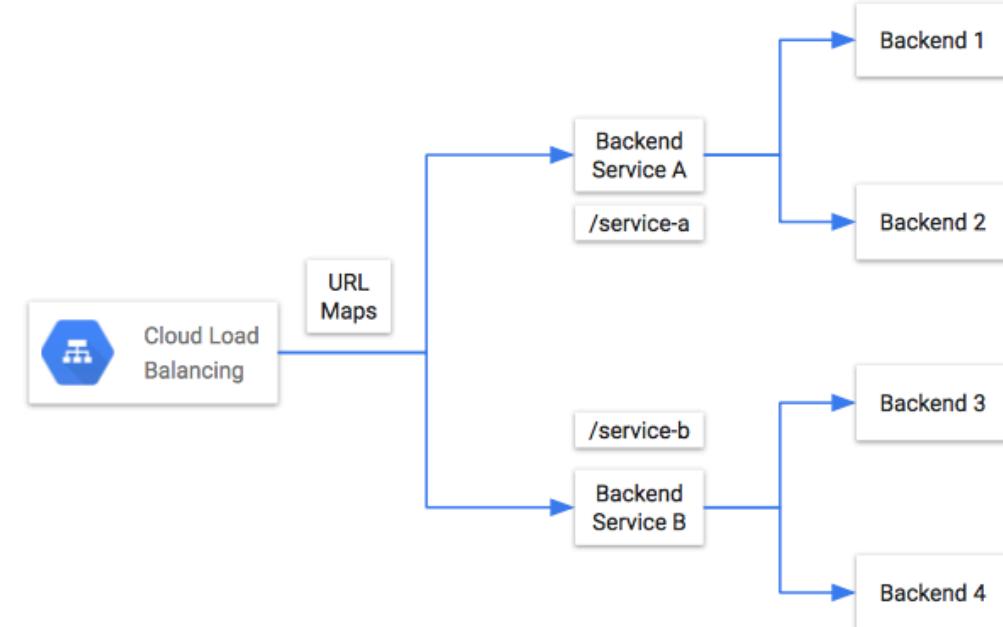
Load Balancing Across MIGs in Multiple Regions

- **Regional MIG** can distribute instances in different zones of a single region
 - Create multiple Regional MIGs in different regions (in the same project)
- **HTTP(S) Load Balancing** can distribute load to the multiple MIGs behind a single external IP address
 - User requests are redirected to the nearest region (Low latency)
- Load balancing sends traffic to **healthy instances**:
 - If health check fails instances are restarted:
 - (REMEMBER) Ensure that health check from load balancer can reach the instances in an instance group (Firewall rules)
 - If all backends within a region are unhealthy, traffic is distributed to healthy backends in other regions
 - Can contain preemptible instances as well!



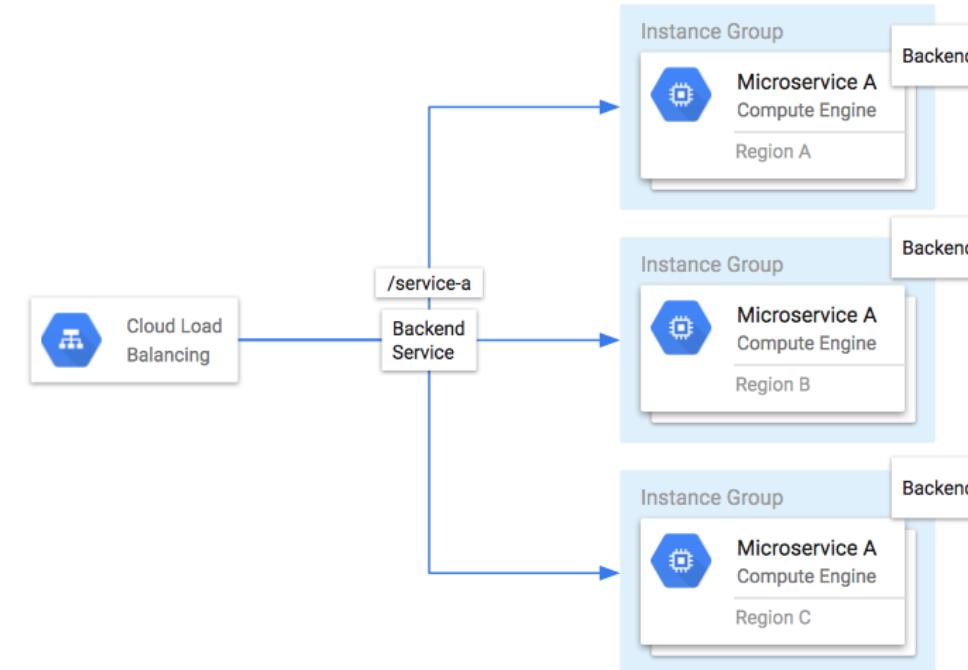
Quick Review of HTTP(S) Load Balancing Concepts

- **Backend Service** - Group of backends or a bucket
- **Backend** - A Managed Instance Group, for example
- **URL Maps** - Route requests to backend services or backend buckets
 - URL /service-a maps to **Backend Service A**
 - URL /service-b maps to **Backend Service B**
- (Remember) Each Backend Service can have multiple backends in multiple regions



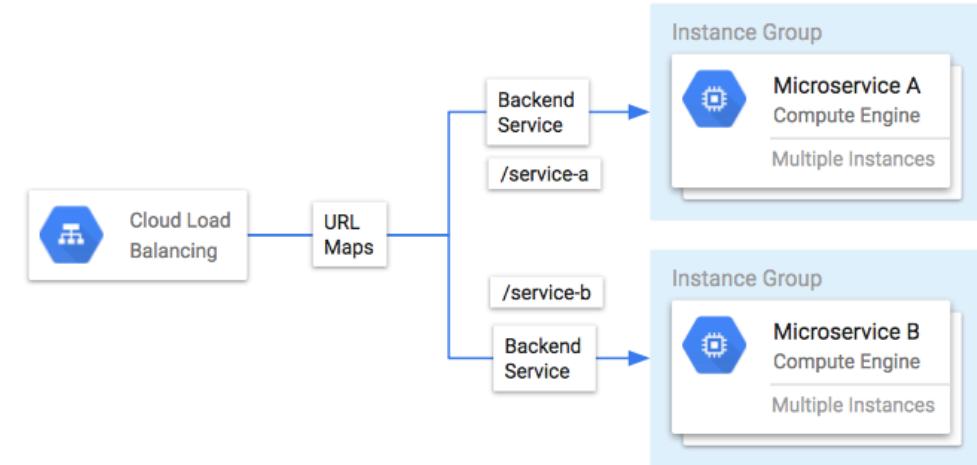
HTTP(S) Load Balancing - 1 - Multi Regional Microservice

- **Backend Services**
 - One Backend Service for the Microservice
- **Backends**
 - Multiple backends for each microservice MIG in each region
- **URL Maps**
 - URL `/service-a` maps to the microservice **Backend Service**
- **Global routing:** Route user to nearest instance (nearest regional MIG)
 - Needs Networking Premium Tier:
 - As in STANDARD Tier:
 - Forwarding rule and its external IP address are regional
 - All backends for a backend service must be in the same region as the forwarding rule



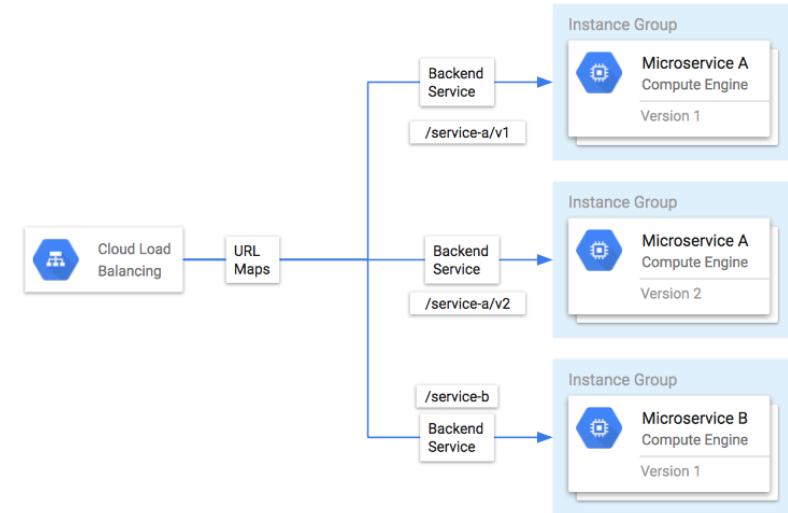
HTTP(S) Load Balancing - 2 - Multiple Microservices

- **Backend Services**
 - One Backend Service for each the Microservice
- **Backends**
 - Each microservice can have multiple backend MIGs in different regions
 - In the example, we see one backend per microservice
- **URL Maps**
 - URL **/service-a** => Microservice A Backend Service
 - URL **/service-b** => Microservice B Backend Service



HTTP(S) Load Balancing - 3 - Microservice Versions

- **Backend Services**
 - A Backend Service for each Microservice version
- **Backends**
 - Each microservice version can have multiple backend MIGs in different regions
 - In the example, we see one backend per microservice version
- **URL Maps**
 - `/service-a/v1` => Microservice A V1 Backend Service
 - `/service-a/v2` => Microservice A V2 Backend Service
 - `/service-b` => Microservice B Backend Service
- **Flexible Architecture:** With HTTP(S) load balancing, route **global requests** across multiple versions of multiple microservices!



Compute Engine & Load Balancing FOR ARCHITECTS

Compute Engine & Load Balancing for Architects

In 28
Minutes

It is not sufficient to get things working. We want more!

- Build Resiliency
- Increase Availability
- Increase Scalability
- Improve Performance
- Improve Security
- Lower Costs
- and

Professional Cloud Architect vs Associate Cloud Engineer

- **Associate Cloud Engineer**
 - Focused on tasks that Cloud Engineers perform in day to day job!
- **Professional Cloud Architect**
 - Understand business and technical requirements
 - Design cloud solutions that meet your functional and non-functional needs
- **GCP Services** are the same:
 - BUT your perspective should be different
 - With **Professional Cloud Architect**
 - You need to know the services
 - AND learn to build highly resilient, highly available, scalable, secure, performant solutions that have low cost!
 - Sounds Complex??
 - (Don't worry) We will understand each of these as we go further
 - Availability, Scalability, Resilience etc..



Google Cloud

What is Availability?

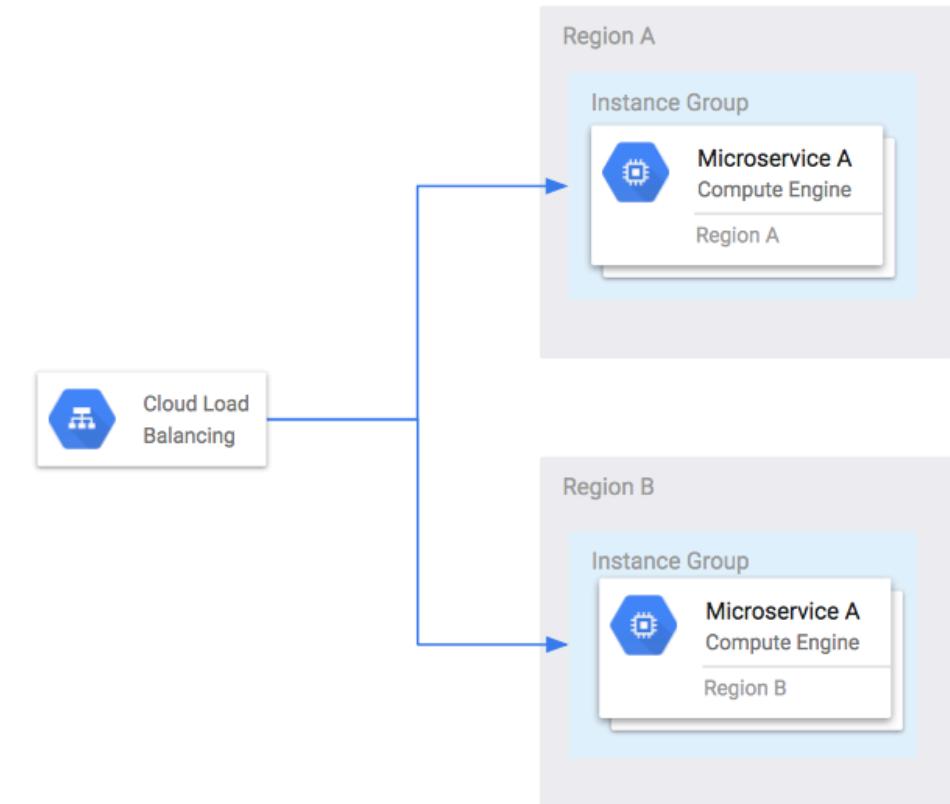
- Are the applications available **when the users need them?**
- **Percentage of time** an application provides the operations expected of it
- **Example:** 99.99% availability. Also called four 9's availability

Availability Table

Availability	Downtime (in a month)	Comment
99.95%	22 minutes	
99.99% (four 9's)	4 and 1/2 minutes	Most online apps aim for 99.99% (four 9's)
99.999% (five 9's)	26 seconds	Achieving 5 9's availability is tough

High Availability for Compute Engine & Load Balancing

- **Highly Available Architecture:**
 - Multiple Regional Instance Groups for each Microservice
 - Distribute Load using a Global HTTPS Load Balancing
 - Configure Health Checks for Instance Group and Load Balancing
 - Enable Live Migration for VM instances

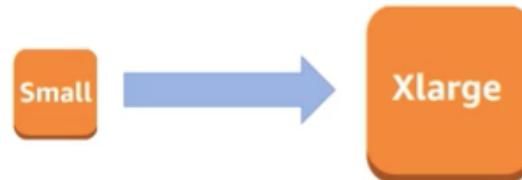


- **Advantages:**
 - Instances distributed across regions
 - Even if a region is down, your app is available
 - Global Load Balancing is highly available
 - Health checks ensure auto healing

What is Scalability?

- A system is handling 1000 transactions per second. Load is expected to increase 10 times in the next month
 - Can we handle a growth in users, traffic, or data size without any drop in performance?
 - Does ability to serve more growth increase proportionally with resources?
- Ability to **adapt** to changes in demand (users, data)
- What are the options that can be considered?
 - Deploy to a bigger instance with bigger CPU and more memory
 - Increase the number of application instances and setup a load balancer
 - And a lot more.

What is Vertical Scaling?



- Deploying application/database to **bigger instance**:
 - A larger hard drive
 - A faster CPU
 - More RAM, CPU, I/O, or networking capabilities
- There are limits to vertical scaling

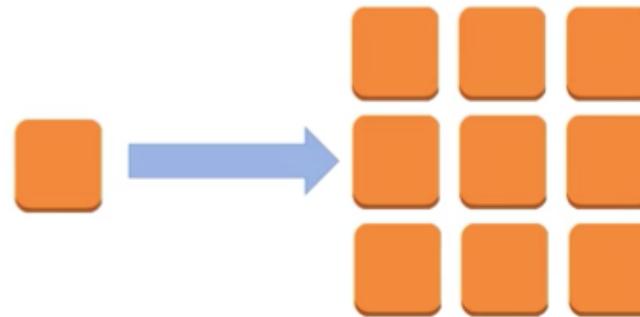
Vertical Scaling for GCE VMs

Machine name	vCPUs ¹	Memory (GB)	Max number of persistent disks (PDs) ²	Max total PD size (TB)	Local SSD	Maximum egress bandwidth (Gbps) ³
e2-standard-2	2	8	128	257	No	4
e2-standard-4	4	16	128	257	No	8
e2-standard-8	8	32	128	257	No	16
e2-standard-16	16	64	128	257	No	16
e2-standard-32	32	128	128	257	No	16

- Increasing VM machine size:
 - e2-standard-2 to e2-standard-4 or
 - e2-standard-16 to e2-standard-32 or
 - ...

What is Horizontal Scaling?

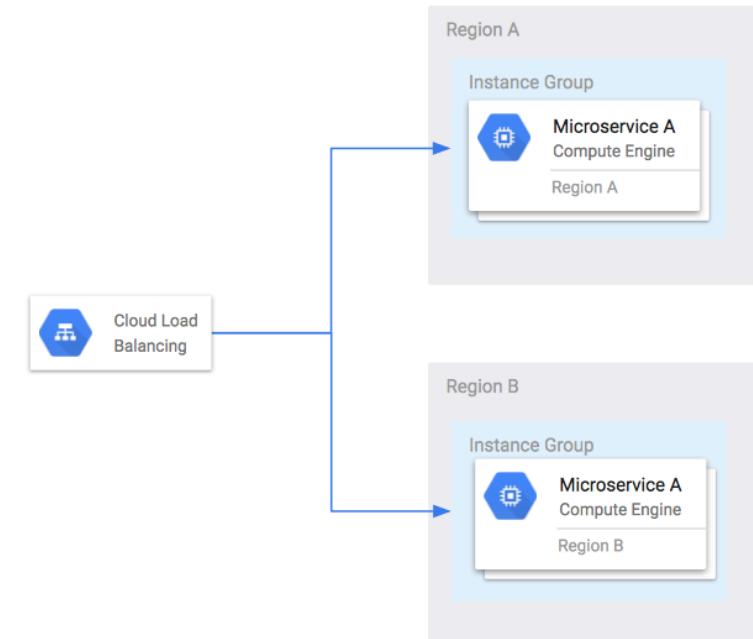
In 28
Minutes



- Deploying multiple instances of application/database
- (Typically but not always) Horizontal Scaling is preferred to Vertical Scaling:
 - Vertical scaling has limits
 - Vertical scaling can be expensive
 - Horizontal scaling increases availability
- (BUT) Horizontal Scaling needs additional infrastructure:
 - Load Balancers etc.

Horizontal Scaling for GCE VMs

- Distribute VM instances
 - in a single zone
 - in multiple zones in single region
 - in multiple zones across multiple regions
- Auto scale: Managed Instance Group (s)
- Distribute load : Load Balancing



Compute Engine : Live Migration & Availability Policy

- How do you keep your VM instances running when a host system needs to be updated (a software or a hardware update needs to be performed)?
- **Live Migration**
 - Your running instance is migrated to another host in the same zone
 - Does NOT change any attributes or properties of the VM
 - SUPPORTED for instances with local SSDs
 - NOT SUPPORTED for GPUs and preemptible instances
- **Important Configuration - Availability Policy:**
 - **On host maintenance:** What should happen during periodic infrastructure maintenance?
 - Migrate (default): Migrate VM instance to other hardware
 - Terminate: Stop the VM instance
 - **Automatic restart** - Restart VM instances if they are terminated due to non-user-initiated reasons (maintenance event, hardware failure etc.)

Compute Engine Features: GPUs

- How do you accelerate math intensive and graphics-intensive workloads for AI/ML etc?
- Add a **GPU** to your virtual machine:
 - High performance for math intensive and graphics-intensive workloads
 - Higher Cost
 - (REMEMBER) Use **images with GPU libraries** (Deep Learning) installed
 - OTHERWISE, GPU will not be used
 - **GPU restrictions:**
 - NOT supported on all machine types (For example, not supported on shared-core or memory-optimized machine types)
 - On host maintenance can only have the value "Terminate VM instance"
- Recommended **Availability policy** for GPUs
 - Automatic restart - on



Security

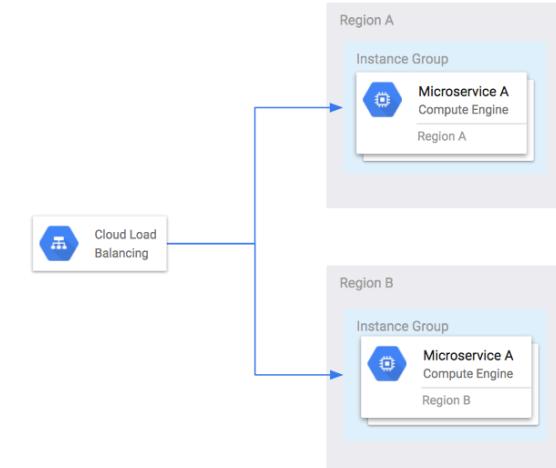
- Use **Firewall Rules** to restrict traffic
- Use **Internal IP Addresses** as much as possible
- Use **Sole-tenant nodes** when you have regulatory needs
- Create a hardened **custom image** to launch your VMs

Performance

- Choose right **Machine Family** for your workload
- Use GPUs and TPUs to increase performance
 - Use GPUs to accelerate machine learning and data processing workloads
 - Use TPUs for massive matrix operations performed in your machine learning workloads
- Prefer creating a hardened **custom image** to installing software at startup

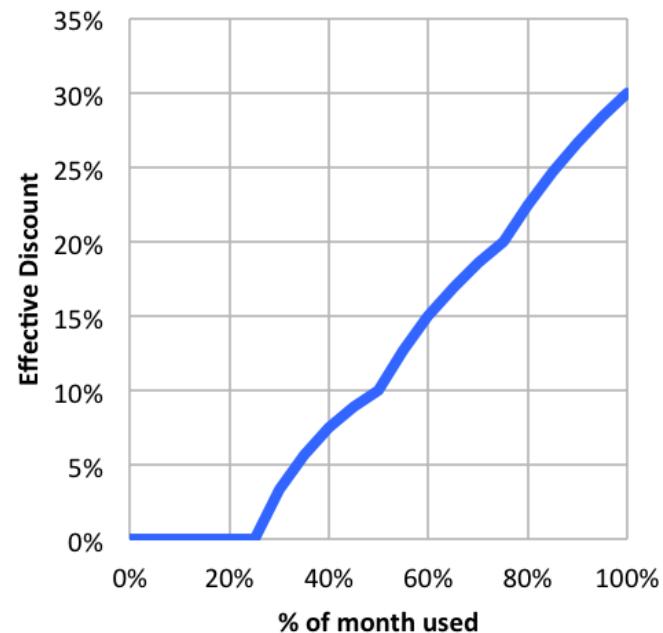
Resiliency for Compute Engine & Load Balancing

- **Resiliency** - "Ability of system to provide acceptable behavior even when one or more parts of the system fail"
- Build Resilient Architectures
 - Run VMs in MIG behind global load balancing
- Have the right data available
 - Use Cloud Monitoring for monitoring
 - Install logging agent to send logs to Cloud Logging
- Be prepared for the unexpected (and changes)
 - Enable Live Migration and Automatic restart when available
 - Configure the right **health checks**
 - (Disaster recovery) Upto date image copied to multiple regions
- We will talk about resiliency as we go further!



Sustained use discounts

- **Automatic discounts** for running VM instances for significant portion of the billing month
 - Example: If you use N1, N2 machine types for more than 25% of a month, you get a 20% to 50% discount on every incremental minute.
 - Discount increases with usage (graph)
 - No action required on your part!
- **Applicable** for instances created by **Google Kubernetes Engine and Compute Engine**
- **RESTRICTION:** Does NOT apply on certain machine types (example: E2 and A2)
- **RESTRICTION:** Does NOT apply to VMs created by App Engine flexible and Dataflow



Source: <https://cloud.google.com>

Committed use discounts

- For workloads with predictable resource needs
- Commit for 1 year or 3 years
- Up to 70% discount based on machine type and GPUs
- Applicable for instances created by Google Kubernetes Engine and Compute Engine
- RESTRICTION: Does NOT apply to VMs created by App Engine flexible and Dataflow



Compute
Engine

Preemptible VM

- **Short-lived cheaper** (upto 80%) compute instances
 - Can be stopped by GCP any time (preempted) within 24 hours
 - Instances get 30 second warning (to save anything they want to save)
- **Use Preempt VM's if:**
 - Your applications are **fault tolerant**
 - You are very **cost sensitive**
 - Your workload is **NOT immediate**
 - Example: Non immediate batch processing jobs
- **RESTRICTIONS:**
 - NOT always available
 - NO SLA and CANNOT be migrated to regular VMs
 - NO Automatic Restarts
 - Free Tier credits not applicable



Spot VMs

- **Spot VMs:** Latest version of preemptible VMs
- **Key Difference:** Does not have a maximum runtime
 - Compared to traditional preemptible VMs which have a maximum runtime of 24 hours
- **Other features similar to traditional preemptible VMs**
 - May be reclaimed at any time with 30-second notice
 - NOT always available
 - Dynamic Pricing: 60 - 91% discount compared to on-demand VMs
 - Free Tier credits not applicable



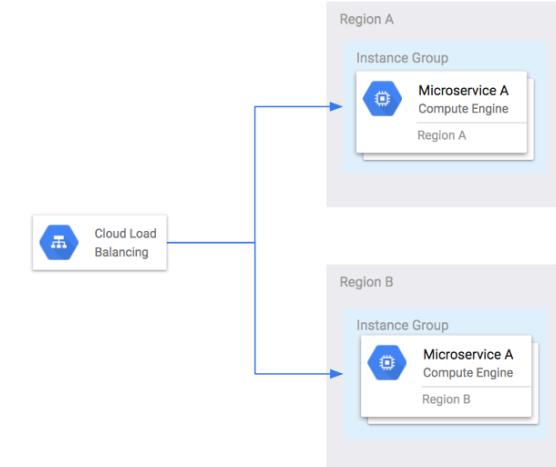
Google Compute Engine - Billing

- You are **billed by the second** (after a minimum of 1 minute)
- You are NOT billed for compute when a compute instance is stopped
 - However, you will be billed for any storage attached with it!
- (RECOMMENDATION) **Always create Budget alerts** and make use of Budget exports to stay on top of billing!
- What are the ways you can save money?
 - Choose the right machine type and image for your workload
 - Be aware of the discounts available:
 - Sustained use discounts
 - Committed use discounts
 - Discounts for preemptible VM instances



Compute Engine & Load Balancing - Cost Efficiency

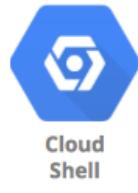
- Use Auto Scaling
 - Have optimal number and type of VM instances running
- Understand Sustained use discounts
- Make use of Committed use discounts for predictable long term workloads
- Use Preemptible VMs for non critical fault tolerant workloads



Gcloud

Gcloud

- Command line interface to interact with Google Cloud Resources
- Most GCP services can be managed from CLI using Gcloud:
 - Compute Engine Virtual Machines
 - Managed Instance Groups
 - Databases
 - and ... many more
- You can create/delete/update/read existing resources and perform actions like deployments as well!
- (REMEMBER) SOME GCP services have specific CLI tools:
 - Cloud Storage - gsutil
 - Cloud BigQuery - bq
 - Cloud Bigtable - cbt
 - Kubernetes - kubectl (in addition to Gcloud which is used to manage clusters)



Installation

- Gcloud is part of Google Cloud SDK
 - Cloud SDK requires Python
 - Instructions to install Cloud SDK (and Gcloud) => <https://cloud.google.com/sdk/docs/install>
- You can also use Gcloud on Cloud Shell

Connecting to GCP

- **gcloud init** - initialize or reinitialize gcloud
 - Authorize gcloud to use your user account credentials
 - Setup configuration
 - Includes current project, default zone etc
- **gcloud config list** - lists all properties of the active configuration

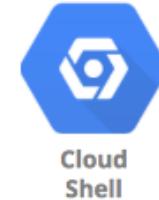
gcloud command structure - Playing with Services

- **gcloud GROUP SUBGROUP ACTION ...**
 - GROUP - config or compute or container or dataflow or functions or iam or ..
 - Which service group are you playing with?
 - SUBGROUP - instances or images or instance-templates or machine-types or regions or zones
 - Which sub group of the service do you want to play with?
 - ACTION - create or list or start or stop or describe or ...
 - What do you want to do?
- **Examples:**
 - gcloud compute instances list
 - gcloud compute zones list
 - gcloud compute regions list
 - gcloud compute machine-types list
 - gcloud compute machine-types list --filter="zone:us-central1-b"
 - gcloud compute machine-types list --filter="zone:(us-central1-b europe-west1-d)"

Cloud Shell

In 28
Minutes

- **Important things you need to know about Cloud Shell:**
 - Cloud Shell is backed by a VM instance (automatically provisioned by Google Cloud when you launch Cloud Shell)
 - 5 GB of free persistent disk storage is provided as your \$HOME directory
 - Prepackaged with latest version of Cloud SDK, Docker etc
 - (Remember) Files in your home directory persist between sessions (scripts, user configuration files like .bashrc and .vimrc etc)
 - Instance is terminated if you are inactive for more than 20 minutes
 - Any modifications that you made to it outside your \$HOME will be lost
 - (Remember) After 120 days of inactivity, even your \$HOME directory is deleted
 - Cloud Shell can be used to SSH into virtual machines using their private IP addresses



Managed Services

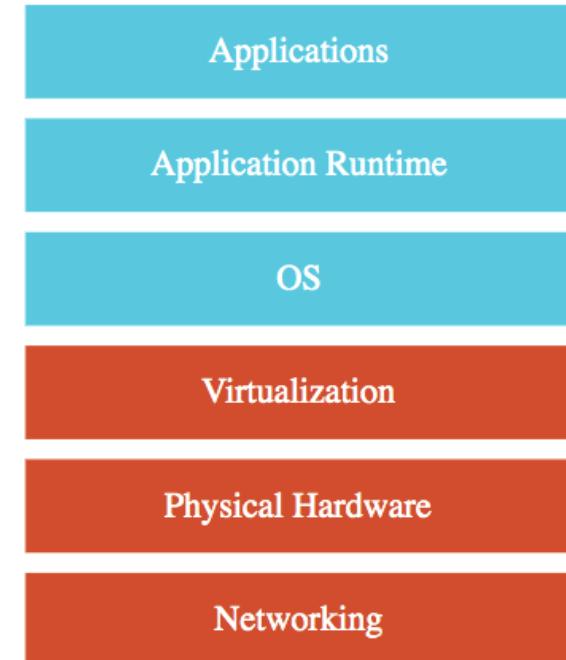
Managed Services

- Do you want to continue running applications in the cloud, the same way you run them in your data center?
- OR are there OTHER approaches?
- You should understand some terminology used with cloud services:
 - IaaS (Infrastructure as a Service)
 - PaaS (Platform as a Service)
 - FaaS (Function as a Service)
 - CaaS (Container as a Service)
 - Serverless
- Let's get on a quick journey to understand these!



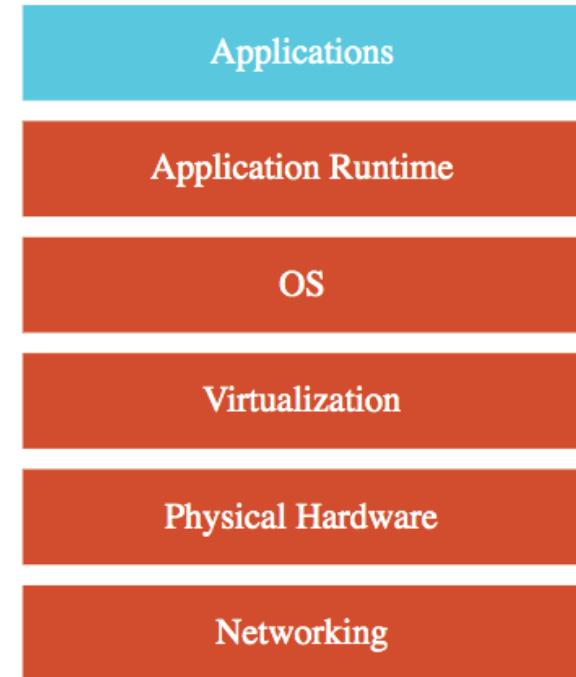
IAAS (Infrastructure as a Service)

- Use **only infrastructure** from cloud provider
- **Example:** Using VM to deploy your applications or databases
- You are responsible for:
 - Application Code and Runtime
 - Configuring load balancing
 - Auto scaling
 - OS upgrades and patches
 - Availability
 - etc.. (and a lot of things!)

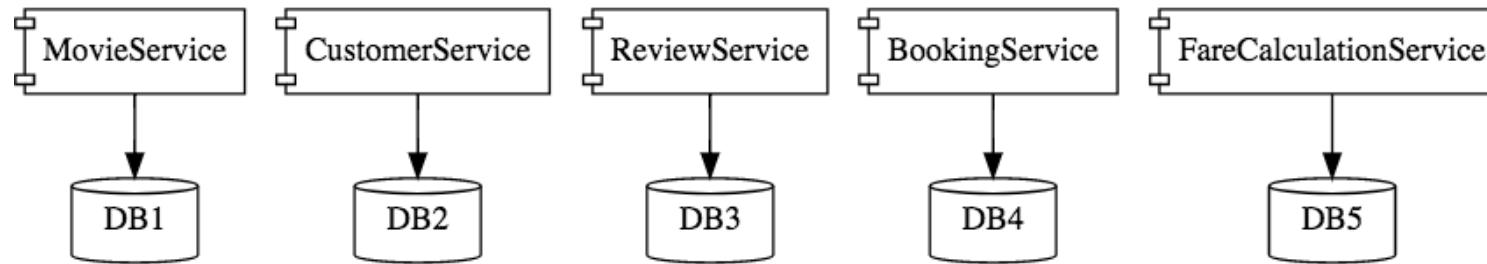


PAAS (Platform as a Service)

- Use a platform provided by cloud
- **Cloud provider** is responsible for:
 - OS (incl. upgrades and patches)
 - Application Runtime
 - Auto scaling, Availability & Load balancing etc..
- **You are responsible for:**
 - Configuration (of Application and Services)
 - Application code (if needed)
- Varieties:
 - **CAAS (Container as a Service):** Containers instead of Apps
 - **FAAS (Function as a Service):** Functions instead of Apps
 - Databases - Relational & NoSQL (Amazon RDS, Google Cloud SQL, Azure SQL Database etc), Queues, AI, ML, Operations etc!



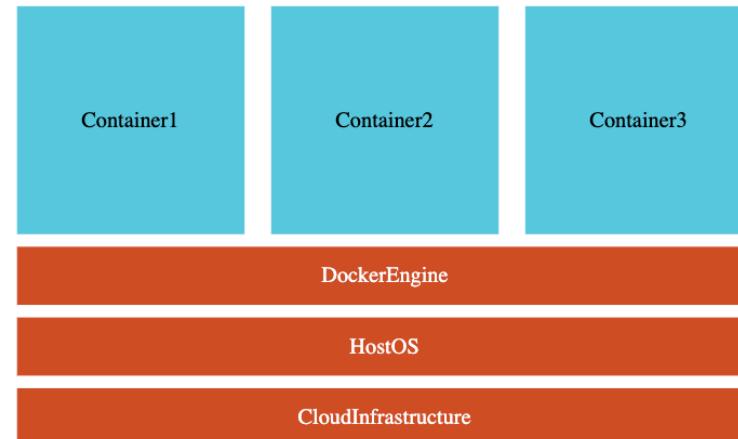
Microservices



- Enterprises are heading towards microservices architectures
 - Build small focused microservices
 - **Flexibility to innovate** and build applications in different programming languages (Go, Java, Python, JavaScript, etc)
- **BUT deployments become complex!**
- How can we have **one way of deploying** Go, Java, Python or JavaScript .. microservices?
 - Enter containers!

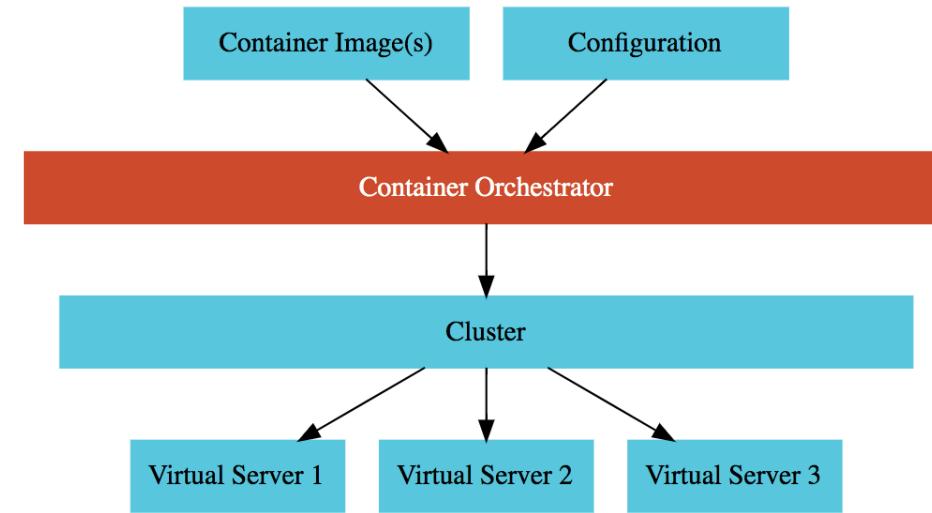
Containers - Docker

- Create Docker images for each microservice
- Docker image **has all needs of a microservice:**
 - Application Runtime (JDK or Python or NodeJS)
 - Application code and Dependencies
- Runs **the same way** on any infrastructure:
 - Your local machine
 - Corporate data center
 - Cloud
- Advantages
 - Docker containers are **light weight**
 - Compared to Virtual Machines as they do not have a Guest OS
 - Docker provides **isolation** for containers
 - Docker is **cloud neutral**



Container Orchestration

- **Requirement :** I want 10 instances of Microservice A container, 15 instances of Microservice B container and
- **Typical Features:**
 - **Auto Scaling** - Scale containers based on demand
 - **Service Discovery** - Help microservices find one another
 - **Load Balancer** - Distribute load among multiple instances of a microservice
 - **Self Healing** - Do health checks and replace failing instances
 - **Zero Downtime Deployments** - Release new versions without downtime



- What do we think about when we develop an application?
 - Where to deploy? What kind of server? What OS?
 - How do we take care of scaling and availability of the application?
- **What if you don't need to worry about servers and focus on your code?**
 - Enter Serverless
 - Remember: Serverless does NOT mean "No Servers"
- **Serverless for me:**
 - You don't worry about infrastructure (ZERO visibility into infrastructure)
 - Flexible scaling and automated high availability
 - Most Important: Pay for use
 - Ideally ZERO REQUESTS => ZERO COST
- **You focus on code** and the cloud managed service takes care of all that is needed to scale your code to serve millions of requests!
 - And you pay for requests and NOT servers!

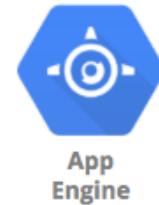
GCP Managed Services for Compute

Service	Details	Category
Compute Engine	High-performance and general purpose VMs that scale globally	IaaS
Google Kubernetes Engine	Orchestrate containerized microservices on Kubernetes Needs advanced cluster configuration and monitoring	CaaS
App Engine	Build highly scalable applications on a fully managed platform using open and familiar languages and tools	PaaS (CaaS, Serverless)
Cloud Functions	Build event driven applications using simple, single-purpose functions	FaaS, Serverless
Cloud Run	Develop and deploy highly scalable containerized applications. Does NOT need a cluster!	CaaS (Serverless)

App Engine

App Engine

- **Simplest way to deploy and scale your applications in GCP**
 - Provides end-to-end application management
- **Supports:**
 - Go, Java, .NET, Node.js, PHP, Python, Ruby using pre-configured runtimes
 - Use custom run-time and write code in any language
 - Connect to variety of Google Cloud storage products (Cloud SQL etc)
- **No usage charges - Pay for resources provisioned**
- **Features:**
 - Automatic load balancing & Auto scaling
 - Managed platform updates & Application health monitoring
 - Application versioning
 - Traffic splitting



Compute Engine vs App Engine

- **Compute Engine**

- IAAS
- MORE Flexibility
- MORE Responsibility
 - Choosing Image
 - Installing Software
 - Choosing Hardware
 - Fine grained Access/Permissions (Certificates/Firewalls)
 - Availability etc



App
Engine



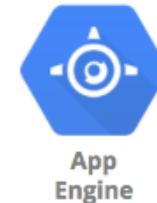
Compute
Engine

- **App Engine**

- PaaS
- Serverless
- LESSER Responsibility
- LOWER Flexibility

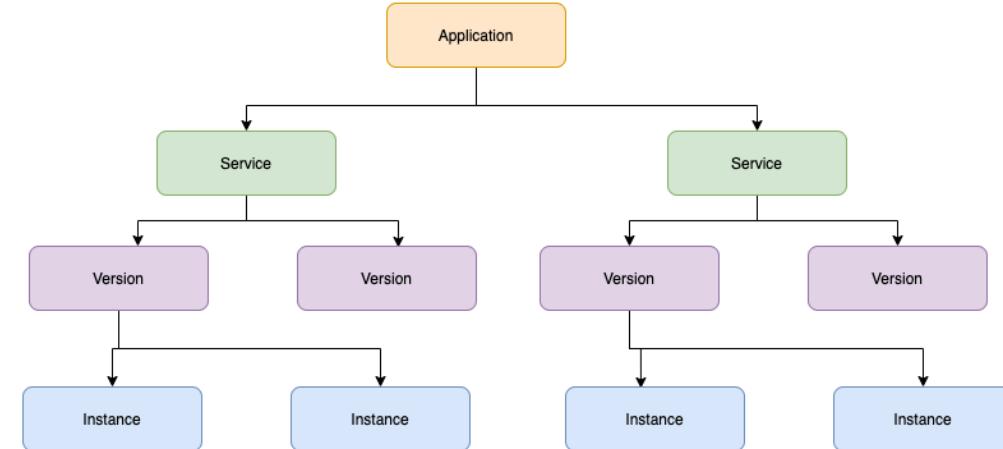
App Engine environments

- **Standard:** Applications run in language specific sandboxes
 - Complete isolation from OS/Disk/Other Apps
 - **V1:** Java, Python, PHP, Go (OLD Versions)
 - ONLY for Python and PHP runtimes:
 - Restricted network Access
 - Only white-listed extensions and libraries are allowed
 - No Restrictions for Java and Go runtimes
 - **V2:** Java, Python, PHP, Node.js, Ruby, Go (NEWER Versions)
 - Full Network Access and No restrictions on Language Extensions
- **Flexible** - Application instances run within Docker containers
 - Makes use of Compute Engine virtual machines
 - Support ANY runtime (with built-in support for Python, Java, Node.js, Go, Ruby, PHP, or .NET)
 - Provides access to background processes and local disks



App Engine - Application Component Hierarchy

- **Application:** One App per Project
- **Service(s):** Multiple Microservices or App components
 - You can have multiple services in a single application
 - Each **Service** can have different settings
 - Earlier called Modules
- **Version(s):** Each version associated with code and configuration
 - Each **Version** can run in one or more instances
 - Multiple versions can co-exist
 - Options to rollback and split traffic



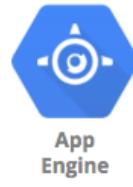
App Engine - Comparison

In 28
Minutes

Feature	Standard	Flexible
Pricing Factors	Instance hours	vCPU, Memory & Persistent Disks
Scaling	Manual, Basic, Automatic	Manual, Automatic
Scaling to zero	Yes	No. Minimum one instance
Instance startup time	Seconds	Minutes
Rapid Scaling	Yes	No
Max. request timeout	1 to 10 minutes	60 minutes
Local disk	Mostly(except for Python, PHP). Can write to /tmp.	Yes. Ephemeral. New Disk on startup.
SSH for debugging	No	Yes

App Engine - Scaling Instances

- **Automatic** - Automatically scale instances based on the load:
 - Recommended for Continuously Running Workloads
 - Auto scale based on:
 - Target CPU Utilization - Configure a CPU usage threshold.
 - Target Throughput Utilization - Configure a throughput threshold
 - Max Concurrent Requests - Configure max concurrent requests an instance can receive
 - Configure Max Instances and Min Instances
- **Basic** - Instances are created as and when requests are received:
 - Recommended for Adhoc Workloads
 - Instances are shutdown if ZERO requests
 - Tries to keep costs low
 - High latency is possible
 - NOT supported by App Engine Flexible Environment
 - Configure Max Instances and Idle Timeout
- **Manual** - Configure specific number of instances to run:
 - Adjust number of instances manually over time



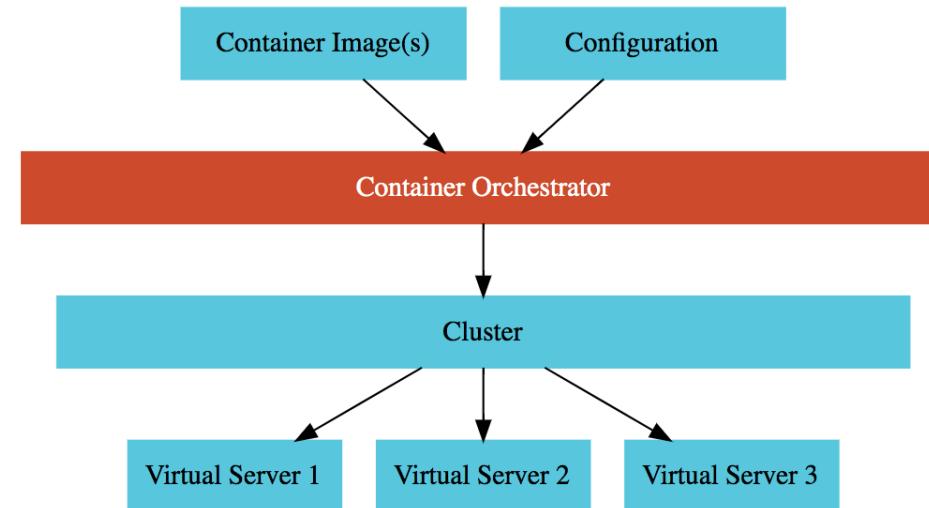
AppEngine Demo

- Deploy an application to cloud using App Engine

Google Kubernetes Engine (GKE)

Kubernetes

- Most popular open source container orchestration solution
- Provides Cluster Management (including upgrades)
 - Each cluster can have different types of virtual machines
- Provides all important container orchestration features:
 - Auto Scaling
 - Service Discovery
 - Load Balancer
 - Self Healing
 - Zero Downtime Deployments



Google Kubernetes Engine (GKE)

- Managed Kubernetes service
- Minimize operations with **auto-repair** (repair failed nodes) and **auto-upgrade** (use latest version of K8S always) features
- Provides **Pod and Cluster Autoscaling**
- Enable **Cloud Logging** and **Cloud Monitoring** with simple configuration
- Uses **Container-Optimized OS**, a hardened OS built by Google
- Provides support for **Persistent disks** and **Local SSD**



Kubernetes Engine

Kubernetes - A Microservice Journey - Getting Started

- **Let's Have Some Fun:** Let's get on a journey with Kubernetes:
 - Let's create a cluster, deploy a microservice and play with it in **13 steps!**
- **1:** Create a Kubernetes cluster with the default node pool
 - *gcloud container clusters create* or use cloud console
- **2:** Login to Cloud Shell
- **3:** Connect to the Kubernetes Cluster
 - *gcloud container clusters get-credentials my-cluster --zone us-central1-a --project solid-course-258105*



Kubernetes Engine

Kubernetes - A Microservice Journey - Deploy Microservice

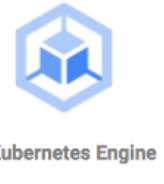
- 4: Deploy Microservice to Kubernetes:
 - Create deployment & service using kubectl commands
 - *kubectl create deployment hello-world-rest-api --image=in28min/hello-world-rest-api:0.0.1.RELEASE*
 - *kubectl expose deployment hello-world-rest-api --type=LoadBalancer --port=8080*
- 5: Increase number of instances of your microservice:
 - *kubectl scale deployment hello-world-rest-api --replicas=2*
- 6: Increase number of nodes in your Kubernetes cluster:
 - *gcloud container clusters resize my-cluster --node-pool my-node-pool --num-nodes 5*
 - You are NOT happy about manually increasing number of instances and nodes!



Kubernetes - A Microservice Journey - Auto Scaling and ..

In 28
Minutes

- 7: Setup auto scaling for your microservice:
 - `kubectl autoscale deployment hello-world-rest-api --max=10 --cpu-percent=70`
 - Also called horizontal pod autoscaling - HPA - `kubectl get hpa`
- 8: Setup auto scaling for your Kubernetes Cluster
 - `gcloud container clusters update cluster-name --enable-autoscaling --min-nodes=1 --max-nodes=10`
- 9: Add some application configuration for your microservice
 - Config Map - `kubectl create configmap todo-web-application-config --from-literal=RDS_DB_NAME=todos`
- 10: Add password configuration for your microservice
 - Kubernetes Secrets - `kubectl create secret generic todo-web-application-secrets-1 --from-literal=RDS_PASSWORD=dummytodos`



Kubernetes Deployment YAML - Deployment

```
apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    app: hello-world-rest-api
    name: hello-world-rest-api
    namespace: default
spec:
  replicas: 3
  selector:
    matchLabels:
      app: hello-world-rest-api
  template:
    metadata:
      labels:
        app: hello-world-rest-api
    spec:
      containers:
        - image: 'in28min/hello-world-rest-api:0.0.3.RELEASE'
          name: hello-world-rest-api
```

Kubernetes Deployment YAML - Service

```
apiVersion: v1
kind: Service
metadata:
  labels:
    app: hello-world-rest-api
    name: hello-world-rest-api
    namespace: default
spec:
  ports:
  - port: 8080
  protocol: TCP
  targetPort: 8080
  selector:
    app: hello-world-rest-api
    sessionAffinity: None
  type: LoadBalancer
```

Kubernetes - A Microservice Journey - The End!

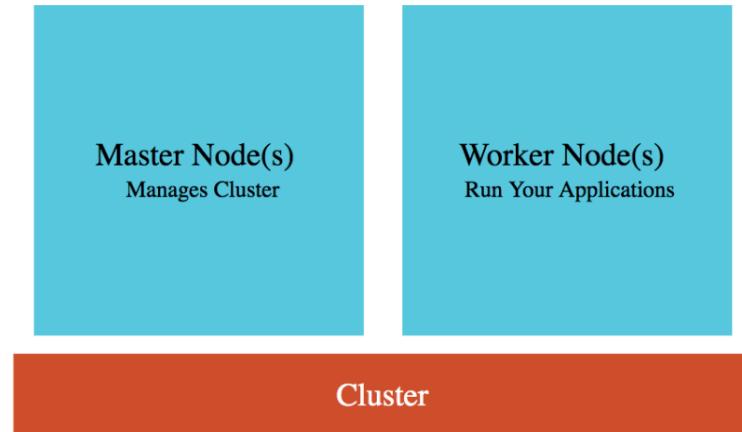
- **11:** Deploy a new microservice which needs nodes with a GPU attached
 - Attach a new node pool with GPU instances to your cluster
 - `gcloud container node-pools create POOL_NAME --cluster CLUSTER_NAME`
 - `gcloud container node-pools list --cluster CLUSTER_NAME`
 - Deploy the new microservice to the new pool by setting up `nodeSelector` in the `deployment.yaml`
 - `nodeSelector: cloud.google.com/gke-nodepool: POOL_NAME`
- **12:** Delete the Microservices
 - Delete service - `kubectl delete service`
 - Delete deployment - `kubectl delete deployment`
- **13:** Delete the Cluster
 - `gcloud container clusters delete`



Kubernetes Engine

Google Kubernetes Engine (GKE) Cluster

- **Cluster** : Group of Compute Engine instances:
 - **Master Node(s)** - Manages the cluster
 - **Worker Node(s)** - Run your workloads (pods)
- **Master Node (Control plane)** components:
 - **API Server** - Handles all communication for a K8S cluster (from nodes and outside)
 - **Scheduler** - Decides placement of pods
 - **Control Manager** - Manages deployments & replicaset
 - **etcd** - Distributed database storing the cluster state
- **Worker Node** components:
 - Runs your pods
 - **Kubelet** - Manages communication with master node(s)

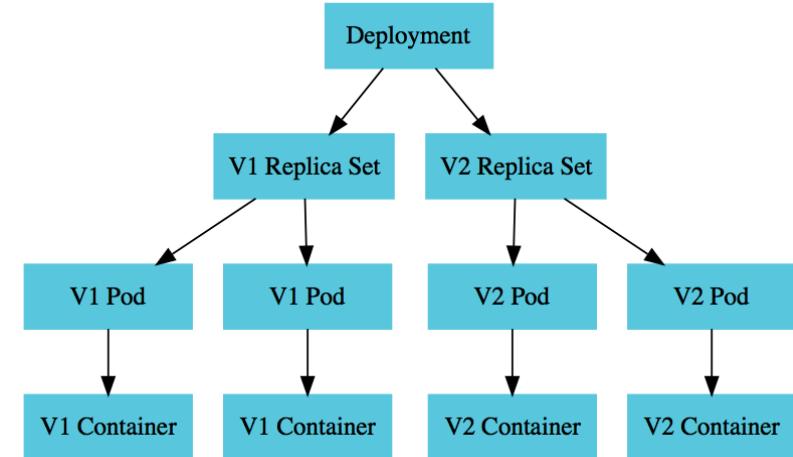


GKE Cluster Types

Type	Description	
Zonal Cluster	Single Zone - Single Control plane. Nodes running in the same zone.	 Kubernetes Engine
	Multi-zonal - Single Control plane but nodes running in multiple zones	
Regional cluster	Replicas of the control plane runs in multiple zones of a given region. Nodes also run in same zones where control plane runs.	
Private cluster	VPC-native cluster. Nodes only have internal IP addresses.	
Alpha cluster	Clusters with alpha APIs - early feature API's. Used to test new K8S features.	

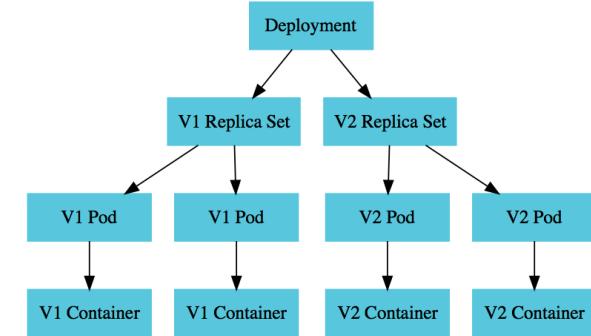
Kubernetes - Pods

- Smallest deployable unit in Kubernetes
- A Pod contains **one or more containers**
- Each Pod is assigned an ephemeral **IP address**
- All containers in a pod share:
 - Network
 - Storage
 - IP Address
 - Ports and
 - Volumes (Shared persistent disks)
- POD statuses : Running /Pending /Succeeded /Failed /Unknown



Kubernetes - Deployment vs Replica Set

- A **deployment** is created for each microservice:
 - `kubectl create deployment m1 --image=m1:v1`
 - Deployment represents a microservice (with all its releases)
 - Deployment manages new releases ensuring zero downtime
- **Replica set** ensures that a specific number of pods are running for a specific microservice version
 - `kubectl scale deployment m2 --replicas=2`
 - Even if one of the pods is killed, replica set will launch a new one
- Deploy V2 of microservice - Creates a new replica set
 - `kubectl set image deployment m1 m1=m1:v2`
 - V2 Replica Set is created
 - Deployment updates V1 Replica Set and V2 Replica Set based on the release strategies



Kubernetes - Service

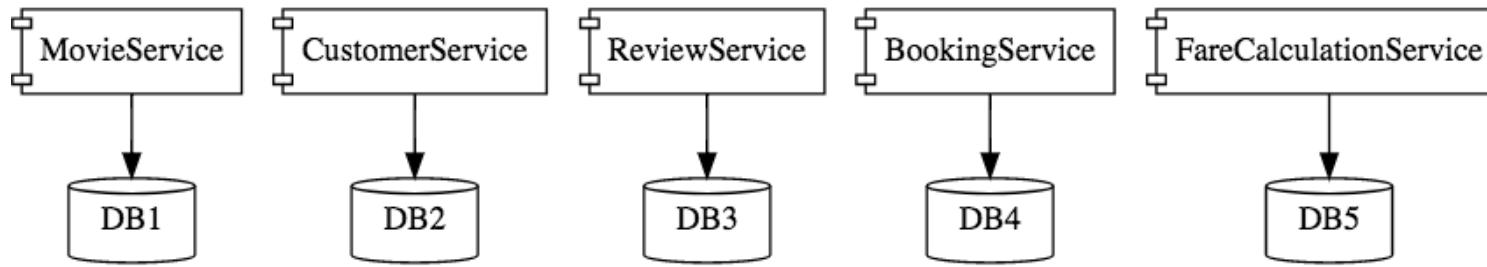
- Each Pod has its own IP address:
 - How do you ensure that external users are not impacted when:
 - A pod fails and is replaced by replica set
 - A new release happens and all existing pods of old release are replaced by ones of new release
- Create Service
 - *kubectl expose deployment name --type=LoadBalancer --port=80*
 - Expose PODs to outside world using a stable IP Address
 - Ensures that the external world does not get impacted as pods go down and come up
- Three Types:
 - **ClusterIP:** Exposes Service on a cluster-internal IP
 - Use case: You want your microservice only to be available inside the cluster (Intra cluster communication)
 - **LoadBalancer:** Exposes Service externally using a cloud provider's load balancer
 - Use case: You want to create individual Load Balancer's for each microservice
 - **NodePort:** Exposes Service on each Node's IP at a static port (the NodePort)
 - Use case: You DO not want to create an external Load Balancer for each microservice (You can create one Ingress

Ingress

```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: gateway-ingress
spec:
  rules:
  - http:
      paths:
      - path: /currency-exchange/*
        backend:
          serviceName: currency-exchange
          servicePort: 8000
      - path: /currency-conversion/*
        backend:
          serviceName: currency-conversion
          servicePort: 8100
```

- Recommended Approach for providing external access to services in a cluster
 - Provides load balancing and SSL termination
 - Control traffic by defining rules on the Ingress resource
 - Recommendation: NodePort Service for each microservice. Expose using a Ingress.

Container Registry - Image Repository



- You've created docker images for your microservices:
 - Where do you store them?
- **Container Registry** - fully-managed container registry provided by GCP
- (Alternative) Docker Hub
- Can be integrated to CI/CD tools to publish images to registry
- You can secure your container images. Analyze for vulnerabilities and enforce deployment policies.
- Naming: HostName/ProjectID/Image:Tag - gcr.io/projectname/helloworld:1

Creating Docker Images - Dockerfile

```
FROM node:8.16.1-alpine
WORKDIR /app
COPY . /app
RUN npm install
EXPOSE 5000
CMD node index.js
```

- Dockerfile contains instruction to create Docker images
 - **FROM** - Sets a base image
 - **WORKDIR** - sets the working directory
 - **RUN** - execute a command
 - **EXPOSE** - Informs Docker about the port that the container listens on at runtime
 - **COPY** - Copies new files or directories into image
 - **CMD** - Default command for an executing container

Understanding Best Practices - Docker Image

```
FROM node:8.16.1-alpine
WORKDIR /app
COPY package.json /app
RUN npm install
EXPOSE 5000
COPY . /app
CMD node index.js
```



- Use light weight images (Prefer Alpine Linux to Ubuntu)
- Do not copy anything that's unnecessary (node_modules for example)
- Move things that change less often to the top to enable LAYER REUSE
 - Code changes often (index.js) BUT Dependencies change less often (package.json contains libraries that the app needs)
 - So, first build dependencies (npm install - which takes time) and then copy rest of code (COPY . /app)

Google Kubernetes Engine - Scenarios - 1

In 28
Minutes

Scenario	Solution
You want to keep your costs low and optimize your GKE implementation	Consider Preemptible VMs, Appropriate region, Committed-use discounts. E2 machine types are cheaper than N1. Choose right environment to fit your workload type (Use multiple node pools if needed).
You want an efficient, completely auto scaling GKE solution	Configure Horizontal Pod Autoscaler for deployments and Cluster Autoscaler for node pools
You want to execute untrusted third-party code in Kubernetes Cluster	Create a new node pool with GKE Sandbox. Deploy untrusted code to Sandbox node pool.

Google Kubernetes Engine - Scenarios - 2

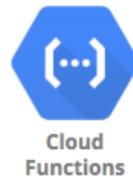
In 28
Minutes

Scenario	Solution
You want enable ONLY internal communication between your microservice deployments in a Kubernetes Cluster	Create Service of type ClusterIP
My pod stays pending	Most probably Pod cannot be scheduled onto a node(insufficient resources)
My pod stays waiting	Most probably failure to pull the image

Google Cloud Functions

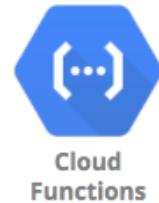
Cloud Functions

- Imagine you want to **execute some code when an event happens?**
 - A file is uploaded in Cloud Storage (OR) An error log is written to Cloud Logging (OR)
 - A message arrives to Cloud Pub/Sub (OR) A http/https invocation is received
- Enter **Cloud Functions**
 - **Run code in response to events**
 - Write your business logic in Node.js, Python, Go, Java, .NET, and Ruby
 - **Don't worry** about servers or scaling or availability (only worry about your code)
 - **Pay only for what you use**
 - Number of invocations
 - Compute time of the invocations
 - Memory and CPU provisioned
 - **Time Bound** - Default 1 min and MAX 60 minutes (3600 seconds)
 - **2 product versions**
 - Cloud Functions (1st gen): First version
 - Cloud Functions (2nd gen): New version built on top of Cloud Run and Eventarc



Cloud Functions - Concepts

- **Event** : Upload object to cloud storage
- **Trigger**: Respond to event with a Function call
 - **Trigger** - Which function to trigger when an event happens?
 - **Functions** - Take event data and perform action?
- Events are **triggered from**
 - Cloud Storage
 - Cloud Pub/Sub
 - HTTP POST/GET/DELETE/PUT/OPTIONS
 - Firebase
 - Cloud Firestore
 - Stack driver logging



Example Cloud Function - HTTP - Node.js

```
const escapeHtml = require('escape-html');

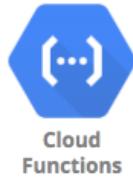
exports.helloHttp = (req, res) => {
  res.send(`Hello ${escapeHtml(req.query.name || req.body.name || 'World')}`);
};
```

Example Cloud Function - Pub/Sub - Node.js

```
/**  
 * Background Cloud Function to be triggered by Pub/Sub.  
 * This function is exported by index.js, and executed when  
 * the trigger topic receives a message.  
 *  
 * @param {object} message The Pub/Sub message.  
 * @param {object} context The event metadata.  
 */  
exports.helloPubSub = (message, context) => {  
  const name = message.data  
  ? Buffer.from(message.data, 'base64').toString()  
  : 'World';  
  
  console.log(`Hello, ${name}!`);  
};
```

Cloud Functions - Remember

- No Server Management: You dont need to worry about scaling or availability of your function
- Cloud Functions automatically spin up and back down in response to events
 - They scale horizontally!
- Cloud Functions are recommended for responding to events:
 - Cloud Functions are NOT ideal for long running processes
 - Time Bound - Default 1 min and MAX 60 minutes(3600 seconds)



Cloud Run & Cloud Run for Anthos

- **Cloud Run - "Container to Production in Seconds"**
 - Built on top of an open standard - Knative
 - **Fully managed** serverless platform for containerized applications
 - ZERO infrastructure management
 - Pay-per-use (For used CPU, Memory, Requests and Networking)
- Fully integrated **end-to-end developer experience**:
 - **No limitations** in languages, binaries and dependencies
 - Easily portable because of **container** based architecture
 - Cloud Code, Cloud Build, Cloud Monitoring & Cloud Logging Integrations
- **Anthos** - Run Kubernetes clusters anywhere
 - Cloud, Multi Cloud and On-Premise
- **Cloud Run for Anthos**: Deploy your workloads to Anthos clusters running on-premises or on Google Cloud
 - Leverage your existing Kubernetes investment to quickly run serverless workloads

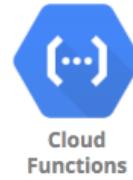


Cloud Run

Cloud Functions - Second Generation - What's New?

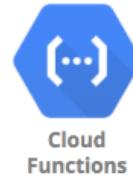
In 28
Minutes

- **2 Product Versions:**
 - Cloud Functions (1st gen): First version
 - Cloud Functions (2nd gen): New version built on top of Cloud Run and Eventarc
- **Recommended:** Use Cloud Functions (2nd gen)
- **Key Enhancements in 2nd gen:**
 - **Longer Request timeout:** Up to 60 minutes for HTTP-triggered functions
 - **Larger instance sizes:** Up to 16GiB RAM with 4 vCPU (v1: Up to 8GB RAM with 2 vCPU)
 - **Concurrency:** Up to 1000 concurrent requests per function instance (v1: 1 concurrent request per function instance)
 - **Multiple Function Revisions and Traffic splitting supported** (v1: NOT supported)
 - **Support for 90+ event types** - enabled by Eventarc (v1: Only 7)
- **DEMO!**



Cloud Functions - Scaling and Concurrency

- Typical serverless functions architecture:
 - Autoscaling - As new invocations come in, new function instances are created
 - One function instance handles ONLY ONE request AT A TIME
 - 3 concurrent function invocations => 3 function instances
 - If a 4th function invocation occurs while existing invocations are in progress, a new function instance will be created
 - HOWEVER, a function instance that completed execution may be reused for future requests
 - (Typical Problem) Cold Start:
 - New function instance initialization from scratch can take time
 - (Solution) Configure Min number of instances (increases cost)
- 1st Gen uses the typical serverless functions architecture
- 2nd Gen adds a very important new feature:
 - One function instance can handle multiple requests AT THE SAME TIME
 - Concurrency: How many concurrent invocations can one function instance handle? (Max 1000)
 - (IMPORTANT) Your function code should be safe to execute concurrently



Cloud Functions - Deployment using gcloud

- **gcloud functions deploy [NAME]**

- **--docker-registry** (registry to store the function's Docker images)
 - Default - container - registry
 - Alternative - artifact - registry
- **--docker-repository** (repository to store the function's Docker images)
 - Example: (projects/\${PROJECT}/locations/\${LOCATION}/repositories/\${REPOSITORY})
- **--gen2** (Use 2nd gen. If this option is not present, 1st gen will be used)
- **--runtime** (nodejs, python, java,...)
 - Reference - <https://cloud.google.com/functions/docs/runtime-support>
- **--service-account** (Service account to use)
 - 1 GEN - default - App Engine default service account - PROJECT_ID@appspot.gserviceaccount.com
 - 2 GEN - Default compute service account - PROJECT_N0-compute@developer.gserviceaccount.com
- **--timeout** (function execution timeout)
- **--max-instances** (function execution exceeding max-instances times out)
- **--min-instances** (avoid cold starts at higher cost)

Cloud Functions - Deployment using gcloud - 2

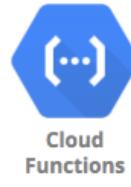
```
//Deploy Pubsub Triggered gen2 function from Cloud Storage Bucket
gcloud functions deploy my-pubsub-function \
    --gen2 \
    --region=europe-west1 \
    --runtime=nodejs16 \
    --source=gs://my-source-bucket/source.zip \
    --trigger-topic=my-pubsub-topic
```

- **gcloud functions deploy [NAME]**

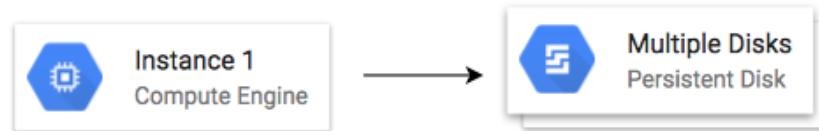
- **--source**
 - Zip file from Google Cloud Storage (gs://my-source-bucket/my_function_source.zip) (OR)
 - Source Repo ([https://URL/projects/\\${PROJECT}/repos/\\${REPO}](https://URL/projects/${PROJECT}/repos/${REPO})) (OR)
 - Local file system
- **--trigger-bucket** (OR) **--trigger-http** (OR) **--trigger-topic** (OR) **--trigger-event-filters** (ONLY in gen2 - Eventarc matching criteria for the trigger)
- **--serve-all-traffic-latest-revision** (ONLY in gen2)

Cloud Functions - Best Practices

- To avoid cold starts, set **min no of instances** (increases cost)
 - Minimize dependencies (loading dependencies increases initialization time)
- Configure **max no of instances** (protect from abnormally high request levels)
- Use Cloud Endpoints (or Apigee or API gateway) for versioning
- Use Cloud Run (& Cloud Functions gen 2) **revisions** for safer releases:
 - Configure which revisions should receive traffic and how much
 - You can rollback to a previous revision, if needed
- Use Secret Manager to securely store secrets (ex: API keys)
- Use **Individual Service Accounts** for each function
 - Grant `roles/cloudfunctions.invoker` role to invoke a cloud function
- Manage dependencies using your language specific tool (npm, pip,..)



Encryption



- **Data at rest:** Stored on a device or a backup
 - Examples : data on a hard disk, in a database, backups and archives
- **Data in motion:** Being transferred across a network
 - Also called Data in transit
 - Examples :
 - Data copied from on-premise to cloud storage
 - An application talking to a database
 - Two Types:
 - In and out of cloud (from internet)
 - Within cloud
- **Data in use:** Active data processed in a non-persistent state
 - Example: Data in your RAM

Encryption

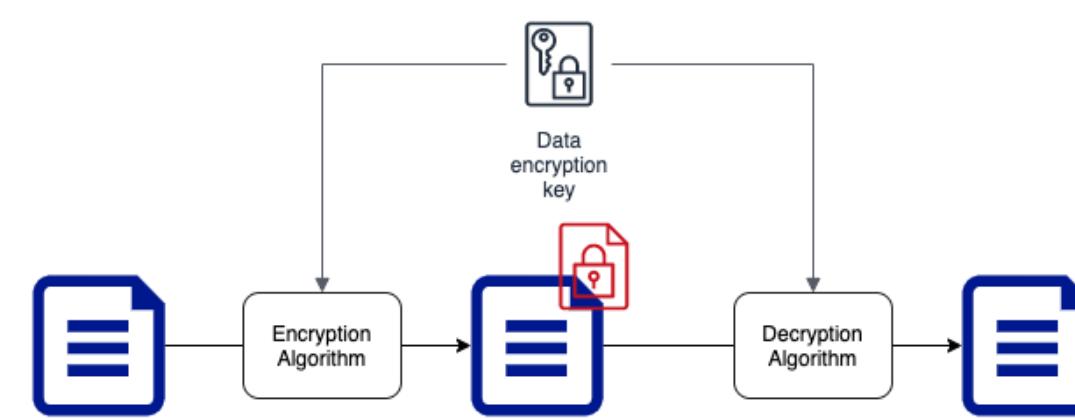
In 28
Minutes



- If you store data as is, what would happen if an **unauthorized entity gets access to it?**
 - Imagine losing an unencrypted hard disk
- **First law of security :** Defense in Depth
- Typically, enterprises encrypt all data
 - Data on your hard disks
 - Data in your databases
 - Data on your file servers
- Is it sufficient if you encrypt data at rest?
 - No. Encrypt data in transit - between application to database as well.

Symmetric Key Encryption

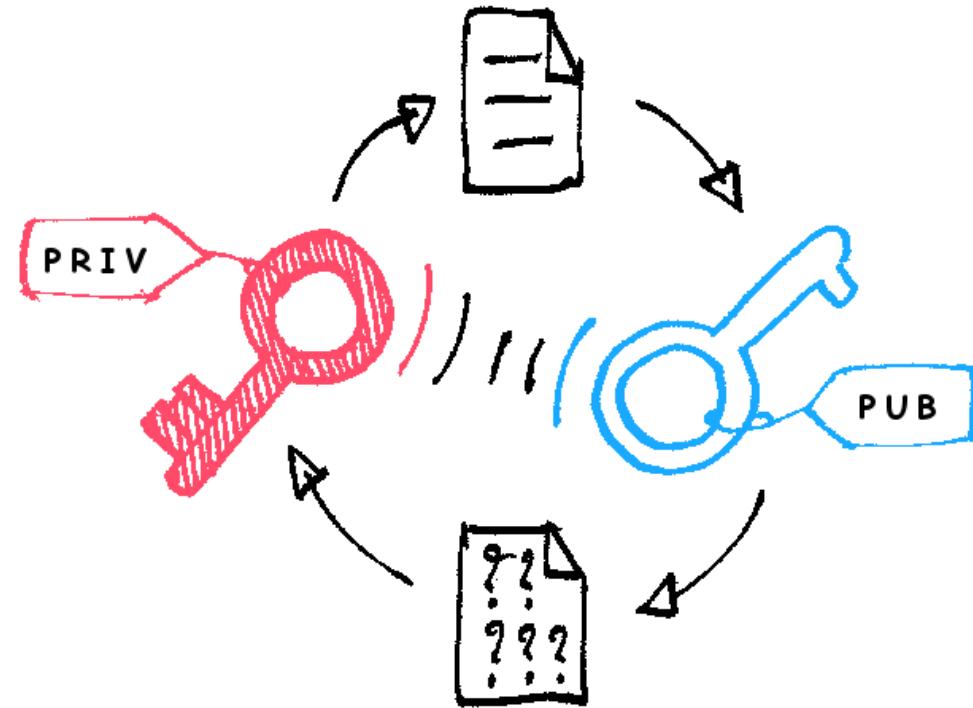
In 28
Minutes



- Symmetric encryption algorithms use the **same key for encryption and decryption**
- Key Factor 1: Choose the **right encryption algorithm**
- Key Factor 2: How do we **secure the encryption key?**
- Key Factor 3: How do we **share the encryption key?**

Asymmetric Key Encryption

- Two Keys : Public Key and Private Key
- Also called **Public Key Cryptography**
- Encrypt data with Public Key and decrypt with Private Key
- Share Public Key with everybody and keep the Private Key with you(YEAH, ITS PRIVATE!)
- No crazy questions:
 - Will somebody not figure out private key using the public key?
- How do you create Asymmetric Keys?



[https://commons.wikimedia.org/wiki/File:Asymmetric_encryption_\(colored\).png](https://commons.wikimedia.org/wiki/File:Asymmetric_encryption_(colored).png)

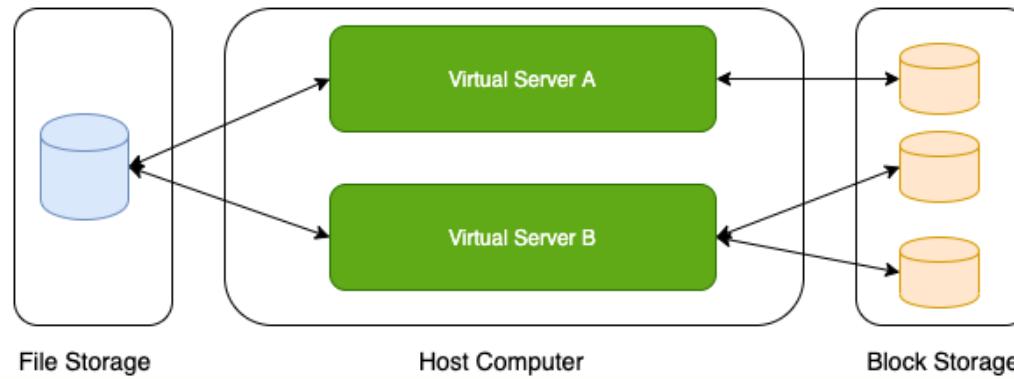
Cloud KMS

- Create and manage **cryptographic keys** (symmetric and asymmetric)
- **Control their use** in your applications and GCP Services
- Provides an API to encrypt, decrypt, or sign data
- Use existing cryptographic keys created on premises
- **Integrates with almost all GCP services** that need data encryption:
 - Google-managed key: No configuration required
 - Customer-managed key: Use key from KMS
 - Customer-supplied key: Provide your own key



Storage

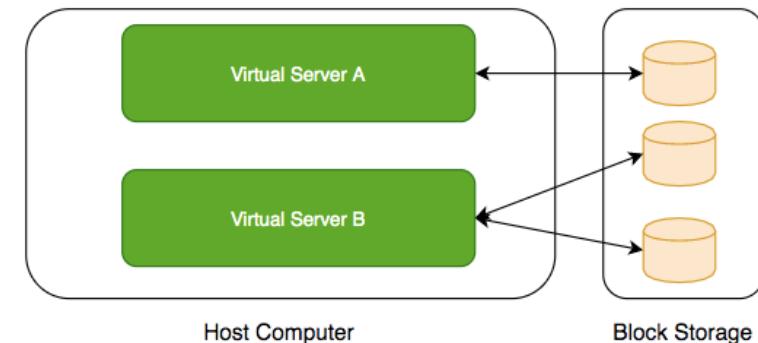
Storage Types - Block Storage and File Storage



- What is the type of storage of your hard disk?
 - **Block Storage**
- You've created a file share to share a set of files with your colleagues in a enterprise. What type of storage are you using?
 - **File Storage**

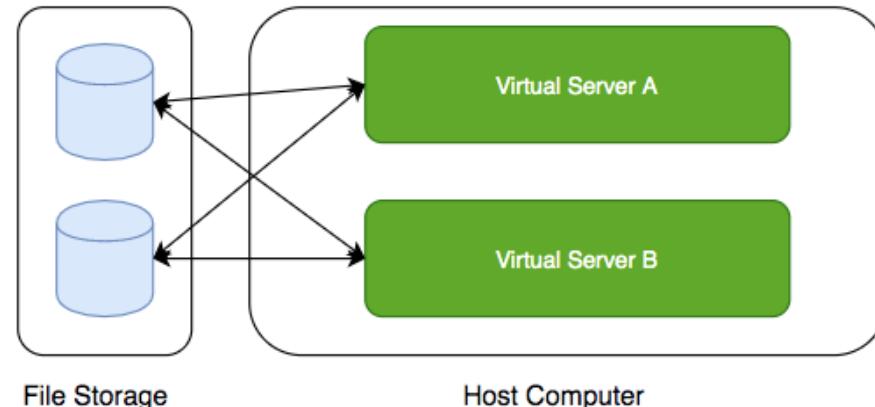
Block Storage

- Use case: Harddisks attached to your computers
- Typically, ONE Block Storage device can be connected to ONE virtual server
 - (EXCEPTIONS) You can attach read only block devices with multiple virtual servers and certain cloud providers are exploring multi-writer disks as well!
- HOWEVER, you can connect multiple different block storage devices to one virtual server
- Used as:
 - Direct-attached storage (DAS) - Similar to a hard disk
 - Storage Area Network (SAN) - High-speed network connecting a pool of storage devices



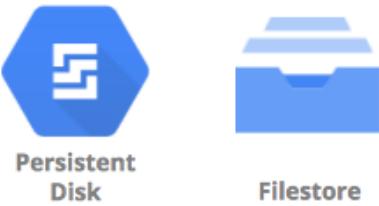
File Storage

- Media workflows need huge shared storage for supporting processes like video editing
- Enterprise users need a quick way to share files in a secure and organized way
- These file shares are shared by several virtual servers



GCP - Block Storage and File Storage

In 28
Minutes



- **Block Storage:**
 - **Persistent Disks:** Network Block Storage
 - Zonal: Data replicated in one zone
 - Regional: Data replicated in multiple zone
 - **Local SSDs:** Local Block Storage
- **File Storage:**
 - **Filestore:** High performance file storage

- Two popular types of block storage can be attached to VM instances:
 - Local SSDs
 - Persistent Disks
- **Local SSDs** are physically attached to the host of the VM instance
 - Temporary data
 - Lifecycle tied to VM instance
- **Persistent Disks** are network storage
 - More durable
 - Lifecycle NOT tied to VM instance

Local SSDs

In 28
Minutes

- **Physically attached** to the host of VM instance:
 - Provide very high (IOPS) and very low latency
 - (**BUT**) **Ephemeral storage** - Temporary data (Data persists only until instance is running)
 - Enable live migration for data to survive maintenance events
 - Data automatically encrypted
 - HOWEVER, you CANNOT configure encryption keys!
 - Lifecycle tied to VM instance
 - ONLY some machine types support Local SSDs
 - Supports SCSI and NVMe interfaces
- Remember:
 - Choose NVMe-enabled and multi-queue SCSI images for best performance
 - Larger Local SSDs (more storage), More vCPUs (attached to VM) => Even Better Performance

Local SSDs - Advantages and Disadvantages

- **Advantages**

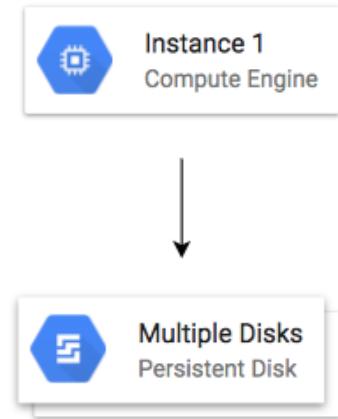
- Very Fast I/O (~ 10-100X compared to PDs)
 - Higher throughput and lower latency
- Ideal for use cases needing high IOPs while storing **temporary information**
 - Examples: Caches, temporary data, scratch files etc

- **Disadvantages**

- **Ephemeral storage**
 - Lower durability, lower availability, lower flexibility compared to PDs
- You **CANNOT detach and attach** it to another VM instance

Persistent Disks (PD)

- Network block storage attached to your VM instance
- Provisioned capacity
- Very Flexible:
 - Increase size when you need it - when attached to VM instance
 - Performance scales with size
 - For higher performance, resize or add more PDs
- Independent lifecycle from VM instance
 - Attach/Detach from one VM instance to another
- Options: Regional and Zonal
 - Zonal PDs replicated in single zone. Regional PDs replicated in 2 zones in same Region.
 - Typically Regional PDs are 2X the cost of Zonal PDs
- Use case : Run your custom database



Persistent Disks vs Local SSDs

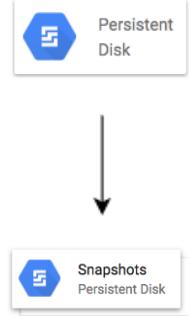
Feature	Persistent Disks	Local SSDs
Attachment to VM instance	As a network drive	Physically attached
Lifecycle	Separate from VM instance	Tied with VM instance
I/O Speed	Lower (network latency)	10-100X of PDs
Snapshots	Supported	Not Supported
Use case	Permanent storage	Ephemeral storage

Persistent Disks - Standard vs Balanced vs SSD

Feature	Standard	Balanced	SSD
Underlying Storage	Hard Disk Drive	Solid State Drive	Solid State Drive
Referred to as	pd-standard	pd-balanced	pd-ssd
Performance - Sequential IOPS (Big Data/Batch)	Good	Good	Very Good
Performance - Random IOPS (Transactional Apps)	Bad	Good	Very Good
Cost	Cheapest	In Between	Expensive
Use cases	Big Data (cost efficient)	Balance between cost and performance	High Performance

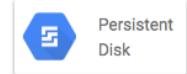
Persistent Disks - Snapshots

- Take **point-in-time snapshots** of your Persistent Disks
- You can also schedule snapshots (configure a schedule):
 - You can also auto-delete snapshots after X days
- Snapshots can be Multi-regional and Regional
- You can share snapshots across projects
- You can create new disks and instances from snapshots
- Snapshots are **incremental**:
 - Deleting a snapshot **only deletes data which is NOT needed by other snapshots**
- Keep similar data together on a Persistent Disk:
 - Separate your operating system, volatile data and permanent data
 - Attach multiple disks if needed
 - This helps to better organize your snapshots and images



Persistent Disks - Snapshots - Recommendations

- Avoid taking snapshots more often than once an hour
- Disk volume is available for use **but Snapshots reduce performance**
 - (RECOMMENDED) Schedule snapshots during off-peak hours
- Creating snapshots from disk is faster than creating from images:
 - But creating disks from image is faster than creating from snapshots
 - (RECOMMENDED) If you are repeatedly creating disks from a snapshot:
 - Create an image from snapshot and use the image to create disks
- Snapshots are **incremental**:
 - BUT you don't lose data by deleting older snapshots
 - Deleting a snapshot **only deletes data which is NOT needed** by other snapshots
 - (RECOMMENDED) Do not hesitate to delete unnecessary snapshots



Persistent
Disk



Snapshots
Persistent Disk

Mounting a Data Persistent Disk on a GCE VM



- Steps to attach a newly created Persistent Disk with an already running VM:
 - A: Attach Disk to running or stopped VM
 - `gcloud compute instances attach-disk INSTANCE_NAME --disk DISK_NAME`
 - B: Format the disk
 - 1: List disks attached to your VM
 - `sudo lsblk`
 - 2: Format to file format of your choice (example: ext4 file system)
 - `sudo mkfs.ext4 -m 0 -E lazy_itable_init=0,lazy_journal_init=0,discard /dev/sdb`
 - C: Mount the disk
 - 1: Create the directory to mount to
 - `sudo mkdir -p /mnt/disks/MY_DIR`
 - 2: Mount the disk
 - `sudo mount -o discard,defaults /dev/sdb /mnt/disks/MY_DIR`
 - 3: Provide permissions

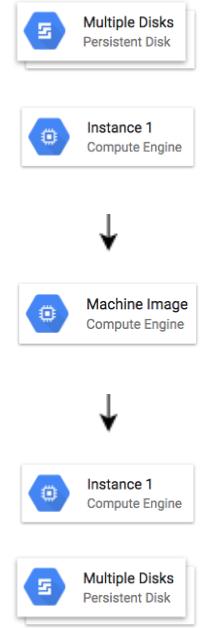
Resizing Data Persistent Disks



- Step I: Resize the Disk
 - `gcloud compute disks resize DISK_NAME --size DISK_SIZE`
- Step II: Take a snapshot (Just for backup in case things go wrong!)
- Step III: Resize the file system and partitions
 - ext4: `sudo resize2fs /dev/sdb`
 - xfs: `sudo xfs_growfs /dev/sdb`

Playing with Machine Images

- (Remember) **Machine Image** is different from Image
- **Multiple disks can be attached** with a VM:
 - One Boot Disk (Your OS runs from Boot Disk)
 - Multiple Data Disks
- An image is created from the boot Persistent Disk
- HOWEVER, a Machine Image is created from a VM instance:
 - Machine Image **contains everything you need** to create a VM instance:
 - Configuration
 - Metadata
 - Permissions
 - Data from one or more disks
- **Recommended for disk backups, instance cloning and replication**



Let's Compare

Scenarios	Machine image	Persistent disk snapshot	Custom image	Instance template
Single disk backup	Yes	Yes	Yes	No
Multiple disk backup	Yes	No	No	No
Differential backup	Yes	Yes	No	No
Instance cloning and replication	Yes	No	Yes	Yes
VM instance configuration	Yes	No	No	Yes

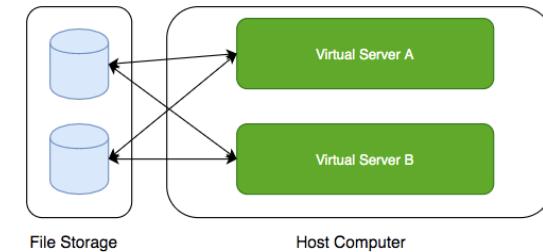
<https://cloud.google.com/compute/docs/machine-images>

Storage - Scenarios - Persistent Disks

Scenario	Solution
You want to improve performance of Persistent Disks (PD)	Increase size of PD or Add more PDs. Increase vCPUs in your VM.
You want to increase durability of Persistent Disks (PD)	Go for Regional PDs (2X cost but replicated in 2 zones)
You want to take hourly backup of Persistent Disks (PD) for disaster recovery	Schedule hourly snapshots!
You want to delete old snapshots created by scheduled snapshots	Configure it as part of your snapshot scheduling!

Cloud Filestore

- **Shared cloud file storage:**
 - Supports NFSv3 protocol
 - Provisioned Capacity
- **Suitable for high performance workloads:**
 - Up to 320 TB with throughput of 16 GB/s and 480K IOPS
- Supports HDD (general purpose) and SSD (performance-critical workloads)
- **Use cases :** file share, media workflows and content management



Review - Global, Regional and Zonal Resources

- **Global**
 - Images
 - Snapshots
 - Instance templates (Unless you use zonal resources in your templates)
- **Regional**
 - Regional managed instance groups
 - Regional persistent disks
- **Zonal**
 - Zonal managed instance groups
 - Instances
 - Persistent disks
 - You can attach a disk only to instances in the same zone as the disk

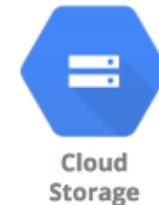
Storage - Scenarios

Scenario	Solution
You want Very High IOPS but your data can be lost without a problem	Local SSDs
You want to create a high performance file sharing system in GCP which can be attached with multiple VMs	Filestore
You want to backup your VM configuration along with all its attached Persistent Disks	Create a Machine Image
You want to make it easy to launch VMs with hardened OS and customized software	Create a Custom Image

Object Storage - Cloud Storage

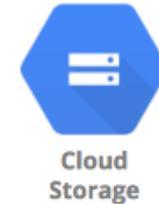
Cloud Storage

- Most popular, very flexible & inexpensive storage service
 - Serverless: Autoscaling and infinite scale
- Store large objects using a **key-value** approach:
 - Treats entire object as a unit (Partial updates not allowed)
 - Recommended when you operate on entire object most of the time
 - Access Control at Object level
 - Also called **Object Storage**
- Provides REST API to access and modify objects
 - Also provides CLI (gsutil) & Client Libraries (C++, C#, Java, Node.js, PHP, Python & Ruby)
- **Store all file types** - text, binary, backup & archives:
 - Media files and archives, Application packages and logs
 - Backups of your databases or storage devices
 - Staging data during on-premise to cloud database migration



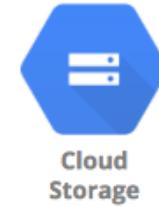
Cloud Storage - Objects and Buckets

- Objects are stored in buckets
 - Bucket names are **globally unique**
 - Bucket names are used as part of object URLs => Can contain ONLY lower case letters, numbers, hyphens, underscores and periods.
 - 3-63 characters max. Can't start with **goog prefix** or should not contain **google (even misspelled)**
 - Unlimited objects in a bucket
 - Each bucket is associated with a project
- Each object is identified by a **unique key**
 - Key is **unique** in a bucket
- Max object size is **5 TB**
 - BUT you can store unlimited number of such objects



Cloud Storage - Storage Classes - Introduction

- Different kinds of data can be stored in Cloud Storage
 - Media files and archives
 - Application packages and logs
 - Backups of your databases or storage devices
 - Long term archives
- Huge variations in access patterns
- Can I pay a cheaper price for objects I access less frequently?
- Storage classes help to optimize your costs based on your access needs
 - Designed for durability of 99.999999999%(11 9's)

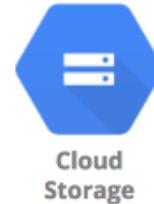


Cloud Storage - Storage Classes - Comparison

Storage Class	Name	Minimum Storage duration	Typical Monthly availability	Use case
Standard	STANDARD	None	> 99.99% in multi region and dual region, 99.99% in regions	Frequently used data/Short period of time
Nearline storage	NEARLINE	30 days	99.95% in multi region and dual region, 99.9% in regions	Read or modify once a month on average
Coldline storage	COLDLINE	90 days	99.95% in multi region and dual region, 99.9% in regions	Read or modify at most once a quarter
Archive storage	ARCHIVE	365 days	99.95% in multi region and dual region, 99.9% in regions	Less than once a year

Features across Storage Classes

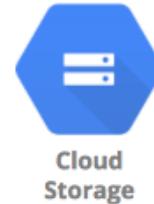
- High durability (99.99999999% annual durability)
- Low latency (first byte typically in tens of milliseconds)
- **Unlimited** storage
 - Autoscaling (No configuration needed)
 - NO minimum object size
- Same APIs across storage classes
- **Committed SLA** is 99.95% for multi region and 99.9% for single region for Standard, Nearline and Coldline storage classes
 - No committed SLA for Archive storage



Cloud
Storage

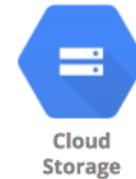
Object Versioning

- Prevents **accidental deletion** & provides history
 - Enabled at bucket level
 - Can be turned on/off at any time
 - **Live version** is the latest version
 - If you delete live object, it becomes noncurrent object version
 - If you delete noncurrent object version, it is deleted
 - Older versions are uniquely identified by (object key + a generation number)
 - Reduce costs by deleting older (noncurrent) versions!



Object Lifecycle Management

- Files are frequently accessed when they are created
 - Generally usage reduces with time
 - How do you save costs by **moving files automatically between storage classes?**
 - Solution: Object Lifecycle Management
- Identify objects using conditions based on:
 - Age, CreatedBefore, IsLive, MatchesStorageClass, NumberOfNewerVersions etc
 - Set multiple conditions: all conditions must be satisfied for action to happen
- Two kinds of actions:
 - **SetStorageClass** actions (change from one storage class to another)
 - **Deletion** actions (delete objects)
- Allowed Transitions:
 - (Standard or Multi-Regional or Regional) to (Nearline or Coldline or Archive)
 - Nearline to (Coldline or Archive)
 - Coldline to Archive

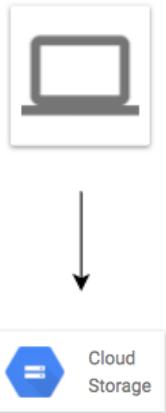


Object Lifecycle Management - Example Rule

```
{  
  "lifecycle": {  
    "rule": [  
      {  
        "action": {"type": "Delete"},  
        "condition": {  
          "age": 30,  
          "isLive": true  
        }  
      },  
      {  
        "action": {  
          "type": "SetStorageClass",  
          "storageClass": "NEARLINE"  
        },  
        "condition": {  
          "age": 365,  
          "matchesStorageClass": ["STANDARD"]  
        }  
      }  
    ]  
  }  
}
```

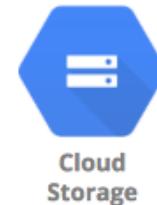
Cloud Storage - Encryption

- (Default) Cloud Storage encrypts data on the server side!
- **Server-side** encryption: Performed by GCS after it receives data
 - **Google-managed** - Default (No configuration needed)
 - **Customer-managed** - Keys managed by customer in **Cloud KMS**:
 - GCS Service Account should have access to customer-managed keys in KMS to be able to encrypt and decrypt files
 - **Customer-supplied** - Customer supplies the keys with every GCS operation
 - Cloud Storage does NOT store the key
 - Customer is responsible for storing and using it when making API calls
 - Use API headers when making API calls
 - x-goog-encryption-algorithm, x-goog-encryption-key (Base 64 encryption key), x-goog-encryption-key-sha256 (encryption key hash)
 - OR when using gsutil: In boto configuration file, configure encryption_key under GSUtil section
- (OPTIONAL) **Client-side** encryption - Encryption performed by customer before upload
 - GCP does NOT know about the keys used
 - GCP is NOT involved in encryption or decryption



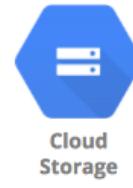
Understanding Cloud Storage Metadata

- Each object in Cloud Storage can have **Metadata** associated with it
 - Key Value Pairs ex: **storageClass:STANDARD**
 - Storage class of an object is represented by metadata
 - **Fixed-key metadata:** Fixed key - Changing value
 - Cache-Control: public, max-age=3600 (Is caching allowed? If so, for how long?)
 - Content-Disposition: attachment; filename="myfile.pdf" (Should content be displayed inline in the browser or should it be an attachment, which can be downloaded)
 - Content-Type: application/pdf (What kind of content does the object have?)
 - etc..
 - **Custom metadata:** You can define your own keys and values
 - **Non-editable metadata:** You cannot edit these directly
 - Storage class of the object, customer-managed encryption keys etc



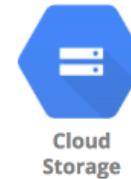
Cloud Storage Bucket Lock - Meet Compliance Needs

- How do you ensure that you **comply with regulatory and compliance requirements** around immutable storage in a Cloud Storage bucket?
- Configure **data retention policy** with retention period:
 - How long should objects in the bucket be retained for?
 - "Objects in the bucket can only be deleted or replaced once their age is greater than the retention period"
 - You can set it **while creating a bucket or at a later point in time**
 - Applies automatically to existing objects in the bucket (as well as new objects added in)
 - Once a retention policy is locked:
 - You **CANNOT remove retention policy or reduce** retention period (You can increase retention period)
 - You **CANNOT delete the bucket** unless all objects in bucket have age greater than retention period
 - Retention policies and Object Versioning are mutually exclusive features



Transferring data from on premises to cloud

- Most popular data destination is **Google Cloud Storage**
- Options:
 - **Online Transfer:** Use gsutil or API to transfer data to Google Cloud Storage
 - Good for one time transfers
 - **Storage Transfer Service:** Recommended for large-scale (petabytes) online data transfers from your private data centers, AWS, Azure, and Google Cloud
 - You can set up a repeating schedule
 - Supports incremental transfer (only transfer changed objects)
 - Reliable and fault tolerant - continues from where it left off in case of errors
 - **Storage Transfer Service vs gsutil:**
 - gsutil is recommended only when you are transferring less than 1 TB from on-premises or another GCS bucket
 - Storage Transfer Service is recommended if either of the conditions is met:
 - Transferring more than 1 TB from anywhere
 - Transferring from another cloud
 - **Transfer Appliance:** Physical transfer using an appliance



Migrating Data with Transfer Appliance

- **Transfer Appliance:** Copy, ship and upload data to GCS
 - Recommended if your data size is greater than 20TB
 - OR online transfer takes > 1 week
 - **Process:**
 - Request an appliance
 - Upload your data
 - Ship the appliance back
 - Google uploads the data
 - Fast copy (upto 40Gbps)
 - AES 256 encryption - Customer-managed encryption keys
 - Order multiple devices (TA40, TA300) if need

	Physical Transfer	Physical / Online Transfer	Online Transfer				
	1 Mbps	10 Mbps	100 Mbps	1 Gbps	10 Gbps	100 Gbps	
1 GB	3 hours	18 minutes	2 minutes	11 seconds	1 second	0.1 seconds	
10 GB	30 hours	3 hours	18 minutes	2 minutes	11 seconds	1 second	
100 GB	12 days	30 hours	3 hours	18 minutes	2 minutes	11 seconds	
1 TB	124 days	12 days	30 hours	3 hours	18 minutes	2 minutes	
10 TB	3 years	124 days	12 days	30 hours	3 hours	18 minutes	
100 TB	34 years	3 years	124 days	12 days	30 hours	3 hours	
1 PB	340 years	34 years	3 years	124 days	12 days	30 hours	
10 PB	3,404 years	340 years	34 years	3 years	124 days	12 days	
100 PB	34,048 years	3,404 years	340 years	34 years	3 years	124 days	

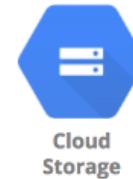
<https://cloud.google.com>

Understanding Cloud Storage Best Practices

- Avoid use of sensitive info in bucket or object names
- Store data in the **closest region** (to your users)
- Ramp up **request rate gradually**
 - No problems upto 1000 write requests per second or 5000 read requests per second
 - BUT beyond that, take at least 20 minutes to double request rates
- Use **Exponential backoff** if you receive 5xx (server error) or 429 (too many requests) errors
 - Retry after 1, 2, 4, 8, 16, .. seconds
- **Do NOT use sequential numbers or timestamp as object keys**
 - Recommended to use completely random object names
 - Recommended to add a hash value before the sequence number or timestamp
- **Use Cloud Storage FUSE to enable file system access to Cloud Storage**
 - Mount Cloud Storage buckets as file systems on Linux or macOS systems

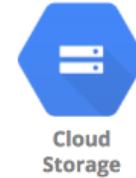
Cloud Storage - Command Line - gsutil - 1

- (REMEMBER) **gsutil** is the CLI for Cloud Storage (**NOT gcloud**)
- Cloud Storage (**gsutil**)
 - *gsutil mb gs://BKT_NAME* (Create Cloud Storage bucket)
 - *gsutil ls -a gs://BKT_NAME* (List current and non-current object versions)
 - *gsutil cp gs://SRC_BKT/SRC_OBJ gs://DESTN_BKT/NAME_COPY* (Copy objects)
 - -o 'GSUtil:encryption_key=ENCRYPTION_KEY' - Encrypt Object
 - *gsutil mv* (Rename/Move objects)
 - *gsutil mv gs://BKT_NAME/OLD_OBJ_NAME gs://BKT_NAME/NEW_OBJ_NAME*
 - *gsutil mv gs://OLD_BUCKET_NAME/OLD_OBJECT_NAME gs://NEW_BKT_NAME/NEW_OBJ_NAME*
 - *gsutil rewrite -s STORAGE_CLASS gs://BKT_NAME/OBJ_PATH* (Ex: Change Storage Class for objects)
 - *gsutil cp* : Upload and Download Objects
 - *gsutil cp LOCAL_LOCATION gs://DESTINATION_BKT_NAME/* (Upload)
 - *gsutil cp gs://BKT_NAME/OBJ_PATH LOCAL_LOCATION* (Download)



Cloud Storage - Command Line - gsutil - 2

- Cloud Storage (gsutil)
 - *gsutil versioning set on/off gs://BKT_NAME* (Enable/Disable Versioning)
 - *gsutil uniformbucketlevelaccess set on/off gs://BKT_NAME*
 - *gsutil acl ch* (Set Access Permissions for Specific Objects)
 - *gsutil acl ch -u AllUsers:R gs://BKT_NAME/OBJ_PATH* (Make specific object public)
 - *gsutil acl ch -u john.doe@example.com:WRITE gs://BKT_NAME/OBJ_PATH*
 - Permissions - READ (R), WRITE (W), OWNER (O)
 - Scope - User, allAuthenticatedUsers, allUsers(-u), Group (-g), Project (-p) etc
 - *gsutil acl set JSON_FILE gs://BKT_NAME*
 - *gsutil iam ch MBR_TYPE:MBR_NAME:IAM_ROLE gs://BKT_NAME* (Setup IAM role)
 - *gsutil iam ch user:me@myemail.com:objectCreator gs://BKT_NAME*
 - *gsutil iam ch allUsers:objectViewer gs://BKT_NAME* (make the entire bucket readable)
 - *gsutil signurl -d 10m YOUR_KEY gs://BUCKET_NAME/OBJECT_PATH* (Signed URL for temporary access)



Cloud Storage - Scenarios

In 28
Minutes

Scenario	Solution
I will frequently access objects in a bucket for 30 days. After that I don't expect to access objects at all. We have compliance requirements to store objects for 4 years. How can I minimize my costs?	Initial Storage Class - Standard Lifecycle policy: Move to Archive class after 30 days. Delete after 4 years.
I want to transfer 2 TB of data from Azure Storage to Cloud Storage	Use Cloud Storage Transfer Service
I want to transfer 40 TB of data from on premises to Cloud Storage	Use Transfer Appliance
Customer wants to manage their Keys	Customer-managed - Keys managed by customer in Cloud KMS
Regulatory compliance: Object should not be modified for 2 years	Configure and lock data retention policy

IAM

Typical identity management in the cloud

- You have **resources** in the cloud (examples - a virtual server, a database etc)
- You have **identities (human and non-human)** that need to access those resources and perform actions
 - For example: launch (stop, start or terminate) a virtual server
- How do you **identify users** in the cloud?
 - How do you configure resources they can access?
 - How can you configure what actions to allow?
- In GCP: *Identity and Access Management (Cloud IAM)* provides this service



Cloud IAM

Cloud Identity and Access Management (IAM)

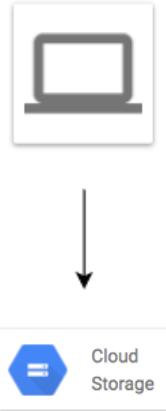
- **Authentication** (is it the right user?) and
- **Authorization** (do they have the right access?)
- **Identities** can be
 - A GCP User (Google Account or Externally Authenticated User)
 - A Group of GCP Users
 - An Application running in GCP
 - An Application running in your data center
 - Unauthenticated users
- Provides very **granular** control
 - Limit a single user:
 - to perform single action
 - on a specific cloud resource
 - from a specific IP address
 - during a specific time window



Cloud IAM

Cloud IAM Example

- I want to provide access to manage a specific cloud storage bucket to a colleague of mine:
 - Important Generic Concepts:
 - Member: My colleague
 - Resource: Specific cloud storage bucket
 - Action: Upload/Delete Objects
 - In Google Cloud IAM:
 - Roles: A set of permissions (to perform specific actions on specific resources)
 - Roles do NOT know about members. It is all about permissions!
 - How do you assign permissions to a member?
 - Policy: You assign (or bind) a role to a member
- 1: Choose a Role with right permissions (Ex: Storage Object Admin)
- 2: Create Policy binding member (your friend) with role (permissions)
- IAM in AWS is very different from GCP (Forget AWS IAM & Start FRESH!)
 - Example: Role in AWS is NOT the same as Role in GCP



IAM - Roles

- **Roles are Permissions:**
 - Perform some set of actions on some set of resources
- **Three Types:**
 - **Basic Roles (or Primitive roles)** - Owner/Editor/Viewer
 - Viewer(roles.viewer) - Read-only actions
 - Editor(roles.editor) - Viewer + Edit actions
 - Owner(roles.owner) - Editor + Manage Roles and Permissions + Billing
 - EARLIEST VERSION: Created before IAM
 - NOT RECOMMENDED: **Don't use in production**
 - **Predefined Roles** - Fine grained roles predefined and managed by Google
 - Different roles for different purposes
 - **Examples:** Storage Admin, Storage Object Admin, Storage Object Viewer, Storage Object Creator
 - **Custom Roles** - When predefined roles are NOT sufficient, you can create your own custom roles



Cloud IAM

IAM - Predefined Roles - Example Permissions

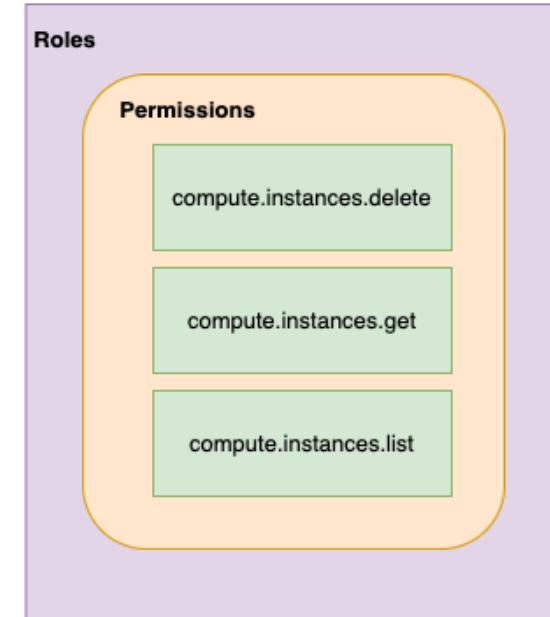
- Important Cloud Storage Roles:
 - **Storage Admin (roles/storage.admin)**
 - storage.buckets.*
 - storage.objects.*
 - **Storage Object Admin (roles/storage.objectAdmin)**
 - storage.objects.*
 - **Storage Object Creator (roles/storage.objectCreator)**
 - storage.objects.create
 - **Storage Object Viewer (roles/storage.objectViewer)**
 - storage.objects.get
 - storage.objects.list
- All four roles have these permissions:
 - resourcemanager.projects.get
 - resourcemanager.projects.list



Cloud IAM

IAM - Most Important Concepts - A Review

- Member : Who?
- Roles : Permissions (What Actions? What Resources?)
- Policy : Assign Permissions to Members
 - Map Roles (What?) , Members (Who?) and Conditions (Which Resources?, When?, From Where?)
 - Remember: Permissions are NOT directly assigned to Member
 - Permissions are represented by a Role
 - Member gets permissions through Role!
- A Role can have multiple permissions
- You can assign multiple roles to a Member



IAM policy

- Roles are assigned to users through **IAM Policy** documents
- Represented by a **policy object**
 - Policy object has list of bindings
 - A binding, binds a role to list of members
- Member type is identified by **prefix**:
 - Example: user, serviceaccount, group or domain



Cloud IAM

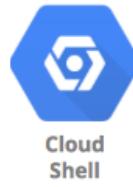
IAM policy - Example

```
{  
  "bindings": [  
    {  
      "role": "roles/storage.objectAdmin",  
      "members": [  
        "user:you@in28minutes.com",  
        "serviceAccount:myAppName@appspot.gserviceaccount.com",  
        "group:administrators@in28minutes.com",  
        "domain:google.com"  
      ]  
    },  
    {  
      "role": "roles/storage.objectViewer",  
      "members": [  
        "user:you@in28minutes.com"  
      ],  
      "condition": {  
        "title": "Limited time access",  
        "description": "Only upto Feb 2022",  
        "expression": "request.time < timestamp('2022-02-01T00:00:00.000Z')",  
      }  
    }  
  ]  
}
```

Playing With IAM

- **gcloud:** Playing with IAM

- **gcloud compute project-info describe** - Describe current project
- **gcloud auth login** - Access the Cloud Platform with Google user credentials
- **gcloud auth revoke** - Revoke access credentials for an account
- **gcloud auth list** - List active accounts
- **gcloud projects**
 - **gcloud projects add-iam-policy-binding** - Add IAM policy binding
 - **gcloud projects get-iam-policy** - Get IAM policy for a project
 - **gcloud projects remove-iam-policy-binding** - Remove IAM policy binding
 - **gcloud projects set-iam-policy** - Set the IAM policy
 - **gcloud projects delete** - Delete a project
- **gcloud iam**
 - **gcloud iam roles describe** - Describe an IAM role
 - **gcloud iam roles create** - create an iam role(--project, --permissions, --stage)
 - **gcloud iam roles copy** - Copy IAM Roles



Service Accounts

- Scenario: An Application on a VM needs access to cloud storage
 - You DONT want to use personal credentials to allow access
- (RECOMMENDED) Use **Service Accounts**
 - Identified by an email address (Ex: id-compute@developer.gserviceaccount.com)
 - Does NOT have password
 - Has a **private/public RSA key-pairs**
 - Can't login via browsers or cookies
- Service account types:
 - **Default service account** - Automatically created when some services are used
 - (NOT RECOMMENDED) Has **Editor role** by default
 - **User Managed** - User created
 - (RECOMMENDED) Provides fine grained access control
 - **Google-managed service accounts** - Created and managed by Google
 - Used by GCP to perform operations on user's behalf
 - In general, we DO NOT need to worry about them



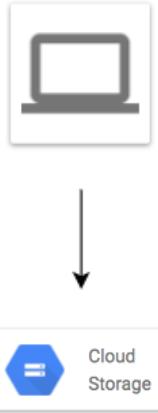
Use case 1 : VM <-> Cloud Storage



- 1: Create a Service Account Role with the right permissions
- 2: Assign Service Account role to VM instance
- **Uses Google Cloud-managed keys:**
 - Key generation and use are automatically handled by IAM when we assign a service account to the instance
 - Automatically rotated
 - No need to store credentials in config files
- **Do NOT delete** service accounts used by running instances:
 - Applications running on those instances will lose access!

Use case 2 : On Prem <-> Cloud Storage (Long Lived)

- You **CANNOT** assign Service Account directly to an On Prem App
- **1:** Create a **Service Account** with right permissions
- **2:** Create a **Service Account User Managed Key**
 - *gcloud iam service-accounts keys create*
 - Download the service account key file
 - Keep it secure (It can be used to impersonate service account)!
- **3:** Make the service account key file accessible to your application
 - Set environment variable GOOGLE_APPLICATION_CREDENTIALS
 - *export GOOGLE_APPLICATION_CREDENTIALS="/PATH_TO_KEY_FILE"*
- **4: Use Google Cloud Client Libraries**
 - Google Cloud Client Libraries use a library - Application Default Credentials (ADC)
 - ADC uses the service account key file if env var GOOGLE_APPLICATION_CREDENTIALS exists!



Use case 3 : On Prem <-> Google Cloud APIs (Short Lived)

- Make calls from outside GCP to Google Cloud APIs with short lived permissions
 - Few hours or shorter
 - Less risk compared to sharing service account keys!
- Credential Types:
 - OAuth 2.0 access tokens
 - OpenID Connect ID tokens
 - Self-signed JSON Web Tokens (JWTs)
- Examples:
 - When a member needs elevated permissions, he can assume the service account role (Create OAuth 2.0 access token for service account)
 - OpenID Connect ID tokens is recommended for service to service authentications:
 - A service in GCP needs to authenticate itself to a service in other cloud



Service Account Use case Scenarios

Scenario	Solution
Application on a VM wants to talk to a Cloud Storage bucket	Configure the VM to use a Service Account with right permissions
Application on a VM wants to put a message on a Pub Sub Topic	
Configure the VM to use a Service Account with right permissions	
Is Service Account an identity or a resource?	It is both. You can attach roles with Service Account (identity). You can let other members access a SA by granting them a role on the Service Account (resource).
VM instance with default service account in Project A needs to access Cloud Storage bucket in Project B	In project B, add the service account from Project A and assign Storage Object Viewer Permission on the bucket

ACL (Access Control Lists)



- **ACL:** Define **who** has access to your buckets and objects, as well as **what level** of access they have
- **How is this different from IAM?**
 - IAM permissions apply to all objects within a bucket
 - ACLs can be used to customized specific accesses to different objects
- User gets access if he is allowed by either IAM or ACL!
- (Remember) **Use IAM for common permissions** to all objects in a bucket
- (Remember) **Use ACLs if you need to customize access to individual objects**

Access Control - Overview

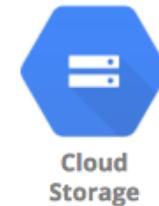
- How do you control access to objects in a Cloud Storage bucket?
- Two types of access controls:
 - Uniform (Recommended) - Uniform bucket level access using IAM
 - Fine-grained - Use IAM and ACLs to control access:
 - Both bucket level and individual object level permissions
- Use Uniform access when all users have same level of access across all objects in a bucket
- Fine grained access with ACLs can be used when you need to customize the access at an object level
 - Give a user specific access to edit specific objects in a bucket



Cloud IAM

Cloud Storage - Signed URL

- You would want to **allow a user limited time access** to your objects:
 - Users do NOT need Google accounts
- Use **Signed URL functionality**
 - A URL that gives **permissions for limited time duration** to perform specific actions
- To create a **Signed URL**:
 - 1: Create a key (YOUR_KEY) for the Service Account/User with the desired permissions
 - 2: Create Signed URL with the key:
 - *gsutil signurl -d 10m YOUR_KEY gs://BUCKET_NAME/OBJECT_PATH*



Cloud Storage - Static website

In 28
Minutes



- 1: Create a bucket with the **same name** as website name (Name of bucket should match DNS name of the website)
 - Verify that the domain is owned by you
- 2: Copy the files to the bucket
 - Add index and error html files for better user experience
- 3: Add member **allUsers** and grant **Storage Object Viewer** option
 - Select **Allow Public Access**

IAM - Scenarios

In 28
Minutes

Scenario	Solution
An Application on a GCE VM needs access to cloud storage	Use a Service Account (Google Cloud-managed keys)
An Application on premises needs access to cloud storage	Use Service Account User Managed Key
Allow a user limited time access to your objects	Signed URL
Customize access to a subset of objects in a bucket	Use ACL (Access Control Lists)
Permission is allowed by IAM but NOT by ACL. Will user be able to access the object?	Yes.

Database Fundamentals

Databases Primer

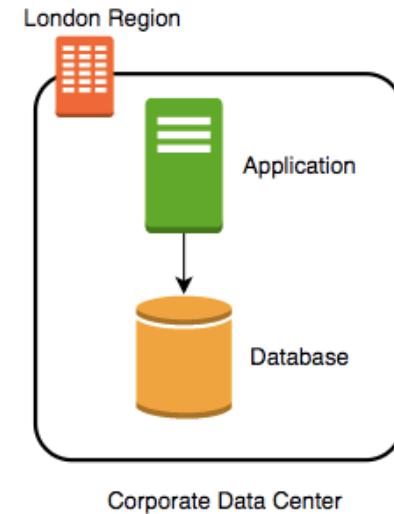
- Databases provide **organized** and **persistent** storage for your data
- To **choose between different database types**, we would need to understand:
 - Availability
 - Durability
 - RTO
 - RPO
 - Consistency
 - Transactions etc
- Let's get started on a **simple journey** to understand these



Database

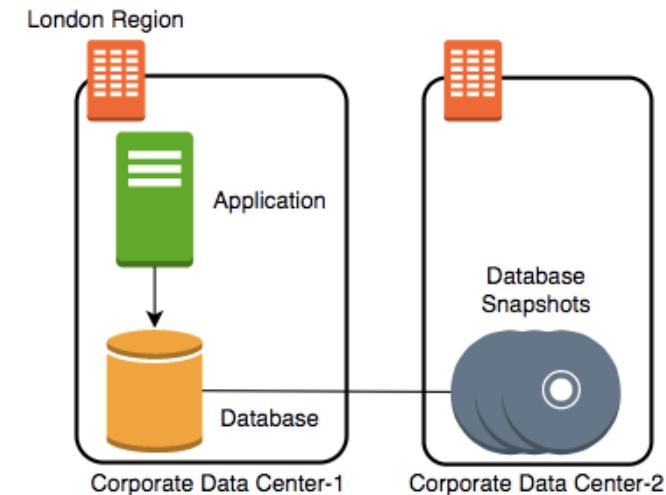
Database - Getting Started

- Imagine a database deployed in a data center in London
- Let's consider some challenges:
 - Challenge 1: Your database will go down if the data center crashes or the server storage fails
 - Challenge 2: You will lose data if the database crashes



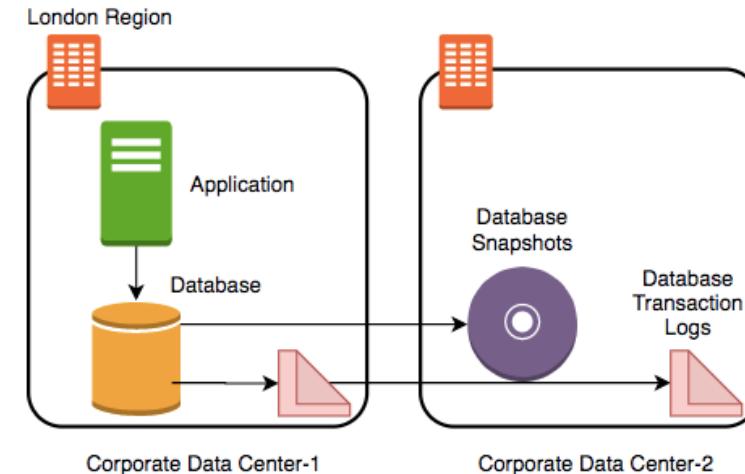
Database - Snapshots

- Let's automate taking copy of the database (**take a snapshot**) every hour to another data center in London
- Let's consider some challenges:
 - **Challenge 1:** Your database will go down if the data center crashes
 - **Challenge 2 (PARTIALLY SOLVED):** You will lose data if the database crashes
 - You can setup database from latest snapshot. But depending on when failure occurs you can lose up to an hour of data
 - **Challenge 3(NEW):** Database will be slow when you take snapshots



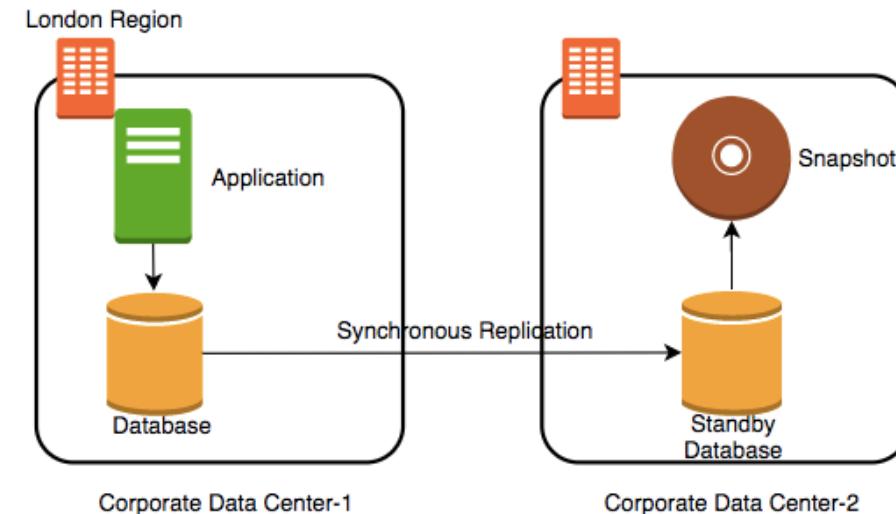
Database - Transaction Logs

- Let's add transaction logs to database and create a process to copy it over to the second data center
- Let's consider some challenges:
 - **Challenge 1:** Your database will go down if the data center crashes
 - **Challenge 2 (SOLVED):** You will lose data if the database crashes
 - You can setup database from latest snapshot and apply transaction logs
 - **Challenge 3:** Database will be slow when you take snapshots



Database - Add a Standby

- Let's add a **standby database** in the second data center with replication
- Let's consider some challenges:
 - **Challenge 1 (SOLVED)**: Your database will go down if the data center crashes
 - You can switch to the standby database
 - **Challenge 2 (SOLVED)**: You will lose data if the database crashes
 - **Challenge 3 (SOLVED)**: Database will be slow when you take snapshots
 - Take snapshots from standby.
 - Applications connecting to master will get good performance always



Availability and Durability

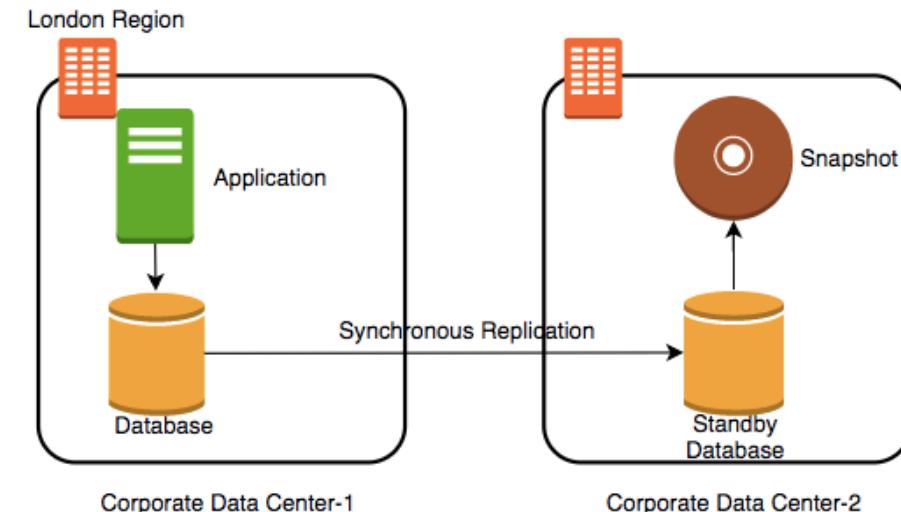
- **Availability**
 - Will I be able to access my data now and when I need it?
 - Percentage of time an application provides the operations expected of it
- **Durability**
 - Will my data be available after 10 or 100 or 1000 years?
- Examples of measuring availability and durability:
 - 4 9's - 99.99
 - 11 9's - 99.999999999
- Typically, an **availability of four 9's** is considered very good
- Typically, a **durability of eleven 9's** is considered very good

Availability

Availability	Downtime (in a month)	Comment
99.95%	22 minutes	
99.99% (4 9's)	4 and 1/2 minutes	Typically online apps aim for 99.99% (4 9's) availability
99.999% (5 9's)	26 seconds	Achieving 5 9's availability is tough

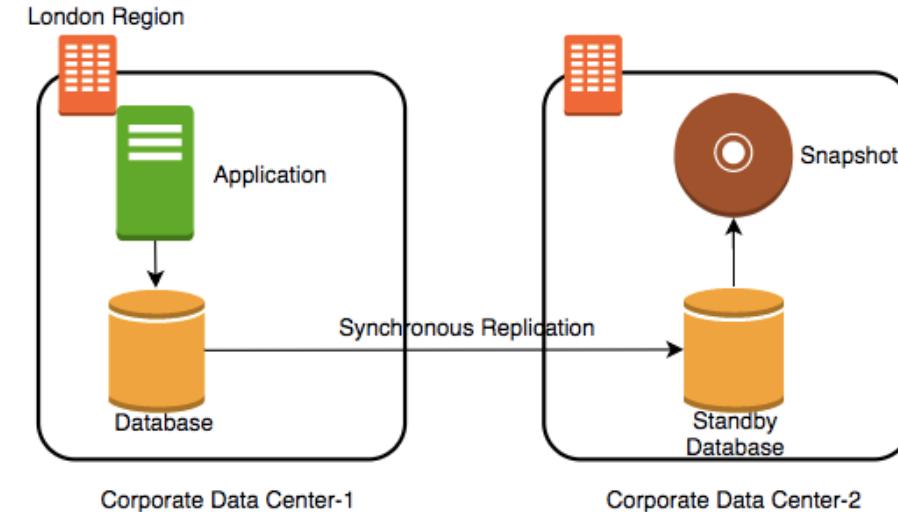
Durability

- What does a durability of 11 9's mean?
 - If you store one million files for ten million years, you would expect to lose one file
- Why should durability be high?
 - Because we hate losing data
 - Once we lose data, it is gone



Increasing Availability and Durability of Databases

- **Increasing Availability:**
 - Have multiple standbys available OR distribute the database
 - in multiple Zones
 - in multiple Regions
- **Increasing Durability:**
 - Multiple copies of data (standbys, snapshots, transaction logs and replicas)
 - in multiple Zones
 - in multiple Regions
- **Replicating data comes with its own challenges!**
 - We will talk about them a little later



Database Terminology : RTO and RPO

- Imagine a **financial transaction being lost**
- Imagine a **trade being lost**
- Imagine a **stock exchange going down for an hour**
- Typically businesses are fine with some downtime but they hate losing data
- Availability and Durability are technical measures
- How do we measure **how quickly we can recover from failure?**
 - **RPO (Recovery Point Objective)**: Maximum acceptable period of data loss
 - **RTO (Recovery Time Objective)**: Maximum acceptable downtime
- Achieving **minimum RTO and RPO is expensive**
- **Trade-off** based on the criticality of the data



Database

Question - RTO and RPO

- You are running an application in VM instance storing its data on a persistent data storage. You are taking snapshots every 48 hours. If the VM instance crashes, you can manually bring it back up in 45 minutes from the snapshot.

What is your RTO and RPO?

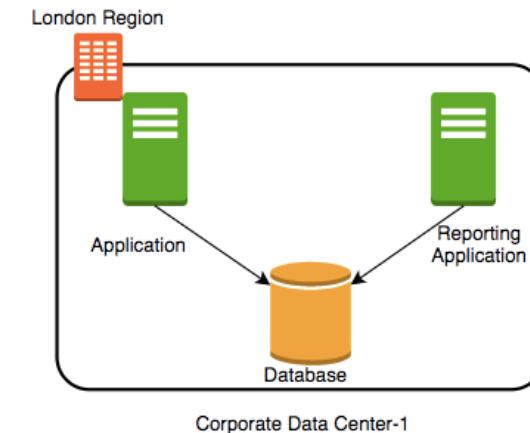
- RTO - 45 minutes
- RPO - 48 hours

Achieving RTO and RPO - Failover Examples

Scenario	Solution
Very small data loss (RPO - 1 minute) Very small downtime (RTO - 5 minutes)	Hot standby - Automatically synchronize data Have a standby ready to pick up load Use automatic failover from master to standby
Very small data loss (RPO - 1 minute) BUT I can tolerate some downtimes (RTO - 15 minutes)	Warm standby - Automatically synchronize data Have a standby with minimum infrastructure Scale it up when a failure happens
Data is critical (RPO - 1 minute) but I can tolerate downtime of a few hours (RTO - few hours)	Create regular data snapshots and transaction logs Create database from snapshots and transactions logs when a failure happens
Data can be lost without a problem (for example: cached data)	Failover to a completely new server

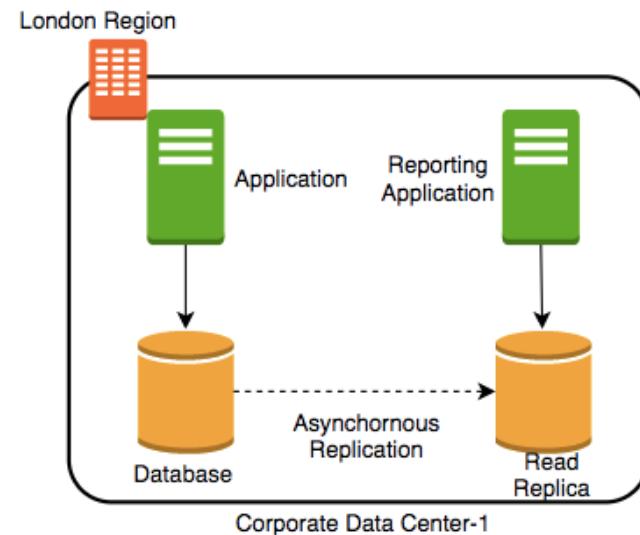
(New Scenario) Reporting and Analytics Applications

- New reporting and analytics applications are being launched using the same database
 - These applications will ONLY read data
- Within a few days you see that the database performance is impacted
- How can we fix the problem?
 - Vertically scale the database - increase CPU and memory
 - Create a database cluster (Distribute the database) - Typically database clusters are expensive to setup
 - Create read replicas - Run read only applications against read replicas



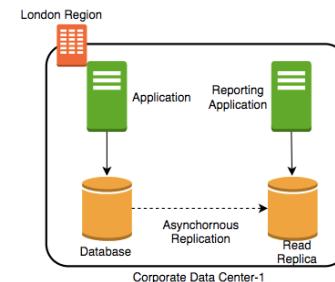
Database - Read Replicas

- Add read replica
- Connect reporting and analytics applications to read replica
- Reduces load on the master databases
- Upgrade read replica to master database (supported by some databases)
- Create read replicas in multiple regions
- Take snapshots from read replicas



Consistency

- How do you ensure that data in multiple database instances (standbys and replicas) is updated simultaneously?
- **Strong consistency** - Synchronous replication to all replicas
 - Will be slow if you have multiple replicas or standbys
- **Eventual consistency** - Asynchronous replication. A little lag - few seconds - before the change is available in all replicas
 - In the intermediate period, different replicas might return different values
 - Used when scalability is more important than data integrity
 - Examples : Social Media Posts - Facebook status messages, Twitter tweets, LinkedIn posts etc
- **Read-after-Write consistency** - Inserts are immediately available
 - However, updates would have eventual consistency



Database Categories

- There are **several categories** of databases:
 - Relational (OLTP and OLAP), Document, Key Value, Graph, In Memory among others
- **Choosing type of database** for your use case is not easy. A few factors:
 - Do you want a **fixed schema**?
 - Do you want flexibility in defining and changing your schema? (schemaless)
 - What level of **transaction properties** do you need? (atomicity and consistency)
 - What kind of **latency** do you want? (seconds, milliseconds or microseconds)
 - How many **transactions** do you expect? (hundreds or thousands or millions of transactions per second)
 - How much **data** will be stored? (MBs or GBs or TBs or PBs)
 - and a lot more...



Cloud SQL



Cloud Spanner



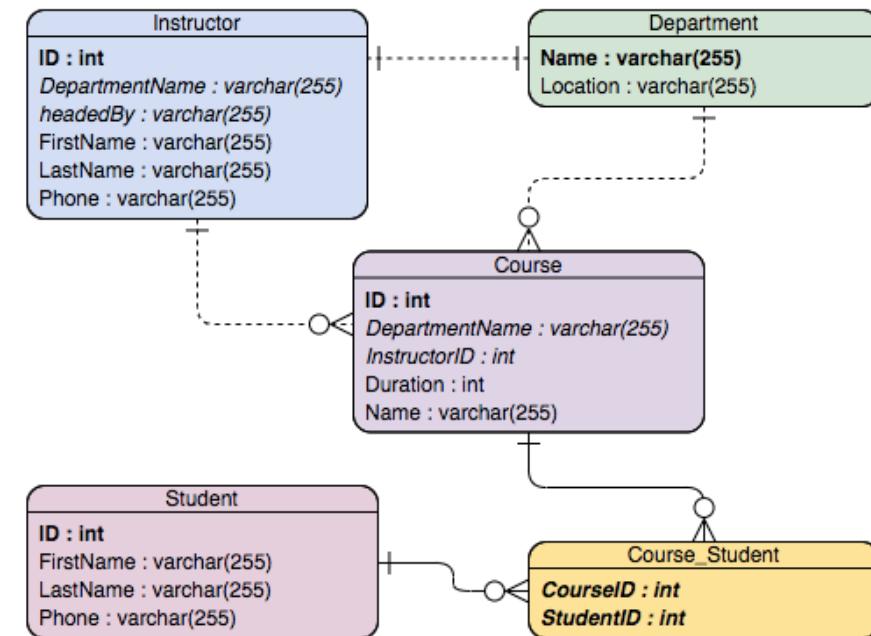
Cloud Datastore



BigQuery

Relational Databases

- This was the **only** option until a decade back!
- Most **popular** (or unpopular) type of databases
- **Predefined schema** with tables and relationships
- Very **strong transactional** capabilities
- Used for
 - OLTP (Online Transaction Processing) use cases and
 - OLAP (Online Analytics Processing) use cases



Relational Database - OLTP (Online Transaction Processing)

In 28
Minutes

- Applications where large number of users make large number of small transactions
 - small data reads, updates and deletes
- Use cases:
 - Most traditional applications, ERP, CRM, e-commerce, banking applications
- Popular databases:
 - MySQL, Oracle, SQL Server etc
- Recommended Google Managed Services:
 - **Cloud SQL** : Supports PostgreSQL, MySQL, and SQL Server for regional relational databases (upto a few TBs)
 - **Cloud Spanner**: Unlimited scale (multiple PBs) and 99.999% availability for global applications with horizontal scaling



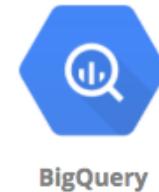
Cloud SQL



Cloud Spanner

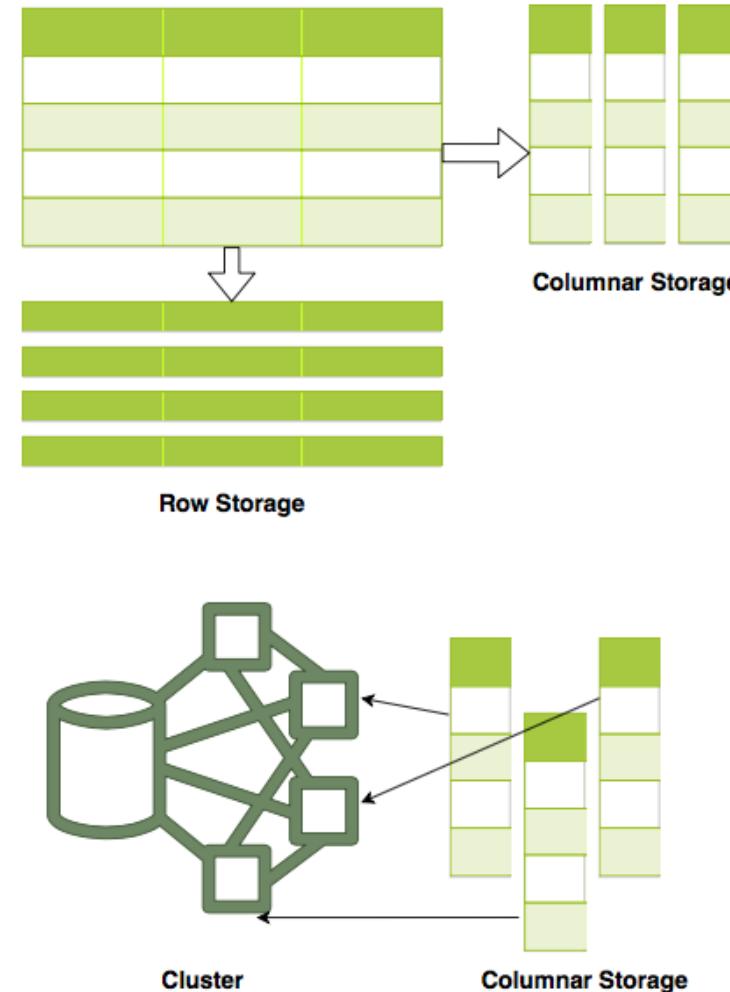
Relational Database - OLAP (Online Analytics Processing)

- Applications allowing users to **analyze petabytes of data**
 - Examples : Reporting applications, Data ware houses, Business intelligence applications, Analytics systems
 - Sample application : Decide insurance premiums analyzing data from last hundred years
 - Data is consolidated from multiple (transactional) databases
- Recommended GCP Managed Service
 - BigQuery: Petabyte-scale distributed data ware house



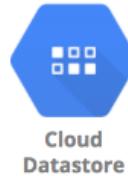
Relational Databases - OLAP vs OLTP

- OLAP and OLTP use similar data structures
- BUT **very different approach in how data is stored**
- **OLTP databases use row storage**
 - Each table row is stored together
 - Efficient for processing small transactions
- **OLAP databases use columnar storage**
 - Each table column is stored together
 - **High compression** - store petabytes of data efficiently
 - **Distribute data** - one table in multiple cluster nodes
 - **Execute single query across multiple nodes** - Complex queries can be executed efficiently



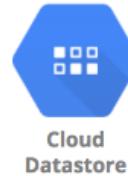
NoSQL Databases

- **New approach** (actually NOT so new!) to building your databases
 - NoSQL = not only SQL
 - Flexible schema
 - Structure data **the way your application needs it**
 - Let the schema evolve with time
 - Horizontally scale to petabytes of data with millions of TPS
- **NOT a 100% accurate generalization** but a great starting point:
 - Typical NoSQL databases trade-off "Strong consistency and SQL features" to achieve "scalability and high-performance"
- **Google Managed Services:**
 - Cloud Firestore (Datastore)
 - Cloud BigTable



Cloud Firestore (Datastore) vs Cloud BigTable

- **Cloud Datastore** - Managed serverless NoSQL document database
 - Provides ACID transactions, SQL-like queries, indexes
 - Designed for transactional mobile and web applications
 - Firestore (next version of Datastore) adds:
 - Strong consistency
 - Mobile and Web client libraries
 - Recommended for small to medium databases (0 to a few Terabytes)
- **Cloud BigTable** - Managed, scalable NoSQL wide column database
 - NOT serverless (You need to create instances)
 - Recommend for data size > 10 Terabytes to several Petabytes
 - Recommended for large analytical and operational workloads:
 - NOT recommended for transactional workloads (Does NOT support multi row transactions - supports ONLY Single-row transactions)



In-memory Databases

- Retrieving data from memory is much faster than retrieving data from disk
- In-memory databases like Redis deliver microsecond latency by storing **persistent data in memory**
- Recommended GCP Managed Service
 - Memory Store
- **Use cases** : Caching, session management, gaming leader boards, geospatial applications



Memorystore

Databases - Summary

Database Type	GCP Services	Description
Relational OLTP databases	Cloud SQL, Cloud Spanner	<p>Transactional usecases needing predefined schema and very strong transactional capabilities (Row storage)</p> <p>Cloud SQL: MySQL, PostgreSQL, SQL server DBs</p> <p>Cloud Spanner: Unlimited scale and 99.999% availability for global applications with horizontal scaling</p>
Relational OLAP databases	BigQuery	<p>Columnar storage with predefined schema.</p> <p>Datawarehousing & BigData workloads</p>
NoSQL Databases	Cloud Firestore (Datastore) , Cloud BigTable	<p>Apps needing quickly evolving structure (schema-less)</p> <p>Cloud Firestore - Serverless transactional document DB supporting mobile & web apps. Small to medium DBs (0 - few TBs)</p> <p>Cloud BigTable - Large databases(10 TB - PBs). Streaming (IOT), analytical & operational workloads. NOT serverless.</p>
In memory	Cloud Memorystore	Applications needing microsecond responses

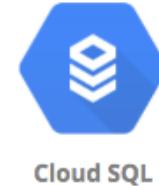
Databases - Scenarios

In 28
Minutes

Scenario	Solution
A start up with quickly evolving schema (table structure)	Cloud Datastore/Firestore
Non relational db with less storage (10 GB)	Cloud Datastore
Transactional global database with predefined schema needing to process million of transactions per second	CloudSpanner
Transactional local database processing thousands of transactions per second	Cloud SQL
Cache data (from database) for a web application	MemoryStore
Database for analytics processing of petabytes of data	BigQuery
Database for storing huge volumes stream data from IOT devices	BigTable
Database for storing huge streams of time series data	BigTable

Relational Databases For Transactional Usecases

- **Fully Managed Relational Database service**
 - Configure your needs and do NOT worry about managing the database
 - Supports MySQL, PostgreSQL, and SQL Server
 - Regional Service providing High Availability (99.95%)
 - Use SSDs or HDDs (For best performance: use SSDs)
 - Up to 416 GB of RAM and 30 TB of data storage
- **Use Cloud SQL for simple relational use cases:**
 - To migrate local MySQL, PostgreSQL, and SQL Server databases
 - To reduce your maintenance cost for a simple relational database
 - (REMEMBER) Use Cloud Spanner(Expensive\$\$\$\$) instead of Cloud SQL if:
 - You have huge volumes of relational data (TBs) OR
 - You need infinite scaling for a growing application (to TBs) OR
 - You need a Global (distributed across multiple regions) Database OR
 - You need higher availability (99.999%)

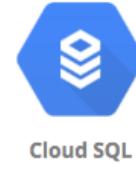


Cloud SQL - Features

In 28
Minutes

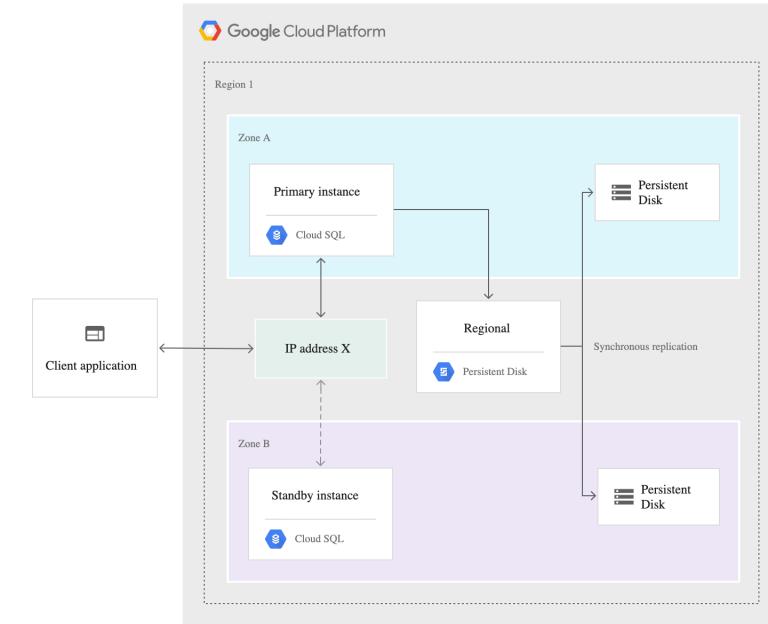
- **Important Cloud SQL Features:**

- Automatic encryption (tables/backups), maintenance and updates
- High availability and failover:
 - Create a Standby with automatic failover
 - Pre requisites: Automated backups and Binary logging
- Read replicas for read workloads:
 - Options: Cross-zone, Cross-region and External (NON Cloud SQL DB)
 - Pre requisites: Automated backups and Binary logging
- Automatic storage increase without downtime (for newer versions)
- Point-in-time recovery: Enable binary logging
- Backups (Automated and on-demand backups)
- Supports migration from other sources
 - Use Database Migration Service (DMS)
- You can export data from UI (console) or gcloud with formats:
 - SQL (Recommended if you import data into other databases) and CSV



Cloud SQL - High Availability

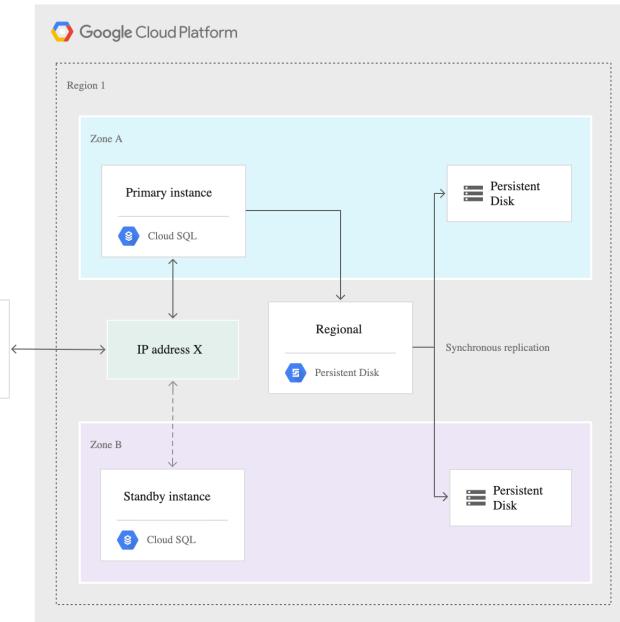
- Create a High Availability (HA) Configuration
 - Choose Primary and Secondary zones within a region
 - You will have two instances : Primary and Secondary instances
- Changes from primary are replicated synchronously to secondary
- In case of Zonal failure, automatic failover to secondary instance:
 - If Primary zone becomes available, failover does not revert automatically
- (Remember) High Availability setup CANNOT be used as a Read Replica



source:cloud.google.com

Understanding Cloud SQL Best Practices

- Use **Cloud SQL Proxy**:
 - Securely connect to Cloud SQL from your apps (GAE, Cloud Functions, Cloud Run, GKE etc)
- Understand **Scalability**
 - Enable **HA configuration** for high availability
 - Primary instance and a standby instance created in the same Region (Remember - Regional)
 - Read replicas help you **offload read workloads** (reporting, analytics etc)
 - (Remember) Read replicas do NOT increase availability
 - Prefer **Number of small Cloud SQL instances** to having one large instance
 - Cloud SQL cannot scale horizontally for writes



<http://cloud.google.com>

Understanding Cloud SQL Best Practices - 2

- Understand **Backup and Export** options:
 - Backups are lightweight and provide point in time recovery
 - BUT Backups are deleted when an instance is deleted
 - AND you can't back up a single database or table
 - Exports take longer but they provide you with more flexibility
 - You can export a single database or table
 - (Remember) Exporting large databases can impact the performance of a Cloud SQL database
 - Use **serverless export** (flag - offload) to reduce impact
 - Cloud SQL creates a separate, temporary instance to offload the export operation
 - Import/Export in multiple small batches instead of large batches

Cloud Spanner

- Fully managed, mission critical, relational(SQL), globally distributed database with VERY high availability (99.999%)
 - Strong transactional consistency at global scale
 - Scales to PBs of data with automatic sharding
- Cloud Spanner scales horizontally for reads and writes
 - Configure no of nodes
 - (REMEMBER) In comparison, Cloud SQL provides read replicas:
 - BUT you cannot horizontally scale write operations with Cloud SQL!
- Regional and Multi-Regional configurations
- **Expensive** (compared to Cloud SQL): Pay for nodes & storage
- **Data Export:** Use Cloud Console to export data
 - Other option is to use Data flow to automate export
 - No gcloud export option

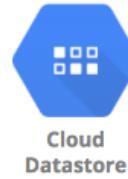


Cloud Spanner

NoSQL Databases

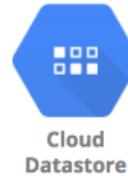
Cloud Datastore and Firestore

- **Datastore** - Highly scalable NoSQL Document Database
 - Automatically scales and partitions data as it grows
 - Recommended for upto a few TBs of data
 - For bigger volumes, BigTable is recommended
 - Supports Transactions, Indexes and SQL like queries (GQL)
 - Does NOT support Joins or Aggregate (sum or count) operations
 - For use cases needing flexible schema with transactions
 - Examples: User Profile and Product Catalogs
 - Structure: Kind > Entity (Use namespaces to group entities)
 - You can export data ONLY from gcloud (NOT from cloud console)
 - Export contains a metadata file and a folder with the data
- **Firestore** = Datastore++ : Optimized for multi device access
 - Offline mode and data synchronization across multiple devices - mobile, IOT etc
 - Provides client side libraries - Web, iOS, Android and more
 - Offers Datastore and Native modes



Understanding Cloud Datastore Best Practices

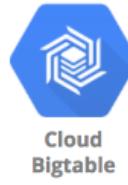
- Cloud Datastore is a **document store with flexible schema**
 - Recommended for storing things like user profiles
 - Another Use Case: Index for objects stored in Cloud Storage
 - You want to allow users to upload their profile pictures:
 - Store objects (pictures) in Cloud Storage
 - Enable quick search by storing metadata (like ids and cloud storage bucket, object details) in Cloud Datastore
- Design your keys and indexes carefully:
 - Avoid monotonically increasing values as keys
 - NOT RECOMMENDED - 1, 2, 3, ..., OR "Customer1", "Customer2", "Customer3", ... or timestamps
 - RECOMMENDED - Use allocateIds() for well-distributed numeric IDs
 - Create indexes only if they would be used in queries
 - For ad hoc queries on large datasets without pre-defined indexes, BigQuery is recommended!
- Prefer batch operations (to single read, write or delete operations):
 - More efficient as multiple operations are performed with same overhead as one operation



Cloud BigTable

In 28
Minutes

- **Petabyte scale, wide column NoSQL DB (HBase API compatible)**
 - Designed for huge volumes of analytical and operational data
 - IOT Streams, Analytics, Time Series Data etc
 - Handle millions of read/write TPS at very low latency
 - Single row transactions (multi row transactions NOT supported)
- **NOT serverless:** You need to create a server instance (Use SSD or HDD)
 - Scale horizontally with multiple nodes (No downtime for cluster resizing)
- **CANNOT export data using cloud console or gcloud:**
 - Either use a Java application (java -jar JAR export\import) OR
 - Use HBase commands
- Use cbt command line tool to work with BigTable (NOT gcloud)
 - Ex: `cbt createtable my-table`



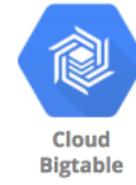
Cloud BigTable - Wide Column Database

Rowid	Column Family 1			Column Family 2			Column Family 3		
	col1	col2	col3	col1	col2	col3	col1	col2	col3
1									
2									
3									

- At the most basic level, each table is a sorted key/value map
 - Each value in a row is indexed using a key - **row key**
 - Related columns are grouped into column families
 - Each column is identified by using column-family:column-qualifier(or name)
- This structure supports high read and write throughput at low latency
 - Advantages : Scalable to petabytes of data with millisecond responses upto millions of TPS
- **Use cases :** IOT streams, graph data and real time analytics (time-series data, financial data - transaction histories, stock prices etc)
- **Cloud Dataflow :** Used to export data from BigTable to CloudStorage

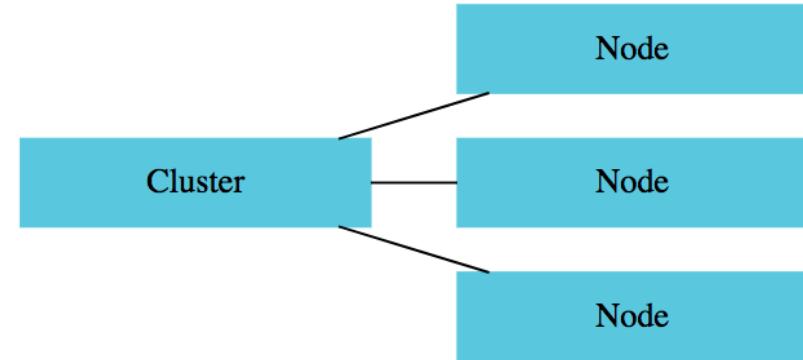
Designing BigTable Tables

- Two things you should know before starting with Bigtable:
 - What data do you want to store? (format, columns etc)
 - What would your frequently used queries look like (ranked by usage)?
- Design your table: Cloud Bigtable is a **key/value store**
 - Each table has **ONLY ONE** index, the row key
 - **Design your row key** based on your frequently used queries
 - You can have multiple row key segments - Separated by a delimiter (ex: ranga#123456#abcd)
 - Avoid sequential row keys (timestamps or sequential numbers)
 - Include timestamp as part of your row key IF you plan to retrieve data based on the timestamp
 - Use reversed timestamp (Ex: Long.MAX_VALUE - timestamp) if you frequently query recent data
 - Records will be ordered from most recent to least recent
 - After your design your table:
 - Test (heavy load for several minutes + one hour simulation) with atleast 30 GB of test data
 - Analyze usage patterns with **Key Visualizer tool** for Cloud Bigtable



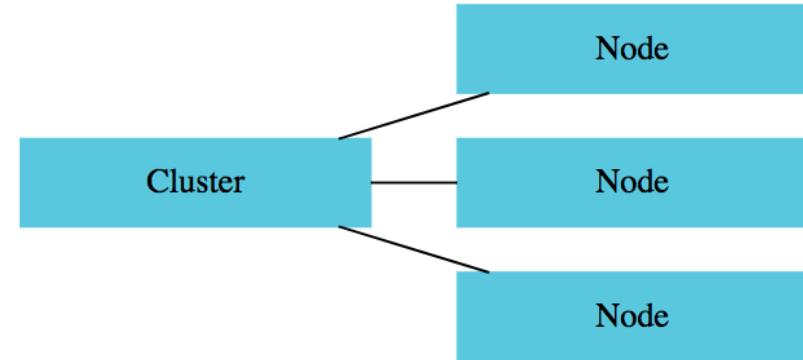
Understanding Cloud BigTable Best Practices

- Recommended for streaming IOT & time series data
- **Automatically shards data into multiple tablets across nodes in cluster:**
 - **Goal 1:** Have same amount of data on each node
 - **Goal 2:** Distribute reads and writes equally across all nodes
 - **(REMEMBER)** Pre-test with heavy load for a few minutes before you run your tests
 - Gives Bigtable a chance to balance data across your nodes
- Cloud Bigtable supports SSD or HDD storage:
 - **SSD** - For most usecases
 - **HDD** - For large non latency-sensitive data sets of size >10 TB with very very few reads



Understanding Cloud BigTable Best Practices - Replication

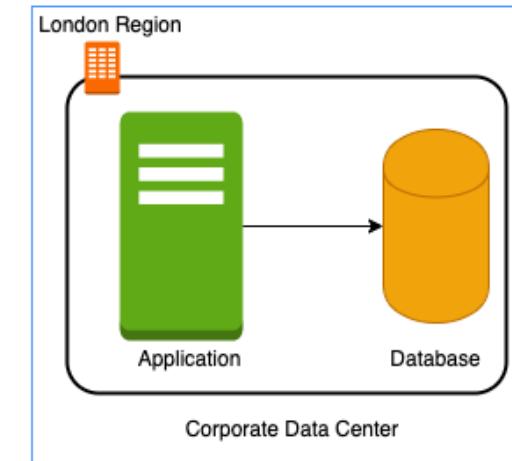
- You can create a Cloud Bigtable instance with **more than one cluster** to enable **replication** (**Cross Region or Cross Zone**) :
 - Independent copy of data is stored in each cluster (in the zone of the cluster)
 - Bigtable automatically replicates changes
 - Replication improves durability and availability of your data
 - Stores separate copies in multiple zones or regions
 - Can automatically failover between clusters if needed
 - Replication helps you to put data closer to your customers
 - Configure an application profile, or app profile with routing policy of multi-cluster routing
 - Automatically route to nearest cluster in an instance



Networking

Need for Google Cloud VPC

- In a corporate network or an on-premises data center:
 - Can anyone on the internet **see the data exchange** between the application and the database?
 - No
 - Can anyone from internet **directly connect to your database?**
 - Typically NO.
 - You need to connect to your corporate network and then access your applications or databases.
- Corporate network provides a **secure internal network** protecting your resources, data and communication from external users
- How do you do create **your own private network** in the cloud?
 - Enter **Virtual Private Cloud (VPC)**



Google Cloud VPC (Virtual Private Cloud)

- Your own **isolated network** in GCP cloud
 - Network traffic within a VPC is isolated (not visible) from all other Google Cloud VPCs
- You **control all the traffic** coming in and going outside a VPC
- **(Best Practice)** Create all your GCP resources (compute, storage, databases etc) **within a VPC**
 - Secure resources from unauthorized access AND
 - Enable secure communication between your cloud resources
- VPC is a **global resource** & contains subnets in one or more region
 - **(REMEMBER)** NOT tied to a region or a zone. VPC resources can be in any region or zone!



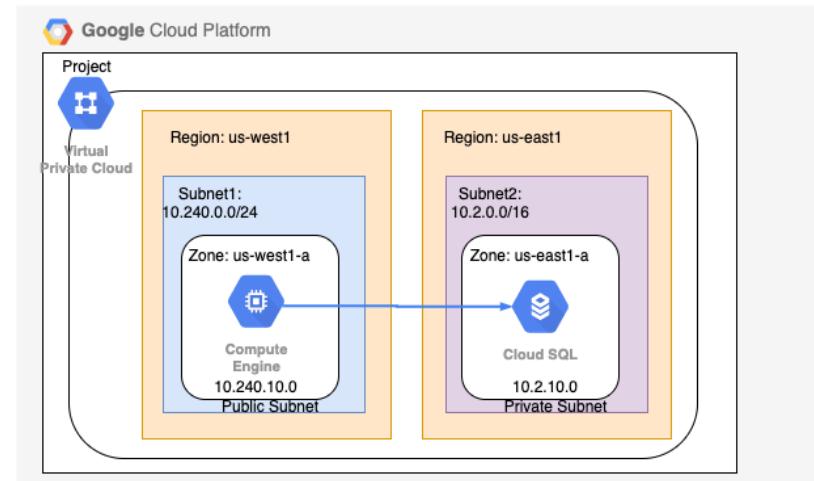
Need for VPC Subnets

- Different types of resources are created on cloud - databases, compute etc
 - Each type of resource has **its own access needs**
 - Load Balancers are accessible from internet (**public resources**)
 - Databases or VM instances should NOT be accessible from internet
 - ONLY applications within your network (VPC) should be able to access them(**private resources**)
- How do you **separate public resources from private resources** inside a VPC?
 - Create separate Subnets!
- (Additional Reason) You want to distribute resources across multiple regions for high availability



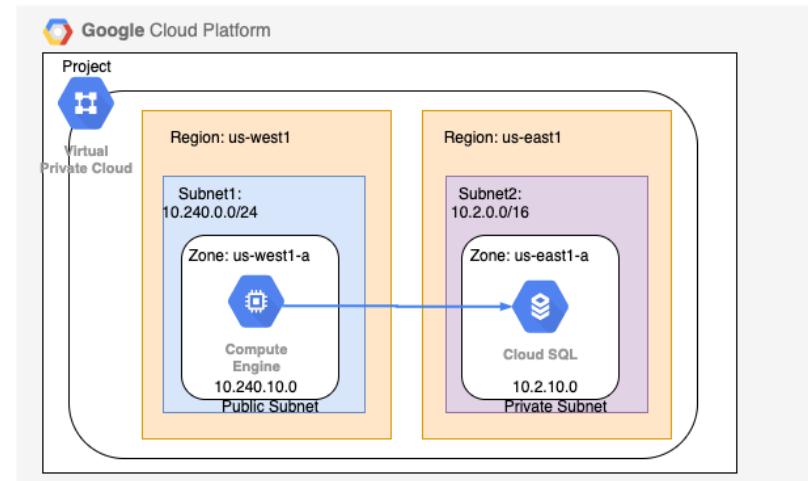
VPC Subnets

- (Solution) Create different subnets for public and private resources
 - Resources in a public subnet **CAN** be accessed from internet
 - Resources in a private subnet **CANNOT** be accessed from internet
 - BUT resources in public subnet can talk to resources in private subnet
- Each Subnet is created in a region
- **Example :** VPC - demo-vpc => Subnets - region us-central1, europe-west1 or us-west1 or ..



Creating VPCs and Subnets

- By default, every project has a default VPC
- You can create YOUR own VPCs:
 - **OPTION 1:** Auto mode VPC network:
 - Subnets are automatically created in each region
 - Default VPC created automatically in the project uses auto mode!
 - **OPTION 2:** Custom mode VPC network:
 - No subnets are automatically created
 - You have complete control over subnets and their IP ranges
 - Recommended for Production
- Options when you create a subnet:
 - **Enable Private Google Access** - Allows VM's to connect to Google API's using private IP's
 - **Enable FlowLogs** - To troubleshoot any VPC related network issues



CIDR (Classless Inter-Domain Routing) Blocks

- Resources in a network use continuous IP addresses to make routing easy:
 - Example: Resources inside a specific network can use IP addresses from 69.208.0.0 to 69.208.0.15
- How do you express a **range of addresses** that resources in a network can have?
 - CIDR block
- A **CIDR block** consists of a **starting IP address(69.208.0.0)** and a **range(/28)**
 - Example: CIDR block 69.208.0.0/28 represents addresses from 69.208.0.0 to 69.208.0.15 - a total of 16 addresses
- **Quick Tip:** 69.208.0.0/28 indicates that the first 28 bits (out of 32) are fixed.
 - Last 4 bits can change => $2^{24} = 16$ addresses

CIDR Exercises

CIDR	Start Range	End Range	Total addresses	Bits selected in IP address
69.208.0.0/24	69.208.0.0	69.208.0.255	256	01000101.11010000.00000000.*****
69.208.0.0/25	69.208.0.0	69.208.0.127	128	01000101.11010000.00000000.0*****
69.208.0.0/26	69.208.0.0	69.208.0.63	64	01000101.11010000.00000000.00*****
69.208.0.0/27	69.208.0.0	69.208.0.31	32	01000101.11010000.00000000.000****
69.208.0.0/28	69.208.0.0	69.208.0.15	16	01000101.11010000.00000000.0000***
69.208.0.0/29	69.208.0.0	69.208.0.7	8	01000101.11010000.00000000.00000**
69.208.0.0/30	69.208.0.0	69.208.0.3	4	01000101.11010000.00000000.00000**
69.208.0.0/31	69.208.0.0	69.208.0.1	2	01000101.11010000.00000000.000000*
69.208.0.0/32	69.208.0.0	69.208.0.0	1	01000101.11010000.00000000.00000000

- Exercise : How many addresses does **69.208.0.0/26** represent?
 - 2 to the power ($32-26 = 6$) = 64 addresses from 69.208.0.0 to 69.208.0.63
- Exercise : How many addresses does **69.208.0.0/30** represent?
 - 2 to the power ($32-30 = 2$) = 4 addresses from 69.208.0.0 to 69.208.0.3
- Exercise : What is the difference between **0.0.0.0/0** and **0.0.0.0/32**?
 - 0.0.0.0/0 represent all IP addresses. 0.0.0.0/32 represents just one IP address 0.0.0.0.

Examples of Recommended CIDR Blocks - VPC Subnets

- **Recommended CIDR Blocks**
 - Private IP addresses RFC 1918: 10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16
 - Shared address space RFC 6598: 100.64.0.0/10
 - IETF protocol assignments RFC 6890: 192.0.0.0/24
- **Restricted Range Examples**
 - You CANNOT use these as CIDR for VPC Subnets
 - Private Google Access-specific virtual IP addresses: 199.36.153.4/30, 199.36.153.8/30
 - Current (local) network RFC 1122: 0.0.0.0/8
 - Local host RFC 1122: 127.0.0.0/8
- **(REMEMBER)** You CAN EXTEND the CIDR Block Range of a Subnet (Secondary CIDR Block)



Virtual Private
Cloud

Firewall Rules

- Configure Firewall Rules to control traffic going in or out of the network:
 - Stateful
 - Each firewall rule has priority (0-65535) assigned to it
 - 0 has highest priority. 65535 has least priority
 - Default implied rule with lowest priority (65535)
 - Allow all egress
 - Deny all ingress
 - Default rules can't be deleted
 - You can override default rules by defining new rules with priority 0-65534
 - Default VPC has 4 additional rules with priority 65534
 - Allow incoming traffic from VM instances in same network (**default-allow-internal**)
 - Allow Incoming TCP traffic on port 22 (SSH) **default-allow-ssh**
 - Allow Incoming TCP traffic on port 3389 (RDP) **default-allow-rdp**
 - Allow Incoming ICMP from any source on the network **default-allow-icmp**



Virtual Private
Cloud

Firewall Rules - Ingress and Egress Rules

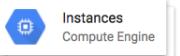
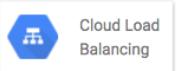
- **Ingress Rules:** Incoming traffic from outside to GCP targets
 - **Target (defines the destination):** All instances or instances with TAG/SA
 - **Source (defines where the traffic is coming from):** CIDR or All instances or instances with TAG/SA
- **Egress Rules:** Outgoing traffic to destination from GCP targets
 - **Target (defines the source):** All instances or instances with TAG/SA
 - **Destination:** CIDR Block
- **Along with each rule,** you can also define:
 - **Priority** - Lower the number, higher the priority
 - **Action on match** - Allow or Deny traffic
 - **Protocol** - ex. TCP or UDP or ICMP
 - **Port** - Which port?
 - **Enforcement status** - Enable or Disable the rule



Virtual Private
Cloud

Firewall Rules - Best Practices

- Use **network tags** and control allowed traffic into a VM using firewall rules
- Ensure that firewall rule allow the right kind of traffic:
 - Only allow traffic from load balancing into VM instances
 - Remove **0.0.0.0/0** from Source IP ranges
 - Add **130.211.0.0/22** and **35.191.0.0/16**
 - Allows health checks from load balancing to VM instances
- **(REMEMBER)** All egress from an VM instance is allowed by default:
 - To allow Specific EGRESS ONLY
 - 1: Create an egress rule with low priority to deny all traffic
 - 2: Create egress rule with high priority to allow traffic on specific port



Shared VPC

- Scenario: Your organization has multiple projects. You want resources in different projects to talk to each other?
 - How to allow resources in different projects to talk with internal IPs securely and efficiently?
- Enter **Shared VPC**
 - Created at organization or shared folder level (Access Needed: Shared VPC Admin)
 - Allows VPC network to be shared between projects in same organization
 - Shared VPC contains one host project and multiple service projects:
 - **Host Project** - Contains shared VPC network
 - **Service Projects** - Attached to host projects
- Helps you achieve **separation of concerns**:
 - Network administrators responsible for Host projects and Resource users use Service Project



Virtual Private
Cloud

VPC Peering

- Scenario: How to connect VPC networks across different organizations?
- Enter **VPC Peering**
 - Networks in same project, different projects and across projects in different organizations can be peered
 - All communication happens using internal IP addresses
 - Highly efficient because all communication happens **inside Google network**
 - Highly secure because **not accessible from Internet**
 - **No data transfer charges** for data transfer between services
 - (REMEMBER) Network administration is NOT changed:
 - Admin of one VPC do not get the role automatically in a peered network

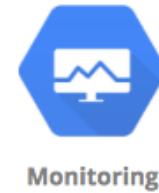


Virtual Private
Cloud

Cloud Operations

Cloud Monitoring

- To operate cloud applications effectively, you should know:
 - Is my application healthy?
 - Are the users experiencing any issues?
 - Does my database has enough space?
 - Are my servers running in an optimum capacity?
- **Cloud Monitoring** - Tools to monitor your infrastructure
 - Measures key aspects of services (Metrics)
 - Create visualizations (Graphs and Dashboard)
 - Configure Alerts (when metrics are NOT healthy)
 - Define Alerting Policies:
 - Condition
 - Notifications - Multiple channels
 - Documentation



Cloud Monitoring - Workspace

- You can use Cloud Monitoring to monitor one or more GCP projects and one or more AWS accounts
- How do you group all the information from multiple GCP projects or AWS Accounts?
- **Create a Workspace**
- Workspaces are needed to organize monitoring information
 - A workspace allows you to see monitoring information from multiple projects
 - Step I: Create workspace in a specific project (Host Project)
 - Step II: Add other GCP projects (or AWS accounts) to the workspace



Monitoring

Cloud Monitoring - Virtual Machines

In 28
Minutes



- **Default metrics monitored** include:
 - CPU utilization
 - Some disk traffic metrics
 - Network traffic, and
 - Uptime information
- Install **Cloud Monitoring agent** on the VM to get more disk, CPU, network, and process metrics:
 - collectd-based daemon
 - Gathers metrics from VM and sends them to Cloud Monitoring

Cloud Logging

- Real time log management and analysis tool
- Allows to store, search, analyze and alert on massive volume of data
- Exabyte scale, fully managed service
 - No server provisioning, patching etc
- Ingest Log data from any source
- Key Features:
 - Logs Explorer - Search, sort & analyze using flexible queries
 - Logs Dashboard - Rich visualization
 - Logs Metrics - Capture metrics from logs (using queries/matching strings)
 - Logs Router - Route different log entries to different destinations



Logging

Cloud Logging - Collection

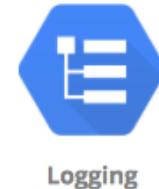
In 28
Minutes



- Most GCP Managed services automatically send logs to Cloud Logging:
 - GKE
 - App Engine
 - Cloud Run
- Ingest logs from GCE VMs:
 - Install **Logging Agent** (based on fluentd)
 - (Recommended) Run Logging Agent on all VM instances
- Ingest logs from on-premises:
 - (Recommended) Use the BindPlane tool from Blue Medora
 - Use the Cloud Logging API

Cloud Logging - Audit and Security Logs

- **Access Transparency Log:** Captures Actions performed by GCP team on your content (NOT supported by all services):
 - ONLY for organizations with Gold support level & above
- **Cloud Audit Logs:** Answers who did what, when and where:
 - Admin activity Logs
 - Data Access Logs
 - System Event Audit Logs
 - Policy Denied Audit Logs



Cloud Logging - Audit Logs

- Which service?
 - protoPayload.serviceName
- Which operation?
 - protoPayload.methodName
- What resource is audited?
 - resource.Type
- Who is making the call?
 - authenticationInfo.principalEmail

```
{  
  protoPayload: {  
    @type: "type.googleapis.com/google.cloud.audit.AuditLog",  
    status: {},  
    authenticationInfo: {  
      principalEmail: "user@example.com"  
    },  
    serviceName: "appengine.googleapis.com",  
    methodName: "SetIamPolicy",  
    authorizationInfo: [...],  
    serviceData: {  
      @type: "type.googleapis.com/google.appengine.legacy.AuditData",  
      policyDelta: { bindingDeltas: [  
        action: "ADD",  
        role: "roles/logging.privateLogViewer",  
        member: "user:user@example.com"  
      ], }  
    },  
    request: {  
      resource: "my-gcp-project-id",  
      policy: { bindings: [...] }  
    },  
    response: {  
      bindings: [  
        {  
          role: "roles/logging.privateLogViewer",  
          members: [ "user:user@example.com" ]  
        }  
      ]  
    },  
    insertId: "53179D9A9B559.AD6ACC7.B48604EF",  
    resource: {  
      type: "gae_app",  
      labels: { project_id: "my-gcp-project-id" }  
    },  
    timestamp: "2019-05-27T16:24:56.135Z",  
    severity: "NOTICE",  
    logName: "projects/my-gcp-project-id/logs/cloudaudit.googleapis.com%2Factivity",  
  }  
}
```

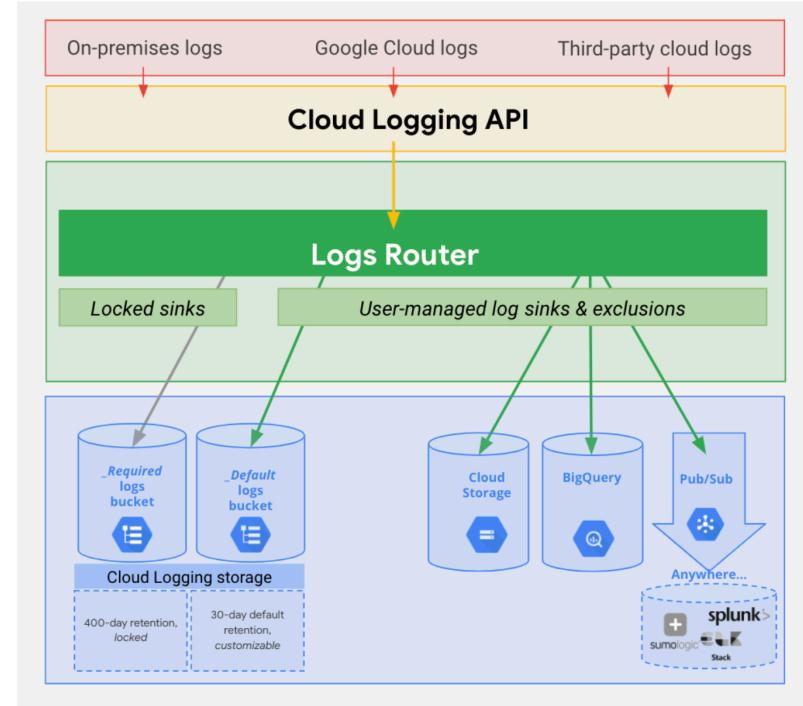
Cloud Audit Logs

In 28
Minutes

Feature	Admin Activity Logs	Data Access Logs	System Event Logs	Policy Denied Logs
Logs for	API calls or other actions that modify the configuration of resources	Reading configuration of resources	Google Cloud administrative actions	When user or service account is denied access
Default Enabled	✓	X	✓	✓
VM Examples	VM Creation, Patching resources, Change in IAM permissions	Listing resources (vms, images etc)	On host maintenance, Instance preemption, Automatic restart	Security policy violation logs
Cloud Storage	Modify bucket or object	Modify/Read bucket or object		
Recommended	Logging/Logs Viewer	Logging/Private	Logging/Logs Viewer	Logging/Logs

Cloud Logging - Controlling & Routing

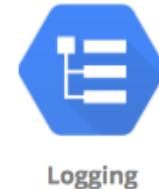
- How do you manage your logs?
 - Logs from various sources reaches **Log Router**
 - Log Router checks against configured rules
 - What to ingest? what to discard?
 - Where to route?
- Two types of Logs buckets:
 - **_Required:** Holds Admin activity, System Events & Access Transparency Logs (retained for 400 days)
 - ZERO charge
 - You cannot delete the bucket
 - You cannot change retention period
 - **_Default:** All other logs (retained for 30 days)
 - You are billed based on Cloud Logging pricing
 - You cannot delete the bucket:
 - But you can disable the **_Default** log sink route to disable ingestion!
 - You can edit retention settings (1 to 3650 days (10 years))



source: (<https://cloud.google.com>)

Cloud Logging - Export

- Logs are ideally stored in Cloud Logging for limited period
 - For long term retention (Compliance, Audit) logs can be exported to:
 - Cloud Storage bucket (ex: bucket/syslog/2025/05/05)
 - Big Query dataset (ex: tables syslog_20250505 > columns timestamp, log)
 - Cloud Pub/Sub topic (base64 encoded log entries)
- How do you export logs?
 - Create **sinks** to these destinations using Log Router:
 - You can create **include** or **exclude** filters to limit the logs

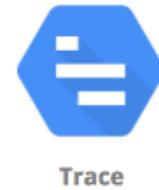


Cloud Logging - Export - Use Cases

- Use Case 1: Troubleshoot using VM Logs:
 - Install Cloud logging agent in all VM's and send logs to Cloud Logging
 - Search for logs in Cloud Logging
- Use Case 2: Export VM logs to BigQuery for querying using SQL like queries:
 - Install Cloud logging agent in all VM's and send logs to Cloud Logging
 - Create a BigQuery dataset for storing the logs
 - Create an export sink in Cloud Logging with BigQuery dataset as sink destination
- Use Case 3: You want to retain audit logs for external auditors at minimum cost
 - Create an export sink in Cloud Logging with Cloud Storage bucket as sink destination
 - Provide auditors with Storage Object Viewer role on the bucket
 - You can use Google Data Studio also (for visualization)

Cloud Trace

- Distributed tracing system for GCP: Collect latency data from:
 - Supported Google Cloud Services
 - Instrumented applications (using tracing libraries) using **Cloud Trace API**
- Find out:
 - How long does a service take to handle requests?
 - What is the average latency of requests?
 - How are we doing over time? (increasing/decreasing trend)
- Supported for:
 - Compute Engine, GKE, App Engine (Flexible/Standard) etc
- Trace client libraries available for:
 - C#, Go, Java, Node.js, PHP, Python & Ruby



Cloud Debugger

- How to debug issues that are happening only in test or production environments?
- **Cloud Debugger:** Capture state of a running application
 - Inspect the state of the application directly in the GCP environment
 - Take snapshots of variables and call stack
 - No need to add logging statements
 - No need to redeploy
 - Very lightweight => Very little impact to users
 - Can be used in any environment: Test, Acceptance, Production



Debugger

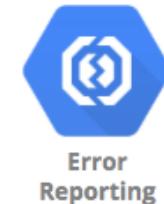
Cloud Profiler

- How do you identify performance bottlenecks in production?
- **Cloud Profiler** - Statistical, low-overhead profiler
 - Continuously gathers CPU and Memory usage from production systems
 - Connect profiling data with application source code
 - Easily identify performance bottlenecks
 - Two major components:
 - Profiling agent (collects profiling information)
 - Profiler interface (visualization)



Error Reporting

- How do you identify production problems in real time?
- Real-time exception monitoring:
 - Aggregates and displays errors reported from cloud services (using stack traces)
 - **Centralized Error Management console:**
 - Identify & manage top errors or recent errors
 - Use **Firebase Crash Reporting** for errors from Android & iOS client applications
 - Supported for Go, Java, .NET, Node.js, PHP, Python, and Ruby
- Errors can be reported by:
 - Sending them to Cloud Logging OR
 - By calling Error Reporting API
- Error Reporting can be accessed from desktop
 - Also available in the Cloud Console mobile app for iOS and Android



Cloud Logging, Monitoring .. - Stackdriver

Stackdriver Service	New Service Name
Stackdriver Monitoring	Cloud Monitoring
Stackdriver Logging	Cloud Logging
Stackdriver Error Reporting	Error Reporting
Stackdriver Trace	Cloud Trace
Stackdriver Profiler	Cloud Profiler

Cloud Operations Scenarios

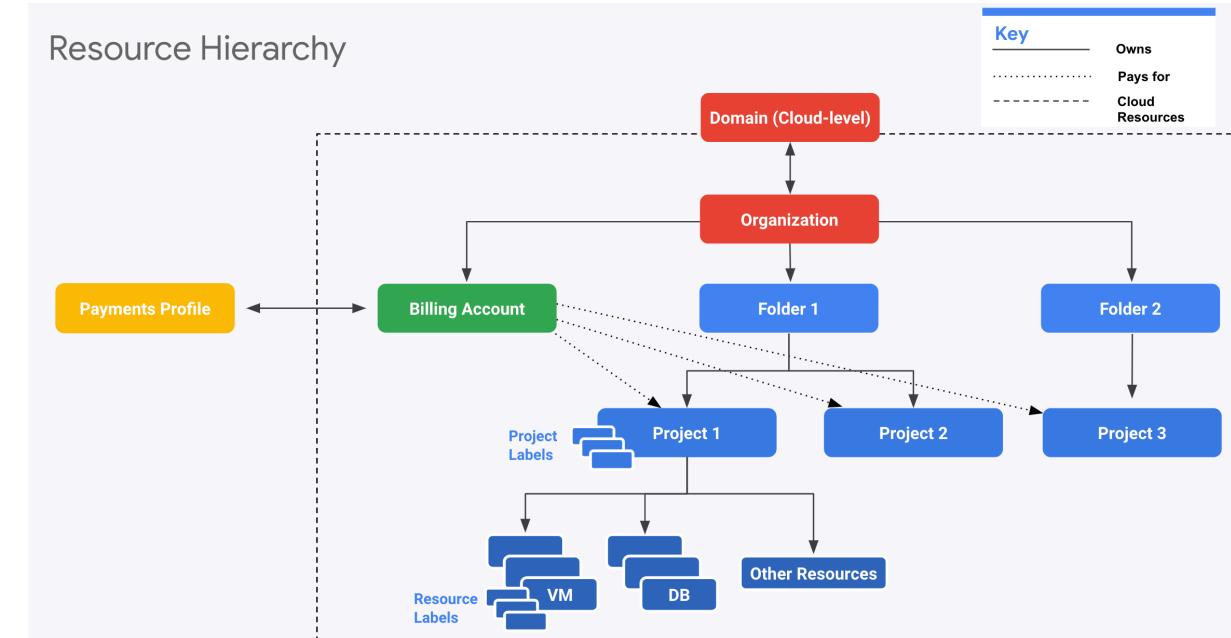
In 28
Minutes

Scenario	Solution
You would like to record all operations/requests on all objects in a bucket (for auditing)	Turn on data access audit logging for the bucket
You want to trace a request across multiple microservices	Cloud Trace
You want to identify prominent exceptions (or errors) for a specific microservice	Error Reporting
You want to debug a problem in production by executing step by step	Cloud Debugger
You want to look at the logs for a specific request	Cloud Logging

Organizing GCP Resources

Resource Hierarchy in GCP

- Well defined hierarchy:
 - Organization > Folder > Project > Resources
- Resources are created in projects
- A Folder can contain multiple projects
- Organization can contain multiple Folders



source: (<https://cloud.google.com>)

Resource Hierarchy - Recommendations for Enterprises

- **Create separate projects for different environments:**
 - Complete isolation between test and production environments
- **Create separate folders for each department:**
 - Isolate production applications of one department from another
 - We can create a shared folder for shared resources
- **One project per application per environment:**
 - Let's consider two apps: "A1" and "A2"
 - Let's assume we need two environments: "DEV" and "PROD"
 - In the ideal world you will create four projects: A1-DEV, A1-PROD, A2-DEV, A2-PROD:
 - Isolates environments from each other
 - DEV changes will NOT break PROD
 - Grant all developers complete access (create, delete, deploy) to DEV Projects
 - Provide production access to operations teams only!

Billing Accounts

- **Billing Account** is mandatory for creating resources in a project:
 - Billing Account contains the payment details
 - Every Project with active resources should be associated with a Billing Account
- Billing Account can be associated with one or more projects
- You can have multiple billing accounts in an Organization
- (RECOMMENDATION) Create Billing Accounts representing your organization structure:
 - A startup can have just one Billing account
 - A large enterprise can have a separate billing account for each department
- Two Types:
 - **Self Serve** : Billed directly to Credit Card or Bank Account
 - **Invoiced** : Generate invoices (Used by large enterprises)

Managing Billing - Budget, Alerts and Exports

- Setup a **Cloud Billing Budget** to avoid surprises:
 - (RECOMMENDED) Configure Alerts
 - Default alert thresholds set at 50%, 90% & 100%
 - Send alerts to Pub Sub (Optional)
 - Billing admins and Billing Account users are alerted by e-mail
- Billing data can be **exported (on a schedule)** to:
 - Big Query (if you want to query information or visualize it)
 - Cloud Storage (for history/archiving)

- **Principle of Least Privilege** - Give least possible privilege needed for a role!
 - Basic Roles are NOT recommended
 - Prefer predefined roles when possible
 - Use Service Accounts with minimum privileges
 - Use different Service Accounts for different apps/purposes
- **Separation of Duties** - Involve atleast 2 people in sensitive tasks:
 - Example: Have separate deployer and traffic migrator roles
 - AppEngine provides App Engine Deployer and App Engine Service Admin roles
 - App Engine Deployer can deploy new version but cannot shift traffic
 - App Engine Service Admin can shift traffic but cannot deploy new version!
- **Constant Monitoring:** Review Cloud Audit Logs to audit changes to IAM policies and access to Service Account keys
 - Archive Cloud Audit Logs in Cloud Storage buckets for long term retention
- **Use Groups when possible**
 - Makes it easy to manage users and permissions

User Identity Management in Google Cloud

- Email used to create free trial account => "**Super Admin**"
 - Access to everything in your GCP organization, folders and projects
 - Manage access to other users **using their Gmail accounts**
- However, this is **NOT recommended** for enterprises
- **Option 1:** Your Enterprise is using **Google Workspace**
 - Use Google Workspace to manage users (groups etc)
 - Link Google Cloud Organization with Google Workspace
- **Option 2:** Your Enterprise uses an Identity Provider of its own
 - **Federate** Google Cloud with your Identity Provider



Cloud IAM

Corporate Directory Federation

- Federate Cloud Identity or Google Workspace with your external identity provider (IdP) such as Active Directory or Azure Active Directory.
- Enable Single Sign On:
 - 1: Users are redirected to an external IdP to authenticate
 - 2: When users are authenticated, SAML assertion is sent to Google Sign-In
- Examples:
 - Federate Active Directory with Cloud Identity by using Google Cloud Directory Sync (GCDS) and Active Directory Federation Services (AD FS)
 - Federating Azure AD with Cloud Identity



Cloud IAM

IAM Members/Identities

- **Google Account** - Represents a person (an email address)
- **Service account** - Represents an application account (Not person)
- **Google group** - Collection - Google & Service Accounts
 - Has an unique email address
 - Helps to apply access policy to a group
- **Google Workspace domain:** Google Workspace (formerly G Suite) provides collaboration services for enterprises:
 - Tools like Gmail, Calendar, Meet, Chat, Drive, Docs etc are included
 - If your enterprise is using Google Workspace, you can manage permissions using your Google Workspace domain
- **Cloud Identity domain** - Cloud Identity is an Identity as a Service (IDaaS) solution that centrally manages users and groups.
 - You can use IAM to manage access to resources for each Cloud Identity account



IAM Members/Identities - Use Cases

In 28
Minutes

Scenario	Solution
All members in your team have G Suite accounts. You are creating a new production project and would want to provide access to your operations team	Create a Group with all your operations team. Provide access to production project to the Group.
All members in your team have G Suite accounts. You are setting up a new project. You want to provide a one time quick access to a team member.	Assign the necessary role directly to G Suite email address of your team member If it is not a one time quick access, the recommended approach would be to create a Group
You want to provide an external auditor access to view all resources in your project BUT he should NOT be able to make any changes	Give them roles/viewer role (Generally basic roles are NOT recommended BUT it is the simplest way to provide view only access to all resources!)
Your application deployed on a GCE VM (Project A) needs to access cloud storage bucket from a different project (Project B)	In Project B, assign the right role to GCE VM service account from Project A

Organization Policy Service

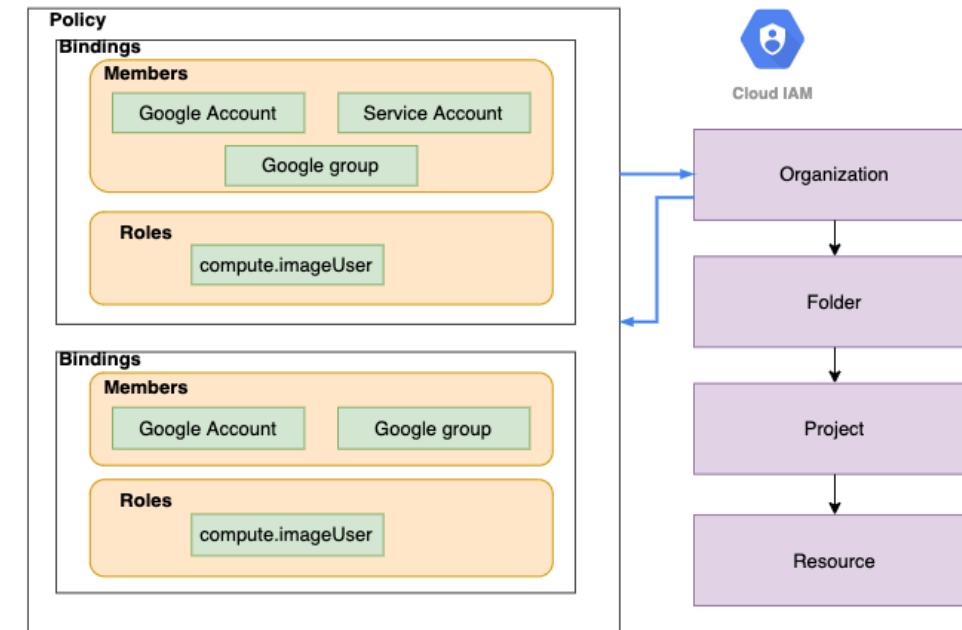
- How to enable **centralized constraints** on all resources created in an Organization?
 - Configure Organization Policy
 - Example: Disable creation of Service Accounts
 - Example: Allow/Deny creation of resources in specific regions
- Needs a Role - Organization Policy Administrator
- (Remember) IAM focuses on **Who**
 - Who can take specific actions on resources?
- (Remember) Organization Policy focuses on **What**
 - What can be done on specific resources?



Cloud IAM

Resource Hierarchy & IAM Policy

- IAM Policy can be set at any level of the hierarchy
- Resources inherit the policies of **All parents**
- The effective policy for a resource is the union of the policy on that resource and its parents
- Policy inheritance is transitive:
 - For example: Organization policies are applied at resource level
- You can't restrict policy at lower level if permission is given at an higher level



Cloud BigQuery Roles

- Cloud BigQuery IAM Roles
 - **BigQuery Admin** - `bigrquery.*`
 - **BigQuery Data Owner** - `bigrquery.datasets.*`, `bigrquery.models.*`, `bigrquery.routines.*`,
`bigrquery.tables.*` (**Does NOT have access to Jobs!**)
 - **BigQuery Data Editor** - `bigrquery.tables.(create/delete/export/get/getData/getIamPolicy/`
`list/update/updateData/updateTag)`, `bigrquery.models.*`, `bigrquery.routines.*`,
`bigrquery.datasets.(create/get/getIamPolicy/updateTag)`
 - **BigQuery Data Viewer** - `get/list bigrquery.(datasets/models/routines/tables)`
 - **BigQuery Job User** - `bigrquery.jobs.create`
 - **BigQuery User** - BigQuery Data Viewer + `get/list (jobs, capacityCommitments, reservations`
`etc)`
- To see data, you need either BigQuery User or BigQuery Data Viewer roles
 - You CANNOT see data with BigQuery Job User roles
- BigQuerv Data Owner or Data Viewer roles do NOT have access to iobs!

Corporate Directory Federation

- Federate Cloud Identity or Google Workspace with your external identity provider(IdP) such as Active Directory or Azure AD
- Enable Single Sign On:
 - 1: Users are redirected to an external IdP to authenticate
 - 2: When users are authenticated, SAML assertion is sent to Google Sign-In
- Examples:
 - Use Identity as a service (IDaaS) such as Okta as IdP (identity provider)
 - Use IDaaS (like Okta) for single sign-on
 - Use Active Directory as IdP
 - Use Active Directory Federation Services (AD FS) for single sign-on
 - Use Google Cloud Directory Sync to synchronize users and groups from Active Directory to Cloud Identity
 - Use Azure AD as IdP
 - Use Azure AD for single sign-on
 - Azure AD can be integrated with on-premises Active Directory



Exploring GCE VMs Further

SSHing into Linux VMs - Options

- Compute Engine Linux VMs uses key-based SSH authentication
- Two Options:
 - Metadata managed: Manually create and configure individual SSH keys
 - OS Login: Manage SSH access without managing individual SSH keys!
 - Recommended for managing multiple users across instances or projects
 - Your Linux user account is linked to your Google identity
 - To enable: Set enable-oslogin to true in metadata
 - *gcloud compute project-info/instances add-metadata --metadata enable-oslogin=TRUE*
 - (Advantage) Ability to import existing Linux accounts from on premises AD and LDAP
 - Users need to have roles : roles/compute.osLogin or roles/compute.osAdminLogin
- (Windows) Windows instances use password
authentication(username and password)
 - Generate using console or gcloud (*gcloud compute reset-windows-password*)



SSHing into Linux VMs - Details

- **Option 1:** Console - SSH Button
 - Ephemeral SSH key pair is created by Compute Engine
- **Option 2:** Gcloud - *gcloud compute ssh*
 - A username and persistent SSH key pair are created by Compute Engine
 - SSH key pair reused for future interactions
- **Option 3:** Use customized SSH keys
 - (Metadata managed): Upload the public key to project metadata OR
 - (OS Login): Upload your public SSH key to your OS Login profile
 - *gcloud compute os-login ssh-keys add* OR
 - Use OS Login API : POST https://oslogin.googleapis.com/v1/users/ACCOUNT_EMAIL:importSshPublicKey
- You can disable Project wide SSH keys on a specific compute instance
 - *gcloud compute instances add-metadata [INSTANCE_NAME] --metadata block-project-ssh-keys=TRUE*



Executing Shutdown Script on a GCE VM

- Execute commands before a GCE VM is stopped, terminated or restarted
 - Perform cleanup or export of logs
 - Applicable for Preemptible and Non Preemptible GCE VMs
- Very similar to startup script
 - Run as:
 - root user in Linux VMs
 - System account in Windows VMs
 - Stored as metadata
 - --metadata-from-file shutdown-script=script.sh
 - You can store startup and shutdown scripts in cloud storage
 - --metadata shutdown-script-url=gs://bucket-in-cloud-storage/file
- Run on best-effort basis
 - Example: WON'T run if you use hard reset (instances.reset)
 - Example: WON'T run if you exceed grace period for preemptible instances



Startup Script Example

Deploy new application version to Compute Engine

```
# Setup logging agent
curl -sS0 https://dl.google.com/cloudagents/install-logging-agent.sh
sudo bash install-logging-agent.sh

# Install GIT
apt-get update
apt-get install -yq git

# Clone Repo
git clone https://github.com/in28minutes/your-app.git /opt/app

# Build and Run app
//...
```

Troubleshooting VM startup and ...

- **Check 1:** Are there Quota errors?
- **Check 2:** Is boot disk full?
- **Check 3: Check serial port output**
 - Each VM instance has 4 virtual serial ports
 - Serial Port Output: OS, BIOS, and other system-level entities write output to serial ports
 - Useful for troubleshooting crashes, failed boots, startup, or shutdown issues
 - Accessible from Cloud Console, the gcloud tool, and the Compute Engine API
 - You can send serial port output to Cloud Logging:
 - `gcloud compute project-info add-metadata --metadata serial-port-logging-enable=true`
 - Interactive access to the serial console allows you to login and debug boot issues (without needing a full boot up)
 - `gcloud compute instances get-serial-port-output`
- **Check 4: Does your disk have a valid file system?**
 - Attach boot disk as a data disk to another VM and check the file system



Moving VM instances between Zones and Regions

- VM instances can be **moved between zones in the same region:**
 - `gcloud compute instances move my-instance --zone us-central1-a --destination-zone us-central1-b`
- **Restrictions:** You cannot use move command for:
 - Instances that are part of a MIG
 - Instances attached with local SSDs
 - Instances in TERMINATED status
 - Moving instances across regions
 - **Go for a Manual Approach:**
 - Create snapshots of attached persistent disks
 - `gcloud compute disks snapshot my-disk-a --snapshot-names my-pd-snapshot --zone ZONE`
 - Create copies of persistent disks in destination zone (of new region)
 - `gcloud compute disks create my-disk-b --source-snapshot my-pd-snapshot --zone ZONE`
 - Create new instance in destination zone (of new region) and attach the persistent disks



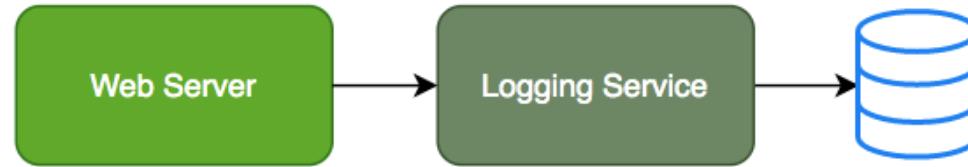
Decoupling Applications with Pub/Sub

Need for Asynchronous Communication

- Why do we need asynchronous communication?

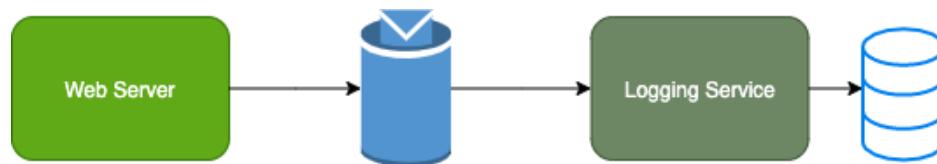
Synchronous Communication

In 28
Minutes



- Applications on your web server make synchronous calls to the logging service
- What if your logging service goes down?
 - Will your applications go down too?
- What if all of sudden, there is high load and there are lots of logs coming in?
 - Log Service is not able to handle the load and goes down very often

Asynchronous Communication - Decoupled



- Create a topic and have your applications put log messages on the topic
- Logging service picks them up for processing when ready
- Advantages:
 - Decoupling: Publisher (Apps) don't care about who is listening
 - Availability: Publisher (Apps) up even if a subscriber (Logging Service) is down
 - Scalability: Scale consumer instances (Logging Service) under high load
 - Durability: Message is not lost even if subscriber (Logging Service) is down

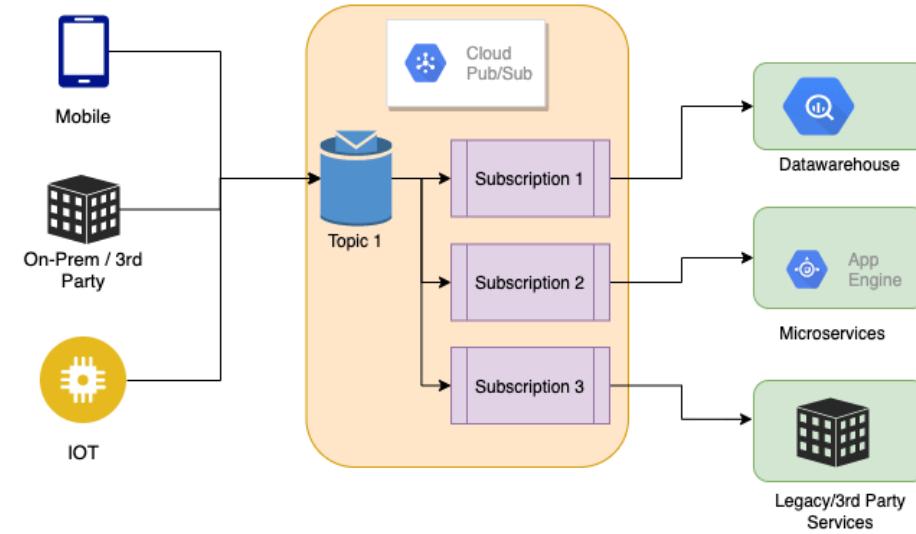
Pub/Sub

- Reliable, scalable, fully-managed asynchronous messaging service
- Backbone for **Highly Available** and **Highly Scalable** Solutions
 - Auto scale to process billions of messages per day
 - Low cost (Pay for use)
- Use cases: Event ingestion and delivery for streaming analytics pipelines
- Supports push and pull message deliveries

Cloud
Pub/Sub

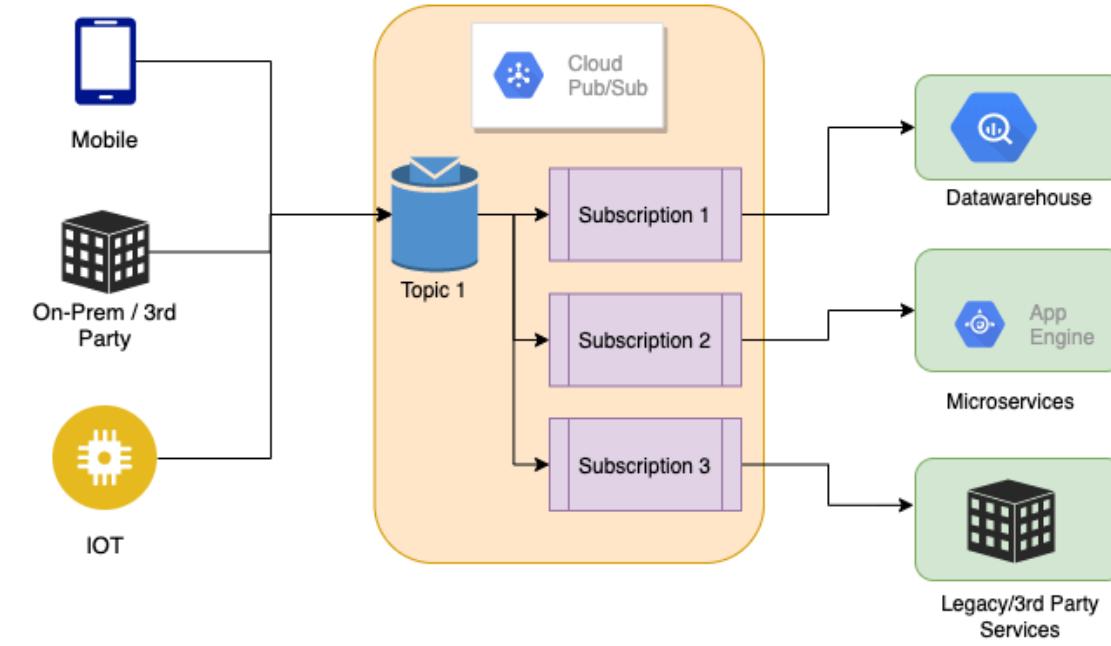
Pub/Sub - How does it work?

- **Publisher** - Sender of a message
 - Publishers send messages by making HTTPS requests to pubsub.googleapis.com
- **Subscriber** - Receiver of the message
 - **Pull** - Subscriber pulls messages when ready
 - Subscriber makes HTTPS requests to pubsub.googleapis.com
 - **Push** - Messages are sent to subscribers
 - Subscribers provide a web hook endpoint at the time of registration
 - When a message is received on the topic, A HTTPS POST request is sent to the web hook endpoints
- **Very Flexible** Publisher(s) and Subscriber(s) Relationships: One to Many, Many to One, Many to Many



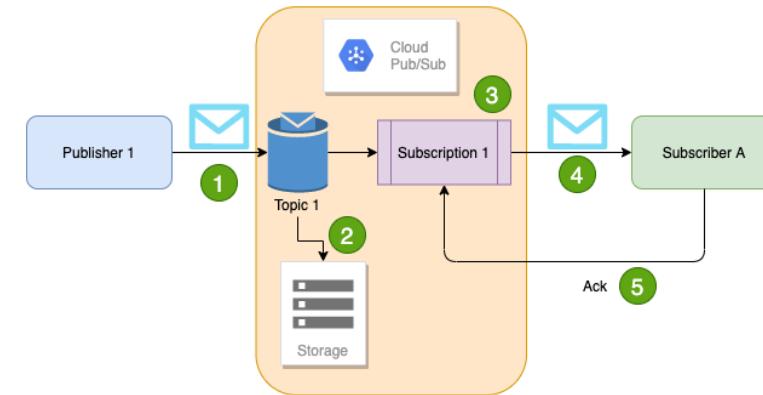
Pub/Sub - Getting Ready with Topic and Subscriptions

- Step 1 : Topic is created
- Step 2 : Subscription(s) are created
 - Subscribers register to the topic
 - Each Subscription represents discrete pull of messages from a topic:
 - Multiple clients pull same subscription => messages split between clients
 - Multiple clients create a subscription each => each client will get every message



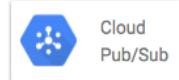
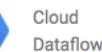
Pub/Sub - Sending and Receiving a Message

- Publisher sends a message to Topic
- Message **individually** delivered to each and every subscription
 - Subscribers can receive message either by:
 - Push: Pub/Sub sends the message to Subscriber
 - Pull: Subscribers poll for messages
- Subscribers send acknowledgement(s)
- Message(s) are removed from subscriptions message queue
 - Pub/Sub ensures the message is retained per subscription until it is acknowledged



Understanding Cloud Pub/Sub Best Practices

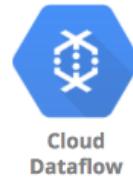
- **Usecases:**
 - Convert synchronous to asynchronous workflows
 - Also useful when consumer is unable to keep up with the producer (buffer data)
 - Alternatives: RabbitMQ, Apache Kafka
 - Apply transformations to IOT data stream
- Some use cases need in-order, exactly-once processing (deduplication) for messages
 - Pub Sub supports in order processing:
 - Option --enable-message-ordering on subscription
 - Add Dataflow into flow to enable message deduplication (exactly-once processing)
 - Maintains a list of message IDs for a time period
 - If a message ID repeats, it is discarded (assumed to be a duplicate)
 - Often sits between data ingestion services (Cloud Pub/Sub, Cloud IOT core ..) and storage/analysis services (Bigtable, BigQuery ..)

Cloud
Pub/SubCloud
Dataflow

Cloud Dataflow

In 28
Minutes

- Cloud Dataflow is a difficult service to describe:
 - Let's look at a few example pipelines you can build:
 - Pub/Sub > Dataflow > BigQuery (Streaming)
 - Pub/Sub > Dataflow > Cloud Storage (Streaming - files)
 - Cloud Storage > Dataflow > Bigtable/CloudSpanner/Datastore/BigQuery (Batch - Load data into databases)
 - Bulk compress files in Cloud Storage (Batch)
 - Convert file formats between Avro, Parquet & csv (Batch)
- Streaming and Batch Usecases
 - Realtime Fraud Detection, Sensor Data Processing, Log Data Processing, Batch Processing (Load data, convert formats etc)
- Use pre-built templates
- Based on Apache Beam (supports Java, Python, Go ...)
- Serverless (and Autoscaling)



Hybrid Cloud

Cloud VPN

In 28
Minutes

- **Cloud VPN** - Connect on-premise to GCP network over internet
 - Implemented using **IPSec VPN Tunnel**
 - Traffic through internet (public)
 - Traffic encrypted using **Internet Key Exchange** protocol
- Two types of Cloud VPN solutions:
 - **HA VPN** (SLA of 99.99% service availability with two external IP addresses)
 - Only dynamic routing (BGP) supported
 - **Classic VPN** (SLA of 99.9% service availability, a single external IP address)
 - Supports Static routing (policy-based, route-based) and dynamic routing using BGP
- **Easy to establish:** Does NOT need carrier circuits or contracts
- **Go for Cloud VPN if:**
 - You want the network to encrypt traffic OR
 - You want a lower throughput, low cost solution OR
 - You are experimenting with connectivity between cloud and on-premises



Cloud VPN

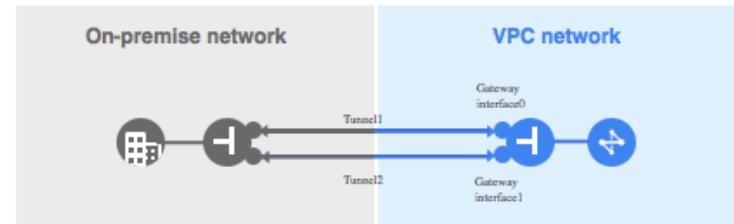
Cloud VPN - VPN Gateway, Peer Gateway and Cloud router

- **High-availability (HA) VPN**
 - High availability (99.99 SLA, within region)
 - Needs a Cloud HA VPN gateway
 - Regional resources with two interfaces
 - Connects to an on-premises VPN gateway (or peer gateway) through VPN tunnels
- **Classic VPN**
 - No high availability
 - Needs a Google Compute Engine VPN gateway
- **(REMEMBER) VPN gateway - Regional resource**
- **(REMEMBER) Cloud Router enables Dynamic Routing:** Enables Automatic route update when network topology changes

VPN options

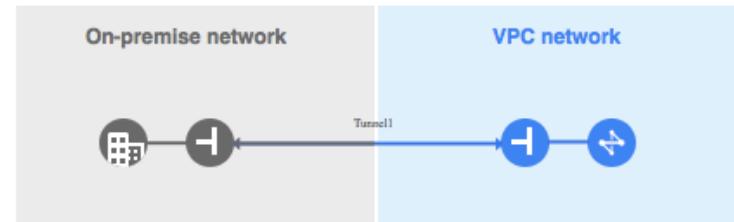
High-availability (HA) VPN

Supports dynamic routing (BGP) only
Supports high availability (99.99 SLA, within region)
[Learn more](#)



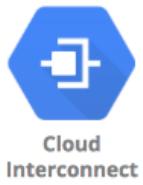
Classic VPN

Supports dynamic routing and static routing
No high availability
[Learn more](#)



Cloud Interconnect

- High speed, highly available, low-latency private connection into Google Cloud from your company's on-premises network
- **Dedicated Interconnect:** Ideal if you need high-bandwidth connection for large data transfers
 - Minimum private connection speed of 10Gbps (**OPTIONS: 10 Gbps or 100 Gbps**)
 - Go upto 8 x 10-Gbps (80 Gbps) circuits, or 2 x 100-Gbps (200 Gbps) circuits for each connection
 - Takes time to establish
- **Partner Interconnect:** Ideal if you need a private connection with lower bandwidth needs
 - 50Mbps to 10Gbps
- Data exchange happens through a private network:
 - Communicate using VPC network's internal IP addresses from on-premise network
 - Reduces egress costs



Hybrid Connectivity - Remember

- When you connect networks, ensure that resources on the networks **use different range of IP addresses!**
- **Always think:** What will we do if things go wrong?
 - Have a fallback option if the primary connection from on-premise to GCP fails
 - Dedicated Interconnect as primary
 - VPN as backup in case of failure
- Remember that there is a third hybrid connectivity option:
 - **Direct Peering:** Connect customer network to google network using network peering
 - Direct path from on-premises network to Google services
 - **Not a GCP Service**
 - Lower level network connection outside of GCP
 - **NOT RECOMMENDED:**
 - Use Cloud Interconnect and Cloud VPN



Cloud
Interconnect

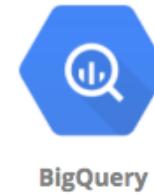


Cloud VPN

BigQuery Datawarehouse

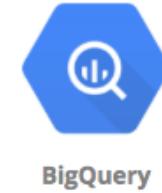
BigQuery - Datawarehouse

- **Exabyte scale modern Datawarehousing solution from GCP**
 - Relational database (SQL, schema, consistency etc)
 - Use SQL-like commands to query massive datasets
 - Traditional (Storage + Compute) + Modern (Realtime + Serverless)
- When we are talking about a Datawarehouse, **importing and exporting data (and formats) becomes very important:**
 - Load data from a **variety of sources**, incl. **streaming data**
 - Variety of import formats - CSV/JSON/Avro/Parquet/ORC/Datastore backup
 - Export to Cloud Storage (long term storage) & Data Studio (visualization)
 - Formats - CSV/JSON (with Gzip compression), Avro (with deflate or snappy compression)
- Automatically expire data (**Configurable Table Expiration**)
- Query **external data sources** without storing data in BigQuery
 - Cloud Storage, Cloud SQL, BigTable, Google Drive
 - Use **Permanent or Temporary** external tables



BigQuery - Accessing and Querying Data

- Access databases using:
 - Cloud Console
 - bq command-line tool (NOT gcloud)
 - BigQuery Rest API OR
 - HBase API based libraries (Java, .NET & Python)
- (Remember) BigQuery queries **can be expensive** as you are running them on large data sets!
- (BEST PRACTICE) **Estimate BigQuery queries before running:**
 - 1: Use UI(console)/bq(--dry-run) - Get scanned data volume (estimate)
 - 2: Use Pricing Calculator: Find price for scanning 1 MB data. Calculate cost.



Partitioning and Clustering BigQuery Tables - Use Case

- You pay for BigQuery queries by the amount of data scanned
- How do you reduce your costs of querying BigQuery and improve performance?
- Scenario: Imagine a Questions table with millions of rows
 - You want to find all questions asked between a date range (date between 2022-10-02 and 2028-10-02) belonging to a specific category
 - If you have a single questions table you need to scan all the rows
 - Partitioning - Divide table into multiple segments (example: by date)
 - Clustering - Group related data (example: by category)

Questions		
Date	Question	Category
2025-10-02	Question Detail ...	GCP
2025-10-02	Question Detail ...	AWS
2025-10-02	Question Detail ...	GCP
2025-10-03	Question Detail ...	Azure
2025-10-03	Question Detail ...	GCP
2025-10-03	Question Detail ...	Azure

Partitioning and Clustering BigQuery Tables

- **Partitioning:** Table is divided into segments
 - Makes it easy to manage and query your data
 - Improves performance and reduces costs
 - Partition based on Ingestion time (arrival time) OR a column (TIMESTAMP, DATE, or DATETIME , or INTEGER)
 - (DEFAULT) All partitions will share same schema as table
 - Allows you to expire (delete) parts of table data easily (partition_expiration_days)
- **Clustering:** Organize table data based on the contents of one or more columns
 - Goal: colocate related data and eliminate scans of unnecessary data
 - Avoid creating too many small partitions (of less than 1 GB). In those cases, prefer Clustering.

Questions_2025_10_02		
Date	Question	Category
2025-10-02	Question Detail	AWS
2025-10-02	Question Detail	GCP
2025-10-02	Question Detail	GCP

Questions_2025_10_03		
Date	Question	Category
2025-10-03	Question Detail	Azure
2025-10-03	Question Detail	Azure
2025-10-03	Question Detail	GCP

Partitioning and Clustering BigQuery Tables - Syntax

```
CREATE TABLE `my_data_set.questions_partitioned_and_clustered`  
...  
...  
PARTITIONED BY  
    DATE(created_date)  
    CLUSTER BY category  
...  
OPTIONS (  
    expiration_timestamp=TIMESTAMP "2025-01-01 00:00:00 UTC",  
    partition_expiration_days=7  
)
```

Expiring Data in BigQuery

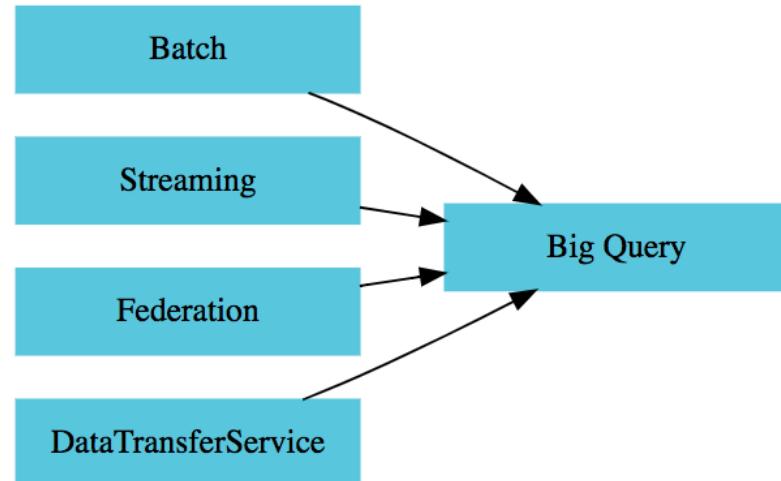
```
CREATE SCHEMA mydataset
OPTIONS(
    default_table_expiration_days=3.75
)

ALTER TABLE mydataset.mytable
SET OPTIONS (
    expiration_timestamp=TIMESTAMP "2025-01-01 00:00:00 UTC",
    partition_expiration_days=7
)
```

- You pay for data stored in BigQuery:
 - How can you automatically delete (expire) data which is not needed?
- Big Query Hierarchy : Data Set > Table > Partitions
 - You can configure expiration at each level
 - Configure default table expiration (default_table_expiration_days) for datasets
 - Configure expiration time (expiration_timestamp) for tables
 - Configure partition expiration (partition_expiration_days) for partitioned tables
- Best Practice: Expire Tables and Partitions you are NOT using!

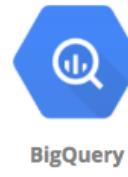
Importing Data into BigQuery

- **Batch Import (FREE):**
 - Import from Cloud Storage and local files
 - Import after processing by Cloud Dataflow and Cloud Dataproc
- **Streaming Import (\$\$\$\$):**
 - From Cloud Pub/Sub, Streaming Inserts
 - Import after processing by Cloud Dataflow and Cloud Dataproc
- **Federated Queries:** Query **external data**
 - Cloud Storage, Cloud SQL, BigTable, Google Drive
- **BigQuery Data Transfer Service:** Import from
 - Google SaaS apps (Google Ads, Cloud Storage etc)
 - External cloud storage providers - Amazon S3
 - Data warehouses - Teradata, Amazon Redshift



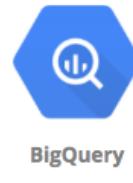
Streaming Data into BigQuery

- Loading data in bulk is free but streaming data is NOT FREE
 - AND there are a lot of limitations (Use with caution!)
- Streaming data can contain duplicates. How can you avoid duplicates?
 - Add **insertId** with each streaming insert:
 - **insertId** is used to provide best effort de-duplication (for up to one minute)
 - For strict de-duplication and transactions, try Google Cloud Datastore
- There are **strict streaming quotas** with BigQuery:
 - IF you are NOT populating **insertId**:
 - Maximum bytes per second - 1 GB per second, per project (REMEMBER per project - NOT per table)
 - ELSE (i.e. you are using **insertId**)
 - Maximum rows per second per project
 - US and EU multi-regions: 500,000, Other locations: 100,000
 - per table limitation: 100,000
 - Maximum bytes per second: 100 MB
 - If you have streaming of millions of rows per second, prefer **BisqTables**



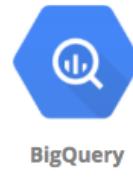
Understanding BigQuery Best Practices

- **Estimate your queries before running them**
 - bq --dry_run flag or dryRun API parameter
- Use clustering and partitioning for your tables
- **Avoid streaming inserts when possible**
 - Loading data in bulk is free but streaming data is NOT FREE
 - Offers Best effort de-duplication (when you use `insertId`)
 - Remember Quota limits
- **Expire Data Automatically:**
 - Configure default table expiration (`default_table_expiration_days`) for datasets
 - Configure expiration time for tables
 - Configure partition expiration for partitioned tables



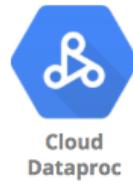
Understanding BigQuery Best Practices - 2

- Consider **Long-term storage** option
 - Long-term storage: Table in which data is NOT edited for 90 consecutive days
 - Lower Storage cost - Similar to Cloud Storage Nearline
- BigQuery is fast for **complex queries**:
 - BUT it is not as well optimized for narrow-range queries (Prefer Cloud Bigtable)
 - (REMEMBER) Too much complexity in setting up a query
- Optimize **BigQuery usage** using audit logs:
 - Analyze queries/jobs that were run earlier
 - Stream your audit logs (BigQueryAuditMetadata) to BigQuery
 - Understand usage patterns (query costs by user)
 - Optimize (visualize using Google Data Studio)



Cloud Dataproc

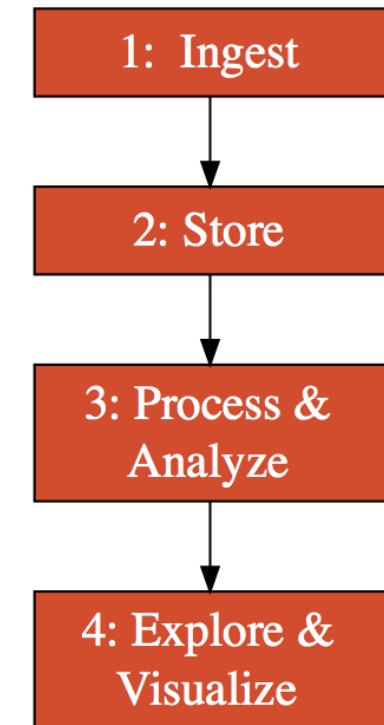
- Managed Spark and Hadoop service:
 - Variety of jobs are supported:
 - Spark, PySpark, SparkR, Hive, SparkSQL, Pig, Hadoop
 - Perform complex batch processing
- Multiple Cluster Modes:
 - Single Node / Standard/ High Availability (3 masters)
 - Use regular/preemptible VMs
- Use case: Move your Hadoop and Spark clusters to the cloud
 - Perform your machine learning and AI development using open source frameworks
- (REMEMBER) Cloud Dataproc is a data analysis platform
 - You can export cluster configuration but NOT data
- (ALTERNATIVE) BigQuery - When you run SQL queries on Petabytes
 - Go for Cloud Dataproc when you need more than queries (Example: Complex batch processing Machine Learning and AI workloads)



Designing Solutions Google Cloud Platform

Data Lifecycle

- **Four Steps:**
 - **Ingest:** Stream or Batch ingest
 - **Store:** Durably and cost efficiently store data in a convenient format
 - **Process and analyze:** Convert data to information (normalizations or aggregations)
 - **Explore and visualize:** Flexibility to play with data/information. Get and share insights.



Data Lifecycle - 1 - Ingest

- **Streaming:** Pub/Sub
- **Batch:** Storage Transfer Service, BigQuery Transfer Service, Transfer Appliance, gsutil etc
- **Database migration:** Migrate data from other sources to Google Cloud
 - Database Migration Service (Simplifying migrations to Cloud SQL)
 - Batch transfer to Cloud Storage
 - Load data into database from Cloud Storage using Dataflow

Cloud
Pub/SubCloud
Storage

Data Lifecycle - 2 - Store

Service	Solution
Cloud Storage	Object Storage (unstructured data)
Cloud SQL	Managed MySQL, PostgreSQL and MS SQL Server databases Relational, pre-defined schema, strong transactions, regional
Cloud Spanner	Horizontally scalable relational database Relational, pre-defined schema, strong transactions, high availability, and global scale
Cloud Firestore	Flexible, scalable, transactional NoSQL database
Cloud Bigtable	Managed wide-column NoSQL Petabyte scale, Real-time apps and large-scale analytical time-series workloads, single-row transactions
BigQuery	Managed data warehouse
Custom Database	Use Cloud Marketplace to deploy an open source database of your choice - MongoDB, Cassandra etc

Data Lifecycle - 3 - Process and analyze

In 28
Minutes

Raw Data > Actionable Information (Clean, Transform)

Service	Solution
Dataprep	Clean and prepare data Fully managed, No-Ops Usecases: Clean data on-boarded from external sources, Prepare data for ML Visual approach for non programmers
Cloud Data Loss Prevention	Scan, discover, classify, and report on data in Cloud Storage, BigQuery, and Datastore (mask, tokenize, and transform sensitive elements)
Dataflow	More flexible ETL pipelines (Fully managed, No-Ops, Batch and Streaming)
Dataproc	Complex processing using Spark and Hadoop Needs a cluster with compute engine VMs Usecases: Machine Learning, Migrate existing Spark and Hadoop workloads

Data Lifecycle - 4 - Explore and visualize

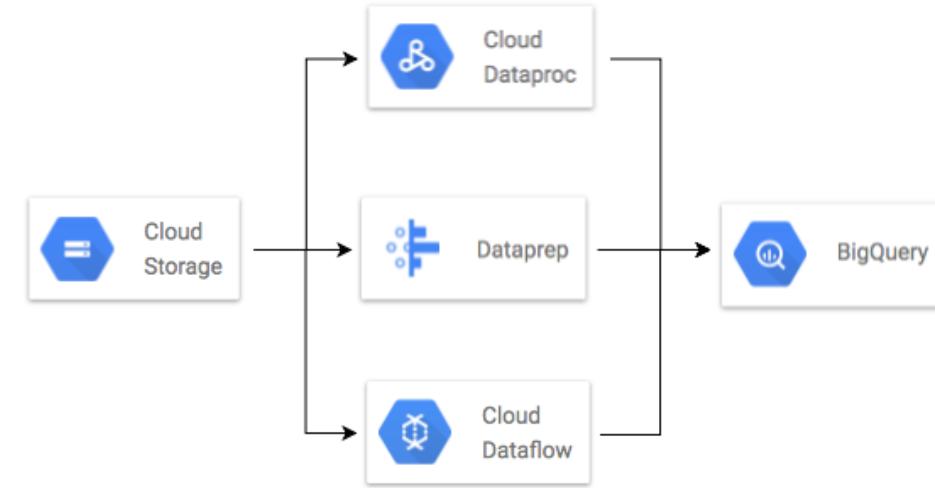
Service	Solution
Cloud BigQuery	Managed data warehouse Standard SQL, serverless, separate storage and compute
ML - Pre built models	Vision API, Speech-to-Text, Natural Language API, Video Intelligence API etc
ML - Custom models	Use AI Platform (based on TensorFlow) Use Dataflow for pre-processing
Cloud Datalab	Web based tool to explore, analyze and visualize data Based on Jupyter notebooks (Use Python, SQL queries etc) Support for popular data-science toolkits - pandas, numpy, and scikit-learn
Cloud Data Studio	Dashboarding and visualization Live charts and graphs based on data in Cloud SQL, BigQuery etc
Cloud Data Catalog	Data discovery and metadata management Unified view of all datasets Tag sensitive data using Cloud Data Loss Prevention (DLP)

Big Data & Analytics in GCP

Service	Solution
Pub/Sub	Foundation for stream analytics and event-driven systems
BigQuery	Serverless data warehouse to analyze petabytes of data Scale storage and compute separately
Google Data Studio	Managed visual analytics service
Dataflow	Data pipelines for (Stream + Batch) use cases
Dataproc	Managed Apache Spark and Apache Hadoop clusters
Dataprep	Clean and prepare data (structured and unstructured)
Datalab	Explore, analyze & visualize data on Jupyter notebooks (Use Python, SQL queries etc) Integrates well with BigQuery
Cloud Composer	Managed workflow orchestration service Create pipelines across clouds and on-premises data centers

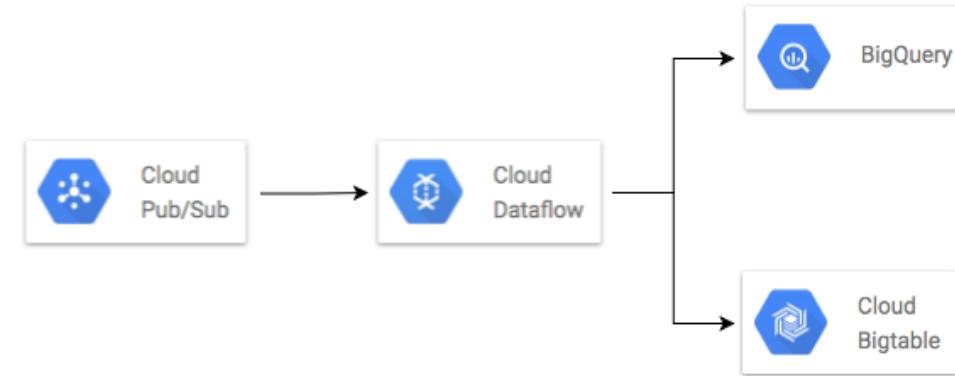
Big Data Flow - Batch Ingest into BigQuery

- Use extract, transform, and load (ETL) to load data into BigQuery
 - **Dataprep:** Clean and prepare data
 - **Dataflow:** Create data pipelines (and ETL)
 - **Dataproc:** Complex processing using Spark and Hadoop



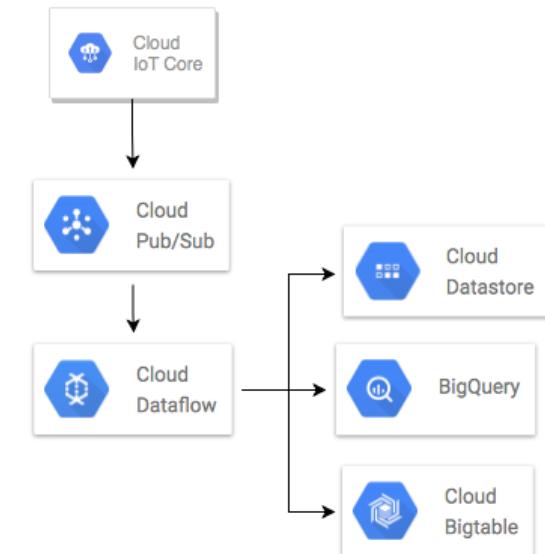
Streaming Workflow - Enable Realtime Querying

- **Query data in Realtime:**
 - **Pub/Sub and Dataflow:** Analyze, aggregate and filter data before storing to BigQuery
 - For pre-defined time series analytics, storing data in Bigtable gives you the ability to perform rapid analysis
 - For ad hoc complex analysis, prefer **BigQuery**



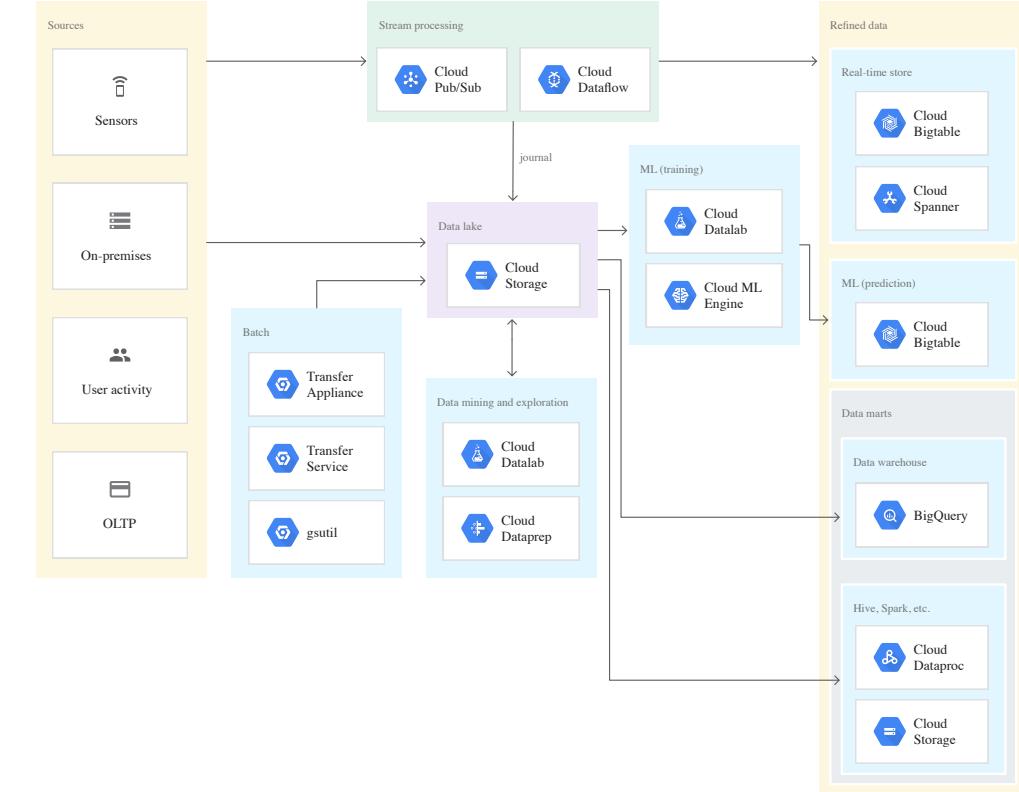
IOT

- **IoT Core:** Manage IoT (registration, authentication, and authorization) devices
 - Send/receive messages/real-time telemetry from/to IoT devices
- **Pub/Sub:** Durable message ingestion service (allows buffering)
- **Dataflow:** Processing data (ETL & more..)
 - Alternative: Use Cloud Functions to trigger alerts
- **Data Storage and Analytics:**
 - Make IOT data available to mobile or web apps => **Datastore**
 - Execute pre-defined time series queries => **Bigtable**
 - More complex or ad hoc analytics/analysis => **BigQuery**



Data Lake - Simplified Big Data Solutions

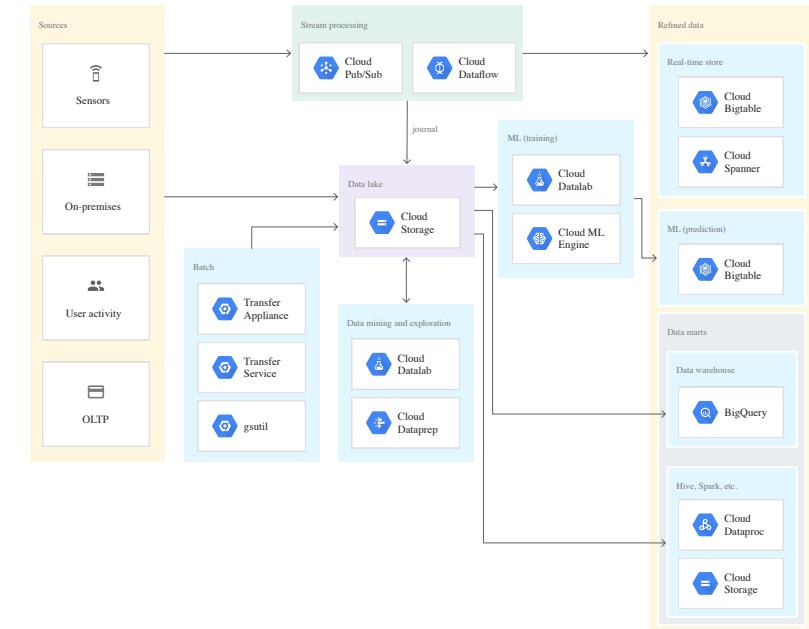
- Usual big data solutions are complex
- How can we make collecting, analyzing (reporting, analytics, machine learning) and visualizing huge data sets easy?
- How to design solutions that scale?
- How to build flexibility while saving cost?
- **Data Lake**
 - Single platform with combination of solutions for data storage, data management and data analytics



<https://cloud.google.com/solutions/build-a-data-lake-on-gcp>

GCP Data Lakes - Storage and Ingestion

- **Storage:** Cloud Storage (low cost + durability + performance + flexible processing)
- **Data Ingestion:**
 - Streaming data - Cloud Pub/Sub + Cloud Dataflow
 - Batch - Transfer Service + Transfer Appliance + gsutil
- **Processing and analytics:**
 - Run in-place querying using SQL queries using BigQuery or (Hive on Dataproc)
- **Data Mining and Exploration:**
 - Clean and transform raw data with Dataprep
 - Use Cloud Datalab (data science libraries such as TensorFlow and NumPy) for exploring

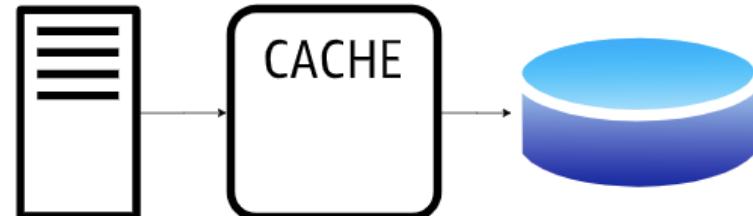


<https://cloud.google.com/solutions/build-a-data-lake-on-gcp>

Caching in Google Cloud Platform

Caching

- How can **reduce the load** on
 - Your data stores
 - Your servers
- Use Caching
- **Questions to ask when Caching?**
 - **How often** does the data change?
 - Caching is amazing if the data does not change frequently!
 - Am I OK with some **stale data**?
 - What should be **TTL** (Time to Live)?
- Caching Use cases:
 - Caching **infrequently changing data** in Database
 - Caching **user sessions** from applications
 - Caching static content
 - Caching **infrequently changing dynamic content**



Memorystore

- **In-memory datastore service:** Reduce access times
- **Fully managed (Provisioning, Replication, Failover & Patching)**
 - Highly available with 99.9% availability SLA
 - Monitoring can be easily setup using Cloud Monitoring
- **Support for Redis and Memcached:**
 - Use Memcached for Caching
 - Reference data, database query caching, session store etc
 - Use Redis for low latency access with persistence and high availability
 - Gaming Leader Boards etc
- **Can be accessed from:**
 - Compute Engine
 - App Engine flexible and standard
 - Google Kubernetes Engine
 - Cloud Functions



Memorystore

App Engine memcache service

```
def get_data():
    data = memcache.get('key')
    if data is not None:

        return data
    else:

        data = query_for_data()

        memcache.add('key', data, 60)
    return data
```

- Legacy in-memory data cache specifically for App Engine applications
 - Example Use Cases:
 - Speed up common datastore queries
 - Caching session data and user preferences
 - Temporary storage (not backed by persistent storage)
 - Two Service levels:
 - Shared memcache (FREE): best-effort caching
 - Dedicated memcache (\$\$\$\$): Fixed cache capacity dedicated to your app

Cloud CDN - Content Delivery Network

- Use Google's global edge network to serve global content with low latency
- Integrates with **External HTTP(S) Load Balancing**
 - LB provides frontend IP addresses and ports
- **Backends can be:**
 - Cloud Storage buckets, Instance groups, App Engine, Cloud Run, or Cloud Functions
 - Endpoints outside of Google Cloud (custom origins)
- **How Cloud CDN works?**
 - External HTTP(S) Load Balancing uses proxies - Google Front Ends (GFEs)
 - Request from user arrives at a Google Front End (GFE)
 - If URL maps to a backend with Cloud CDN configured:
 - If content is found in cache(cache hit), GFE sends cached response
 - If content is NOT found in the cache (cache miss), request is forwarded to backend (origin server)
 - Response is sent to user and cached
 - Using TTL settings to control cache duration



Cloud CDN - Best Practices

In 28
Minutes

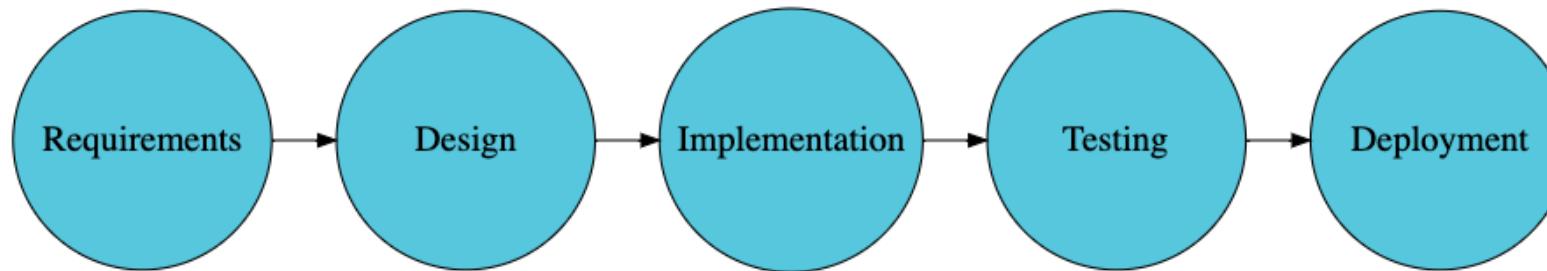


- **Cache static content**
 - Example: Cache-Control: public, max-age=259200 (72 hours)
- **Be careful with expiring time-sensitive (or dynamic) content**
 - Smaller cache periods. Example: Cache-Control: public, max-age=300 (5 minutes)
- **Using custom cache keys to improve cache hit ratio**
 - **Default cache key** - Entire URI - *https://yourwebsite.com/my-image/1.jpg*.
 - cache miss - *http://yourwebsite.com/my-image/1.jpg* (http vs https)
 - cache miss - *http://yourwebsite.com/my-image/1.jpg?mobile=1* (query string does not match)
 - **Customize cache key:** Any combination of protocol, host, or query string
 - `gcloud compute backend-services update BACKEND_SERVICE --enable-cdn --no-cache-key-include-protocol --no-cache-key-include-host --no-cache-key-include-query-string`
- **Using Versioned URLs to update content**
 - *https://yourwebsite.com/my-image/1.jpg?v=1*
 - *https://yourwebsite.com/my-image/1.jpg?v=2*

Evolution

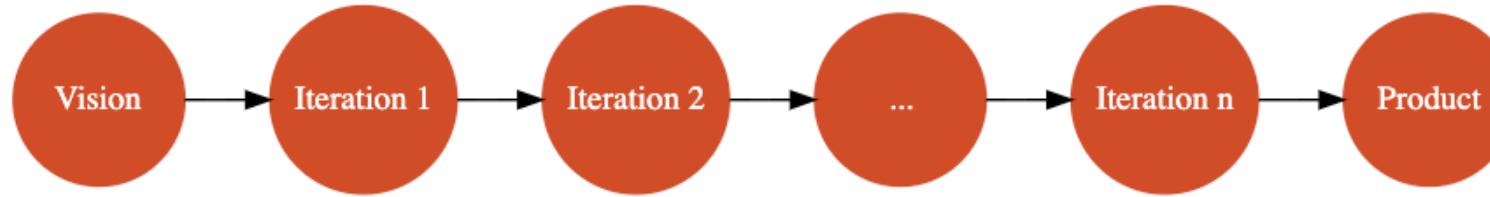
Agile > DevOps > SRE

Software development life cycle (SDLC) - Waterfall



- **Software development in multiple long phases:**
 - Requirements
 - Design
 - Implementation
 - Testing
 - Deployment

Software development life cycle (SDLC) - Spiral

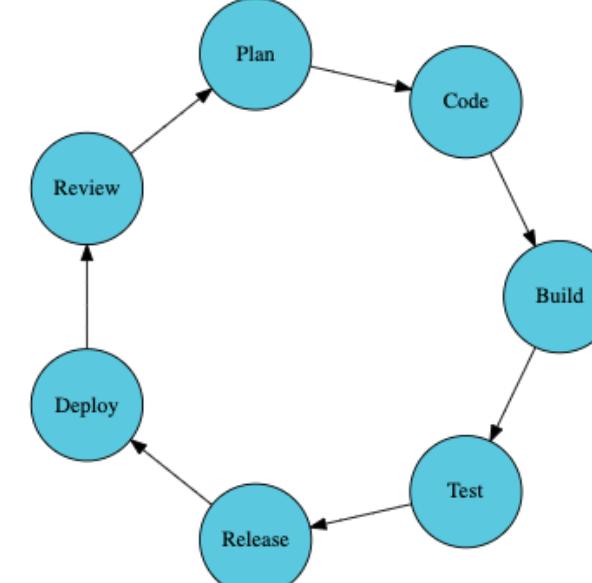


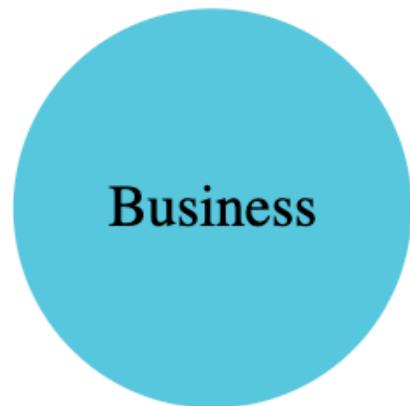
- **Software development in smaller iterations:**

- Start
- Iteration 1
- Iteration 2
- ...
- ...

Software development life cycle (SDLC) - Agile

- **Principles**
 - Individuals and interactions over processes and tools
 - Working software over comprehensive documentation
 - Customer collaboration over contract negotiation
 - Responding to change over following a plan
 - Now there are 12 principles (<https://agilemanifesto.org/principles.html>)
- **Agile is recommended for most software development:**
 - BUT add a bit of rigidity from waterfall model for critical safety software (Flight navigation software, Medical devices software etc)

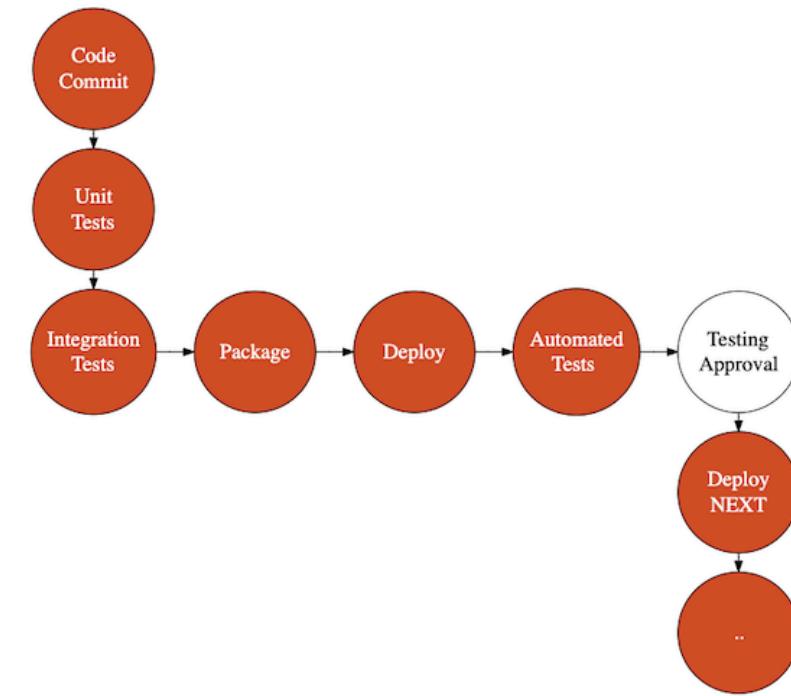




- Getting Better at "**Three Elements of Great Software Teams**"
 - **Communication** - Get teams together
 - **Feedback** - Earlier you find a problem, easier it is to fix
 - **Automation** - Automate testing, infrastructure provisioning, deployment, and monitoring

DevOps - CI, CD

- **Continuous Integration**
 - Continuously run your tests and packaging
- **Continuous Deployment**
 - Continuously deploy to test environments
- **Continuous Delivery**
 - Continuously deploy to production



DevOps - CI CD - Recommended Things to Do

- **Static Code Analysis**
 - Lint, Sonar
 - Including Static Security Checks (Source Code Security Analyzer software like Veracode or Static Code Analyzer)
- **Runtime Checks**
 - Run Vulnerability Scanners (automated tools that scan web applications for security vulnerabilities)
- **Tests**
 - Unit Tests (JUnit, pytest, Jasmine etc)
 - Integration Tests (Selenium, Robot Framework, Cucumber etc)
 - System Tests (Selenium, Robot Framework, Cucumber etc)
 - Sanity and Regression Tests

DevOps - CI, CD Tools

- **Cloud Source Repositories:** Fully-featured, private Git repository
 - Similar to Github
- **Container Registry:** Store your Docker images
- **Jenkins:** Continuous Integration
- **Cloud Build:** Build deployable artifacts (jars or docker images) from your source code and configuration
- **Spinnaker:** Multi-cloud continuous delivery platform
 - Release software changes with high velocity and confidence
 - Supports deployments to Google Compute Engine, Google Kubernetes Engine, Google App Engine and other cloud platforms
 - Supports Multiple Deployment Strategies



Container Registry



Cloud Source Repositories

DevOps - Infrastructure as Code

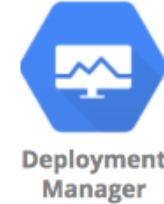
In 28
Minutes



- Treat **infrastructure the same way as application code**
- Track your infrastructure **changes over time** (version control)
- Bring **repeatability** into your infrastructure
- Two Key Parts
 - **Infrastructure Provisioning**
 - Provisioning compute, database, storage and networking
 - Open source cloud neutral - Terraform
 - GCP Service - Google Cloud Deployment Manager
 - **Configuration Management**
 - Install right software and tools on the provisioned resources

Google Cloud Deployment Manager - Introduction

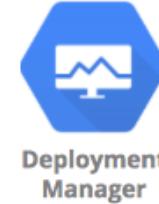
- Lets consider an example:
 - I would want to create a new VPC and a subnet
 - I want to provision a Load balancer, Instance groups with 5 Compute Engine instances and an Cloud SQL database in the subnet
 - I would want to setup the right Firewall
- AND I would want to create 4 environments
 - Dev, QA, Stage and Production!
- Deployment Manager can help you do all these with a simple (actually NOT so simple) script!



Deployment
Manager

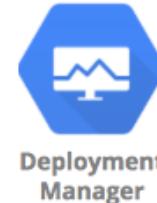
Google Cloud Deployment Manager - Advantages

- Automate deployment and modification of Google Cloud resources in a controlled, predictable way
 - Deploy in multiple environments easily!
- Avoid configuration drift
- Avoid mistakes with manual configuration
- Think of it as version control for your environments
- **Important Note** - Always modify the resources created by Deployment Manager using Deployment Manager



Google Cloud Deployment Manager

- All configuration is defined in a simple text file - YAML
 - I want a VPC, a subnet, a database and ...
- Deployment Manager understands dependencies
 - Creates VPCs first, then subnets and then the database
- (Default) Automatic rollbacks on errors (Easier to retry)
 - If creation of database fails, it would automatically delete the subnet and VPC
- Version control your configuration file and make changes to it over time
- Free to use - Pay only for the resources provisioned
 - Get an automated estimate for your configuration



Cloud Deployment Manager - Example

```
- type: compute.v1.instance
  name: my-first-vm
  properties:
    zone: us-central1-a
    machineType: <<MACHINE_TYPE>>
    disks:
      - deviceName: boot
        type: PERSISTENT
        boot: true
        autoDelete: true
        initializeParams:
          sourceImage: <<SOURCE_IMAGE>>
  networkInterfaces:
    - network: <<NETWORK>>
      # Give instance a public IP Address
      accessConfigs:
        - name: External NAT
          type: ONE_TO_ONE_NAT
```

Cloud Deployment Manager - Terminology

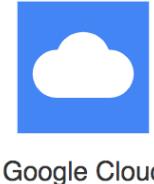
- **Configuration file:** YAML file with resource definitions for a single deployment
- **Templates:** Reusable resource definitions that can be used in multiple configuration files
 - Can be defined using:
 - Python (preferred) OR
 - JinJa2 (recommended only for very simple scripts)
- **Deployment:** Collection of resources that are deployed and managed together
- **Manifests:** Read-only object containing original deployment configuration (including imported templates)
 - Generated by Deployment Manager
 - Includes fully-expanded resource list
 - Helpful for troubleshooting

Cloud Marketplace (Cloud Launcher)

- Installing custom software might involve setting up multiple resources:
 - Example: Installing WordPress needs set up of compute engine and a relational database
- How do you simplify the set up of custom software solutions like Wordpress or even more complex things like SAP HANA suite on GCP?
- **Cloud Marketplace:** Central repo of easily deployable apps & datasets
 - Similar to App Store/Play Store for mobile applications
 - You can search and install a complete stack
 - Commercial solutions - SAP HANA etc
 - Open Source Packages - LAMP, WordPress, Cassandra, Jenkins etc
 - OS Licenses: BYOL, Free, Paid
 - Categories: Datasets/Developer tools/OS etc
 - When selecting a solution, you can see:
 - Components - Software, infrastructure needed etc
 - Approximate price

Site Reliability Engineering (SRE)

- DevOps++ at Google
- SRE teams **focus on every aspect of an application**
 - availability, latency, performance, efficiency, change management, monitoring, emergency response, and capacity planning
- **Key Principles:**
 - Manage by Service Level Objectives (SLOs)
 - Minimize Toil
 - Move Fast by Reducing Cost of Failure
 - Share Ownership with Developers



Site Reliability Engineering (SRE) - Key Metrics

- **Service Level Indicator(SLI):** Quantitative measure of an aspect of a service
 - Categories: availability, latency, throughput, durability, correctness (error rate)
 - Typically aggregated - "Over 1 minute"
- **Service Level Objective (SLO) - SLI + target**
 - 99.99% Availability, 99.99999999% Durability
 - Response time: 99th percentile - 1 second
 - Choosing an appropriate SLO is complex
- **Service Level Agreement (SLA):** SLO + consequences (contract)
 - What is the consequence of NOT meeting an SLO? (Defined in a contract)
 - Have stricter internal SLOs than external SLAs
- **Error budgets:** $(100\% - \text{SLO})$
 - How well is a team meeting their reliability objectives?
 - Used to manage development velocity

Site Reliability Engineering (SRE) - Best Practices

- **Handling Excess Loads**

- **Load Shedding**

- API Limits
 - Different SLAs for different customers
 - Streaming Data
 - If you are aggregating time series stream data, in some scenarios, you can drop a part of data



Google Cloud

- **Reduced Quality of Service**

- Instead of talking to a recommendations API, return a hardcoded set of products!
 - Not always possible:
 - Example: if you are making a payment

- **Avoiding Cascading Failures**

- **Plan to avoid thrashing**

- Circuit Breaker
 - Reduced Quality of Service

Site Reliability Engineering (SRE) - Best Practices - 2

- **Penetration Testing (Ethical Hacking)**
 - Simulate an attack with the objective of finding security vulnerabilities
 - Should be authorized by project owners
 - No need to inform Google
 - Ensure you are only testing your projects and are in compliance with terms of service!
 - Can be white box (Hacker is provided with information about infrastructure and/or applications) or black box (No information is provided)
- **Load Testing (JMeter, LoadRunner, Locust, Gatling etc)**
 - Simulate real world traffic as closely as possible
 - Test for spiky traffic - suddenly increases in traffic



Google Cloud

Site Reliability Engineering (SRE) - Best Practices - 3

- **Resilience Testing** - "How does an application behaves under stress?"
- **Resilience** - "Ability of system to provide acceptable behavior even when one or more parts of the system fail"
- **Approaches:**
 - **Chaos Testing (Simian Army)** - cause one or more layers to fail
 - "unleashing a wild monkey with a weapon in your data center to randomly shoot down instances and chew through cables"
 - Add huge stress on one of the layers
 - **Include network in your testing (VPN, Cloud Interconnect etc..)**
 - Do we fall back to VPN if direct interconnect fails?
 - What happens when internet is down?
 - **Best Practice: DiRT** - disaster recovery testing at Google
 - Plan and execute outages for a defined period of time
 - Example: Disconnecting complete data center

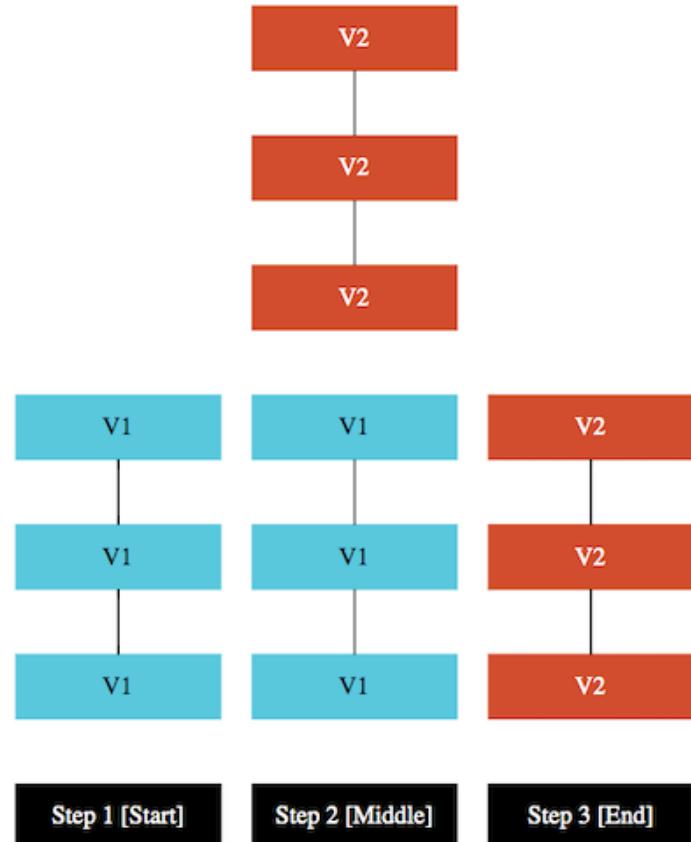


Google Cloud

Release Management

Release Management

- **Goals:** vary from app to app
 - Zero Downtime
 - Only one version live at a time
 - Minimize Costs (and infrastructure needed)
 - Test with production traffic before going live
- **Best Practices:**
 - Small incremental changes
 - Automation (as much as possible)
 - Handling problems with new releases:
 - Analyze logs and metrics from Cloud Monitoring and Logging
 - Rollback to previous release and try replicating the problem in other environments



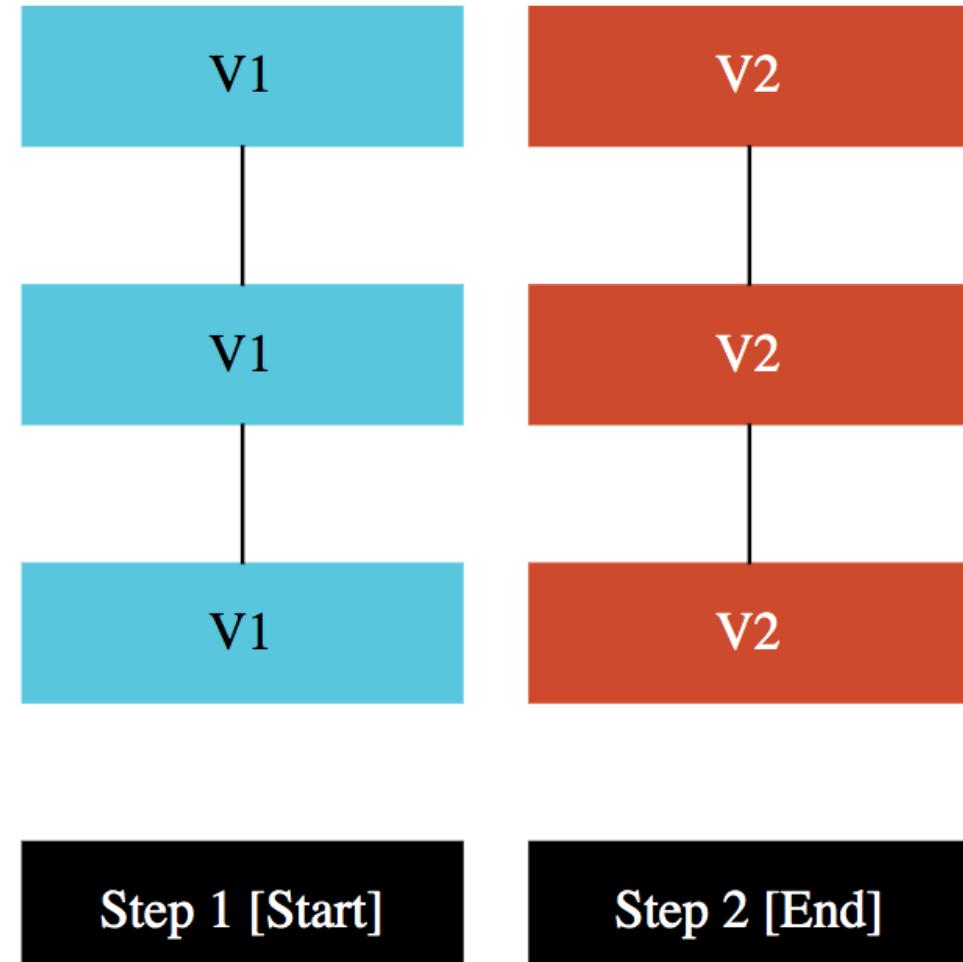
Deployment Approach : Recreate

- **Approach:**

- Terminate Version 1
- Roll out Version 2

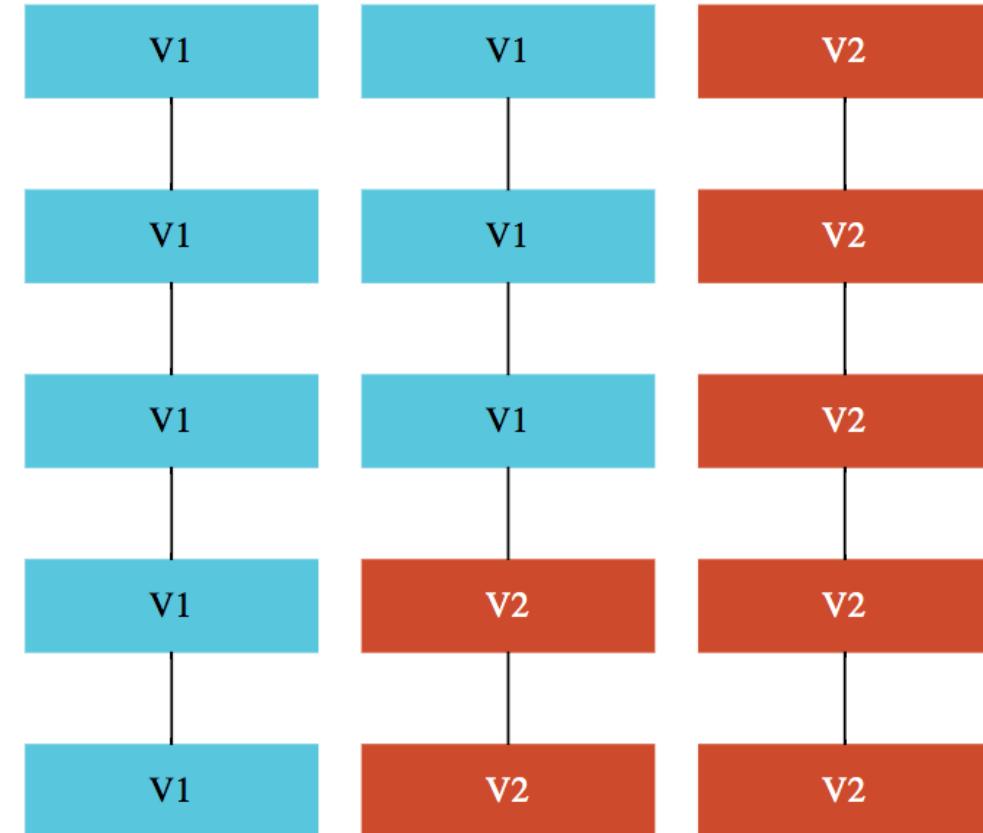
- **Characteristics:**

- App down during the release
- Rollback needs redeployment
 - AND more downtime
- Cost effective and Fast
 - BUT disruptive
- Avoid need for backward compatibility (data and applications)



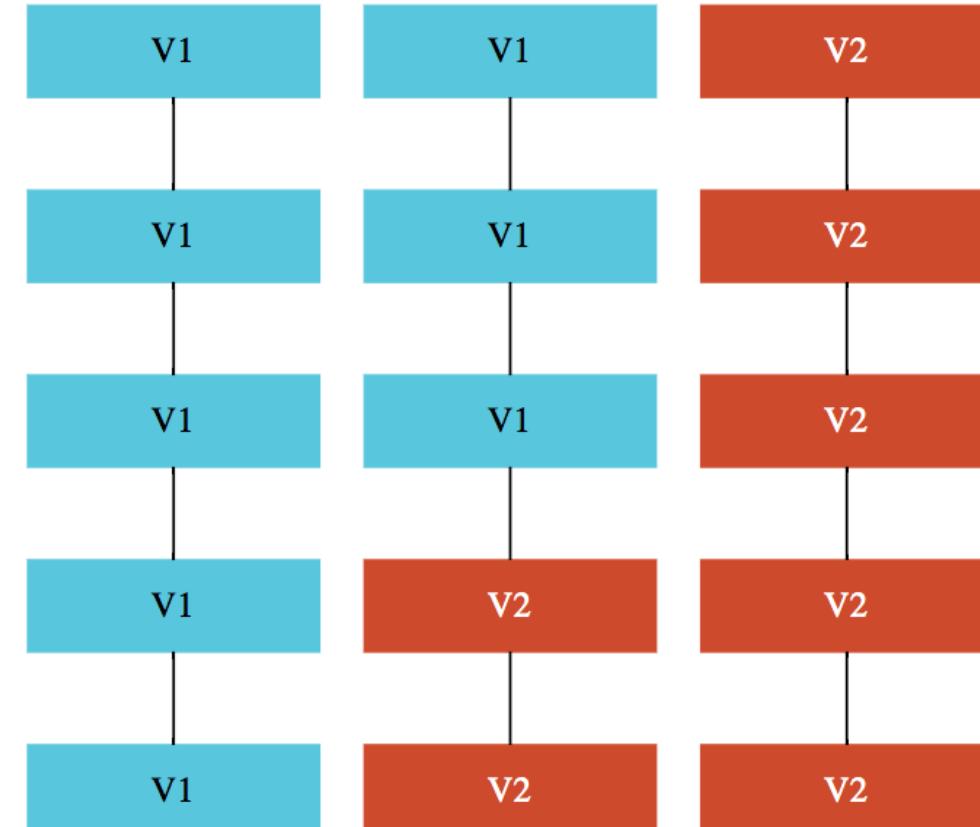
Deployment Approach : Canary

- **Approach:**
 - Step 1: V2 rolled out to a subset of instances
 - Step 2: On successful testing, V2 rolled out to all instances
 - OR V2 is rolled back in case of failure
- **Characteristics:**
 - Fast
 - Zero downtime
 - No extra infrastructure
 - Minimizes impact to users (in case of release failures)
 - Needs Backward compatibility (data and applications)



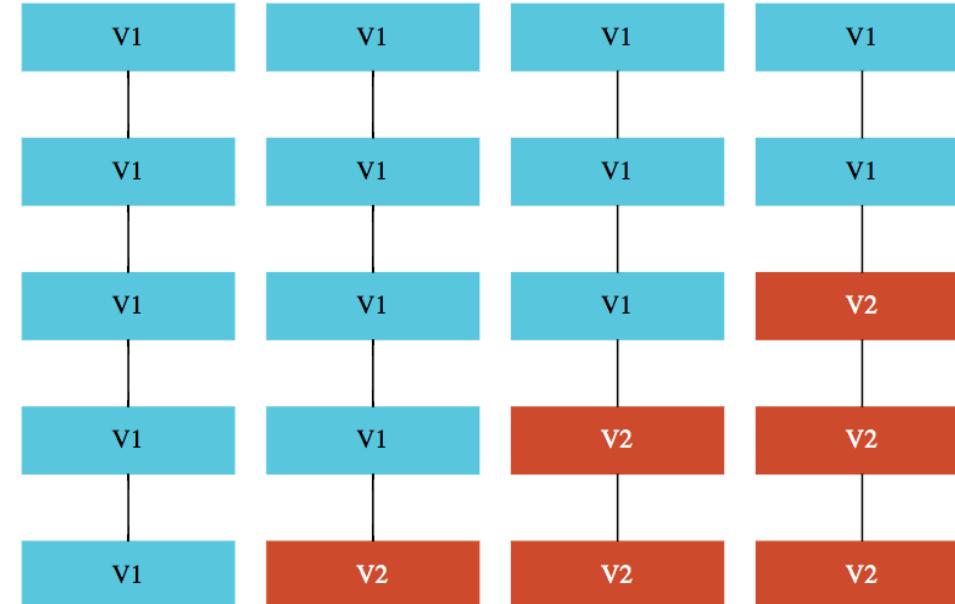
Testing Approach : A/B Testing

- **Use case:** You want to see if users like a feature!
- **Approach:**
 - **Step 1:** V2 (with new feature) rolled out to a subset of users
 - **Step 2:** On successful testing, V2 rolled out to all users
 - OR we go back to V1 in case users don't like the feature!
- **Characteristics:**
 - Gives the ability to test if your users like a feature



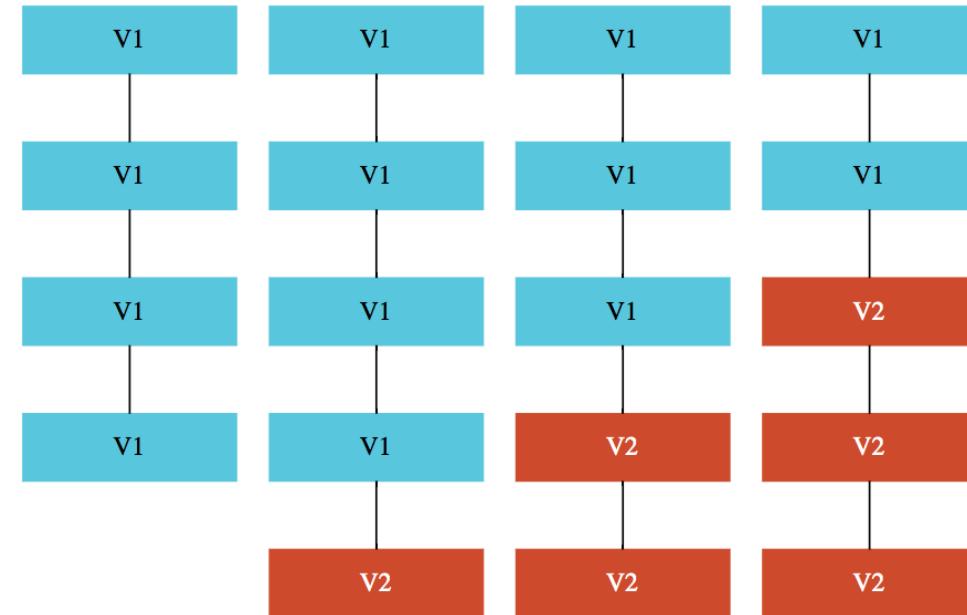
Deployment Approach : Rolling

- **Approach:**
 - Step 1: V2 rolled out to a percentage of instances (Example window size: 5%)
 - Step 2..N: V2 gradually rolled out to rest of the instances (Example: 5% at a time)
- **Characteristics:**
 - Slow
 - Zero downtime
 - Needs automation and additional setup
 - No extra infrastructure
 - Minimizes impact to users (in case of release failures)
 - Needs Backward compatibility (data and applications)



Deployment Approach : Rolling with Additional Batch

- **Approach:**
 - Step 1: Additional batch of new instances are created with V2 (Example: 5%)
 - Step 2..N: V2 gradually rolled out to the instances batch by batch (Example: 5% at a time)
- **Characteristics:**
 - Same as Rolling Deployment except for:
 - Needs Little bit of extra infrastructure
 - ZERO reduction in number of instances handling user requests



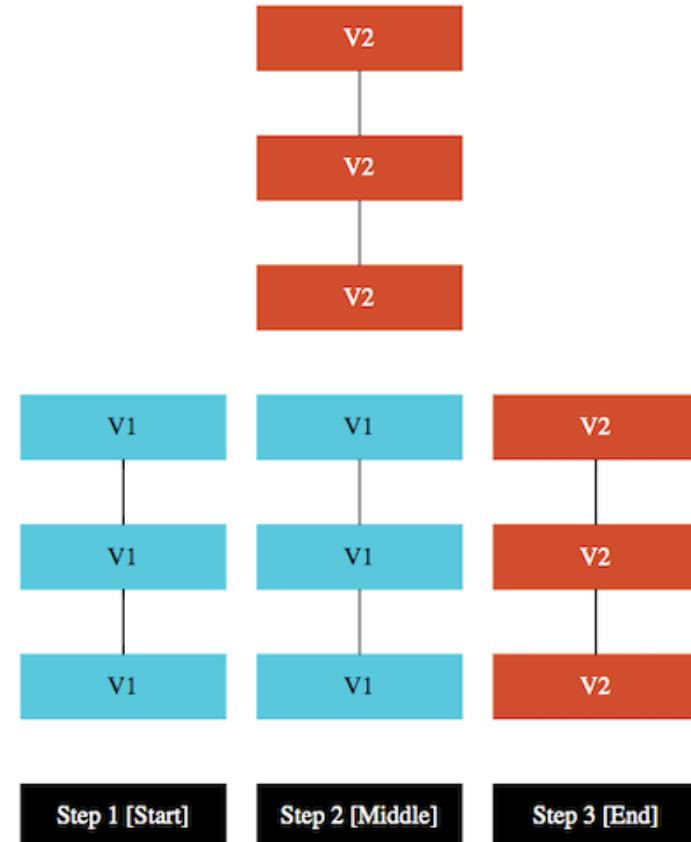
Deployment Approach : Blue Green

- **Approach:**

- Step 1: V1 is Live
- Step 2: Create (or replicate) a parallel environment with V2
- Step 3: Switch all traffic from V1 to V2 (and remove V1 Environment)

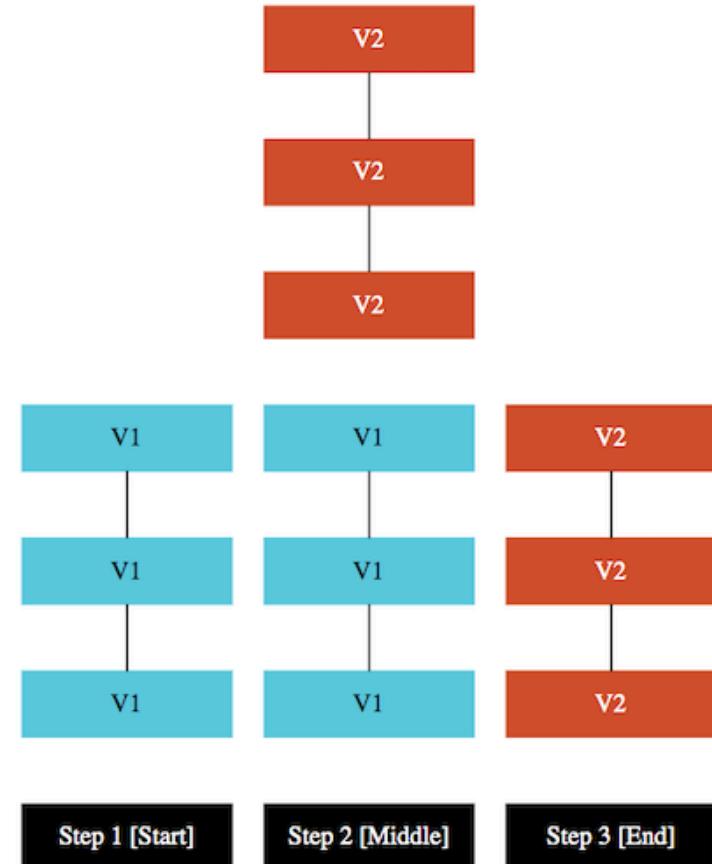
- **Characteristics:**

- Instant
- Zero Downtime
- Easy Rollback
- Needs additional infra (during the release)
- ZERO reduction in available capacity
- Needs Backward compatibility (data and apps)



Testing Approach : Shadow

- **Approach:**
 - Step 1: V1 is Live
 - Step 2: Create (or replicate) a parallel environment with V2
 - Mirror traffic to V1 and V2
 - Step 3: Switch all traffic from V1 to V2 (and remove V1 Environment)
- **Characteristics:**
 - Zero production impact: Test V2 with real production traffic before releasing
 - You can also capture and replay live production traffic
 - Complicated : You don't want double payments (might need stubbing)
 - Needs a lot of additional infrastructure



Deployment Approaches - Managed instance group (MIG)

Option	Details
Rolling Release	<pre>gcloud compute instance-groups managed rolling-action start-update my-mig --version=template=v2-template --max-surge=5 or 10% (Max instances updated at a time) --max-unavailable=5 or 10% (Max instances that can be down during update)</pre>
Canary Release	<pre>gcloud compute instance-groups managed rolling-action start-update my-mig --version=template=v1-template --canary-version=template=v2-template,target-size=10%</pre>
Blue Green Deployment	Create a new MIG and make manual adjustments to Load Balancer backends as needed

App Engine - Releasing New Versions

Option	Details
Deploy & shift all traffic at once	<code>gcloud app deploy</code>
Deploy v2 without shifting traffic	<code>--no-promote</code>
Shift traffic to V2 all at once	<code>gcloud app services set-traffic s1 --splits V2=1</code>
Gradual Migration	Add <code>--migrate</code> option to previous command
A/B testing	<code>gcloud app services set-traffic s1 --splits=v2=.5,v1=.5</code>

GKE - Releasing New Versions

In 28
Minutes

Option	Details
Recreate	<p>Set <code>strategy > type</code> on Deployment to Recreate</p> <p>Use '<code>kubectl set image deployment</code>' OR Update deployment YAML to perform deployment</p>
Rolling Update	<p>Set <code>strategy > type</code> on Deployment to <code>RollingUpdate</code></p> <p>Use '<code>kubectl set image deployment</code>' or Update deployment YAML to perform deployment</p> <p>Configure <code>maxSurge</code> and <code>maxUnavailable</code> to control deployment</p>
Blue Green Deployment	<p>Create New Deployment.</p> <p>Control traffic using Ingress (or Service)</p>
Canary Deployment	Needs Service Mesh like Istio

Compliance and Regulations

Compliance and Regulations

- Google Cloud Platform is compliant with several important certifications/regulations/standards:
 - ISO/IEC 27001 (security controls that can help manage information risks)
 - ISO/IEC 27017 (information security controls applicable to the provision and use of cloud services)
 - ISO/IEC 27018 (critical components of cloud privacy - personally identifiable information (PII))
 - ISO/IEC 27701 (global privacy standard)
 - PCI DSS - Payment Card Industry (PCI) Data Security Standards (DSS)
 - SOC 1 (Auditing standard - How well does the vendor keep their books?)
 - SOC 2 (Assessment of service provider controls - Security, Availability, Processing Integrity, Confidentiality, Privacy)



Google Cloud

Compliance and Regulations - 2

- Google Cloud Platform is compliant with several important certifications/regulations/standards:
 - COPPA: Children's Online Privacy Protection Act of 1998
 - Special requirements on websites created for children under the age of 13
 - HIPAA: Health Insurance Portability and Accountability Act of 1996
 - Data privacy and security requirements for organizations handling protected health information (PHI)
 - General Data Protection Regulation (GDPR): Strengthens personal data protection in Europe
- **Customers are responsible** for ensuring that their applications are compliant with certifications/regulations/standards



Google Cloud

- **HIPAA Compliance is a shared responsibility:**
 - GCP supports HIPAA compliance
 - BUT customers are responsible for evaluating HIPAA compliance
- **Execute a Google Cloud Business Associate Agreement (BAA)**
 - You can request a BAA directly from your account manager
 - HIPAA BAA covers GCP's entire infrastructure (across all regions)
 - Multi-regional service redundancy including usage of Preemptible VMs
 - Do not use Google Cloud Products not covered by BAA
- **Recommended Best Practices:**
 - Follow IAM best practices
 - Consider enabling Object Versioning on Cloud Storage buckets
 - Recommended: Export audit logs to Cloud Storage (archival) and BigQuery (analytics)
 - Disable request caching of PHI in Cloud CDN
- Reference: <https://cloud.google.com/security/compliance/hipaa/>

PCI Data Security Standards (DSS)

- Payment Card Industry DSS: Enhance card-holder security
- Best Practices and Recommendations:
 - Create a new Google Cloud account for your payment processing environment
 - Isolate from other environments
 - Restrict access to payment processing environment
 - Follow "principle of least privilege"
 - Control inbound and outbound traffic
 - Creating firewall rules to allow ONLY following inbound traffic
 - HTTPS requests from customers
 - Responses from third-party payment processor
 - Office network - for auditing and management
 - Only allow outbound traffic (HTTPS) to third-party payment processor
 - Compute Engine and GKE are recommended compute services (App Engine, Cloud Functions do NOT support egress firewall rules)
 - Create an HTTPS load balancer with signed SSL certificate
 - Create hardened secure Linux image with the minimum software needed to run



Google Cloud

PCI DSS - Other Recommendations

- **Best Practices and Recommendations:**
 - **Automated most of the processes:**
 - Deploy using Cloud Deployment Manager
 - Configuration management using Chef, Puppet, Ansible, or Salt
 - Build an automated recovery plan
 - **Implement Forseti Security:** open-source tools to improve security of GCP environments:
 - Inventory: Compare your Google Cloud resources (inventory) over time
 - Scanner: Compare role-based access policies over time
 - Enforcer: Compare firewall rules with a specified state
 - Explain: Provides visibility into your IAM policies
 - **Enable VPC Flow Logs, Access Transparency Logs, Firewall Rules Logging and Configure Monitoring Alerts**
 - Stream audit logs to Cloud Storage and use bucket locks to make logs immutable
 - Streaming audit logs to BigQuery for analysis
 - **Using Cloud Data Loss Prevention to sanitize data:**
 - Grant apps access to PCI data only after it has been sanitized with Cloud Data Loss Prevention

Migration

Cloud Migration Planning

- Phase 1: Assess the workloads to be migrated
- Phase 2: Plan the foundation
- Phase 3: Deploy the workloads
- Phase 4: Optimize your environment



Google Cloud

Cloud Migration Approaches

- You have a **combination of following options:**
 - **Rehosting** ("lift and shift")
 - **Replatforming**
 - Few adjustments to suit the cloud
 - Example: Containerizing
 - **Repurchasing**
 - Move to a new, cloud-native product
 - Move to a new database
 - **Refactoring**
 - Example: Serverless Computing
 - Most expensive
 - **Retiring**
 - End of service
 - **Retaining**
 - Do NOT move to Cloud
 - Stay on-premises



Google Cloud

Cloud Migration Planning - Phases 1 & 2

- **Phase 1: Assess the workloads to be migrated**
 - Take inventory and Catalog apps
 - Experiment and design proofs of concept
 - Calculate total cost of ownership
 - Choose which workloads to migrate first
 - Based on Business value, Teams, Dependencies, Refactoring effort, Licensing and compliance needs, Availability and reliability requirements
- **Phase 2: Plan the foundation**
 - Design Resource Organization Hierarchy, Configure IAM (users, groups, integrate with Identity Provider (IdP)) and Design network topology and connectivity
 - Plan for Security (data, apps), Monitoring (and alerting) and Governance
 - Plan your Migration Team



Google Cloud

Cloud Migration Planning - Phases 3 & 4

- **Phase 3: Deploy the workloads**

- Migrate Data
 - Consider Cost, Time, Offline versus online transfer options and Security
- Deploy Applications
- Prefer automation :
 - Automate configuration management with Ansible, Chef or Puppet
 - Automate build and deployment using Jenkins, SonarQube, Cloud Build or Spinnaker
 - Implement Infrastructure as Code using Terraform or Deployment Manager



Google Cloud

- **Phase 4: Optimize your environment**

- Ensure that logging, monitoring and alerting are in place
- Reduce overhead by preferring Managed Services
- Optimize costs using autoscaling

Database Migration - MS SQL Server to GCP

- Cloud SQL for SQL Server is fully managed
- **Migration Steps**
 - Create a Cloud SQL for SQL Server instance
 - Move backup of your database to Cloud Storage
 - Import the database into Cloud SQL for SQL Server
 - Validate the imported data



Migration - Deploying Containers to GCP

Option	Details
App Engine flexible environment	Highly scalable BUT cannot scale down to ZERO Doesn't let you customize underlying Compute Engine VMs Very less management burden
Cloud Run and Cloud Run for Anthos	Highly scalable AND You can scale down to zero instances Cannot customize the environment Almost ZERO Management burden
Google Kubernetes Engine (GKE) and Anthos clusters	Highly scalable You need to manage the clusters You can customize cluster nodes as per your needs
Compute Engine	You can use Container-Optimized OS (COS) image with Docker installed <code>gcloud compute instances create-with-container VM_NAME --container-image DOCKER_IMAGE</code> Launches your container on startup of VM NOT Recommended: You need to manage everything : Scalability,

Architects & Architecture Must Know

Architect : Understand Business Requirements

- **Business Requirements :** Define what your business needs
- Examples of Business Requirements:
 - Reduce costs (Understand Capital Expenditure vs Operational Expenditure)
 - Managed Services
 - Autoscaling
 - Preemptible VMs
 - Increase pace of innovation
 - Evaluate and adopt emerging processes like DevOps (CI/CD) and SRE
 - Evaluate and adopt emerging architectures like Microservices
 - Reduce mean time to recovery
 - Improve compliance with regulations
 - More visibility into applications and infrastructure (metrics)
 - More intelligence from the available data



Google Cloud

Total Cost of Ownership(TCO)

- **Total Cost of Ownership(TCO) includes:**
 - Licensing Costs (Software + Hardware)
 - Computing Costs
 - Storage Costs
 - Networking Costs (Connection cost + Data Ingress + Data Egress)
 - Personnel Costs (Dev + Test + Ops + Business + ..)
 - Other Costs
 - Penalties for missed SLAs or Compliance needs
 - Third Party APIs
 -
- **When designing solutions, ensure that you take Total Cost of Ownership(TCO) into account!**



Google Cloud

Architect : KPIs for Business Requirements

- How can you measure if you are meeting Business Objectives?
- Define KPIs (Key Performance Indicator) where possible:
 - How is the business doing with respect to an objective?
 - Project KPIs
 - Example: Number of new customers in a week
 - Example: Percentage of virtual machines running in the cloud
 - Operational KPIs
 - Example: Operational cost per customer



Google Cloud

Defining Technical Requirements

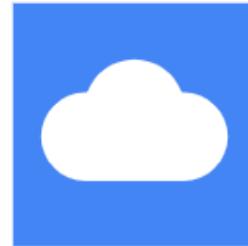
- Technical aspects that your system must adhere to

- **Functional:**

- Must use containers
 - Must use hardened Linux OS
 - Needs container orchestration
 - Must be auto scaling
 - Must be NoSQL database with flexible schema
 - Able to store huge volumes of archives at very low cost
 - Must use a private network (traffic should not go over internet)

- **Non Functional:**

- Availability
 - Scalability
 - Durability
 - Security



Google Cloud

Planning for High Availability

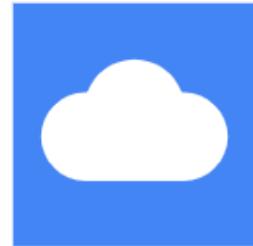
Service/Feature	Features / Best Practices
Geographical Distribution	Availability: Global > Multi-Regional > Regional > Zonal
Compute Engine	Live Migration Managed Instance Groups with AutoScaling and Health Checks (Auto healing) Distribute instances across regions/zones and distribute using Global Load Balancing
GKE	Multi master, Regional clusters with pod and cluster autoscaling
Managed Services	Use Managed Services like App Engine, Cloud Functions, Cloud Storage, Cloud Filestore, Cloud Datastore, BigQuery
Persistent Disks	Live resizing improves availability Use Regional Persistent Disks
Cloud Bigtable	Place clusters in different zones or regions (Each cluster belongs to a zone and you can create multiple clusters for high availability in the same instance)

Planning for High Availability - 2

Service/Feature	Features / Best Practices
Cloud Datastore	Use Multi-region locations
Cloud SQL	<p>Use HA configuration (regional) - Cluster with primary instance and a standby instance is created</p> <p>Read replicas will NOT be promoted (So Read Replicas do NOT increase availability for Cloud SQL)</p>
Network Tier	Prefer Premium Network Tier to Standard Network Tier
Hybrid Network Connectivity	<p>Speed and Availability: Dedicated interconnect > Partner interconnect > VPN</p> <p>Have a backup connection.</p> <p>Example: VPN can be a backup for Dedicated interconnect or have multiple Dedicated interconnect connections</p>

Planning for Scalability

- **Highly Scalable Compute Engine Architecture** : VMs in MIG configured by instance template (and load balanced with Load Balancing)
 - (Remember) Unmanaged Instance Groups do not support Auto Scaling
- Use Pod and Cluster Autoscaling for GKE
- Be cautious with resources that cannot scale fast
 - Cloud SQL does NOT scale horizontally (only vertically)
- Stateful applications are more difficult to scale than Stateless applications
 - You can move state to a cache (like Memorystore) or a database
- Local SSD is least scalable (Max 8 per VM)
 - Data survives reboots and live migrations
 - But data is lost when VM is stopped or terminated!



Google Cloud

Planning for Scalability - 2

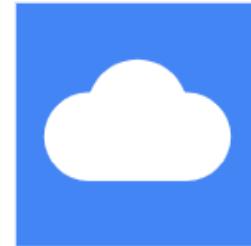
- **Persistent Disks:** Can be scaled horizontally and vertically
 - You can increase the size of PD while it is attached to a VM
 - You can attach new PDs (up to a max of 128) while a VM is running
- **Compute:**
 - App Engine and Cloud Functions are serverless (Auto Scaling)
- **Databases:**
 - Pub/Sub, BigQuery and Cloud Datastore are serverless (Auto Scaling)
 - BigTable, Cloud Spanner, Cloud SQL, Dataproc are **NOT serverless**
 - You need to choose the number of nodes in the cluster (and also the type of node for Cloud SQL and Dataproc)



Google Cloud

Planning for Security

- **Confidentiality** - Only right people have the right access
 - Follow IAM Best Practices
 - Encryption at rest and in transit (along with physical controls)
 - (DEFAULT) GCP encrypts all data at rest
- **Integrity** - Protect data from unauthorized change
 - Follow IAM Best Practices
 - Role Based Access
 - Separation of duties
 - Hash verifications and digital signatures
- **Availability** - Systems/Data is/are available when users need them
 - Solutions: Firewalls, Redundancy, Automatic Failover etc
 - Protect from Denial of Service (DoS) attacks



Google Cloud

Implementing DDoS Protection and Mitigation

- **(DDoS) attack:** Attempting to bring your apps down with large scale attacks
- **Shared responsibility** between GCP and Customer
 - GCP provides certain features to protect you from DDoS attacks
 - Anti-spoofing protection for the private network
 - Isolation between virtual networks
 - App Engine (sits behind Google Front End) automatically protects you from Layer 4 and below attacks
 - Examples: SYN floods, IP fragment floods, port exhaustion, etc.
 - What you can do to protect apps from DDoS attacks?
 - Reduce the attack surface: Make use of subnetworks and networks, firewall rules and IAM
 - Isolate your internal traffic
 - Use Private IP Address (unless you need Public IP Addresses)
 - Use private load balancing for internal traffic
 - Use Proxy-based Load Balancing: Automatically protects you from Layer 4 and below attacks
 - Integrate Load Balancing with Cloud Armor
 - IP-based and geo-based access control
 - Enforce Layer 7 security policies on hybrid and multi-cloud deployments
 - Pre-configured WAF rules (OWASP Top 10)

Digital Signatures - Cloud KMS

In 28
Minutes

- When do we use Digital Signatures?
 - Purpose 1: Verification of the integrity of the signed data
 - Purpose 2: Non-repudiation if the signer claims the signature is not authentic
 - Sender cannot deny sending the message later
- Workflow:
 - Sender performs private key operation over the data to create a digital signature
 - Recipient uses the public key to verify the digital signature
 - If verification is unsuccessful, then the data has been altered
- Cloud KMS can be used to create an asymmetric key pair (public and private key pair) that supports digital signing
 - Provides APIs/commands to create digital signatures
 - `gcloud kms asymmetric-sign`
- Use cases: Validating code builds



- **Cloud Armor:** Protect your apps from denial of service and OWASP Top 10 attacks
 - Protects you from **common web attacks** (OWASP Top 10) like XSS (cross-site scripting) and SQL injection
 - Protect applications deployed in Google Cloud, in a hybrid deployment, or in a multi-cloud architecture
 - Integrates very well with Google Cloud Load Balancing
 - Provides **preconfigured security policies** (OWASP Top 10 risks etc)
 - Customize rules as per your needs
 - **Usecases**
 - Enable access for users at specific IP addresses with allowlists
 - Block access for users at specific IP addresses with denylists
 - Protect applications against OWASP Top 10 risks



Secret Manager

```
SecretVersionName secretVersionName = SecretVersionName.of(projectId, secretId, versionId)
AccessSecretVersionResponse response = client.accessSecretVersion(secretVersionName);
String secretValue = response.getPayload().getData().toStringUtf8();
```

- How do you **manage your database passwords, your API keys** securely?
- **Secret Manager** - Store API keys, passwords etc
 - Multiple versions of secrets
 - Automate rotation with Cloud Functions
 - Auditing with Cloud Audit Logs
 - Encrypted by default
- Best Practice: **Do NOT store credentials/passwords in code**
 - Use Secret Manager and access secrets in your application

Architect Responsibilities: Stakeholder Management

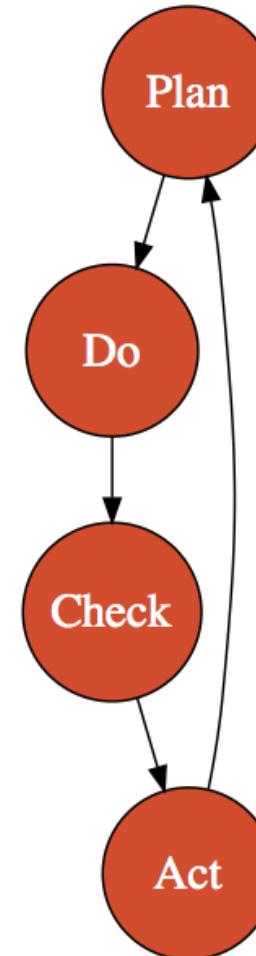
- **Stakeholder:** "a person with an interest or concern in something"
 - Different stakeholders have different priorities/interests/degrees of influence:
 - CEO/CTO
 - Business Process Owner
 - Product Owner
 - Architecture Team
 - Scrum Master
 - Development Team
 - Testing Team
 - Security Team
 - Compliance Team
- As an Architect, you need to **understand the needs of different stakeholders and design solutions** that meet them
 - Key to Stakeholder Management: **Early clear communication**
 - Identify stakeholders, their interests and build a plan to effectively communicate with them



Google Cloud

Architect Responsibilities: Change Management

- **Changes are inevitable**
 - People, Process and Technology (systems, hardware, and software)
 - But humans hate change
- **Change Management:** How do you deal with changes and reduce impact?
 - Understand the change:
 - Reason: Why make the change?
 - Risks: What are the risks?
 - Resources: Who is impacted? Who can help make the change? What is needed?
 - Responsibility: Who can make it happen?
 - Follow the cycle: Plan - Do - Check - Act



Architect Responsibilities: Business Continuity Planning

- **BCP:** How to keep business running in face of disasters?
- **Disaster Recovery :** Parts of BCP focused IT operations
 - Example: Use cloud/on-prem as DR environment
 - Example: Have a backup network connection between cloud and on-prem
 - **Disaster Recovery Plan:** How should an enterprise respond to different types of disasters?
 - Includes People, Process and Technology (systems, hardware, and software)
 - Identifies critical services, personnel and plans
 - Define Recovery Time Objective (RTO) and Recovery Point Objective (RPO)
 - Design and Architect your solutions to meet your RTO and RPO objectives
 - DR plans should be **continuously tested** (Game days)
 - Are you able to restore a database from archive?
- **BCP = Disaster Recovery Plan** (Restore critical IT applications and systems) + **Business recovery** (Restore critical business processes)
 - Example: Outsourcing

Architect Responsibilities: Incident Management

- **Incident** : "unplanned event that causes a service disruption"
- **Goal**: "How to avoid incidents?" and "How to react to incidents quickly?"
- **How to avoid incidents?**
 - Monitoring and Alerting
 - Examples: Alerts : free disk space in Storage/Databases, CPU Utilization, Size of message queue
 - Programming Best Practices
 - Retry with exponential backoff
- **How to react to incidents quickly?**
 - Monitoring (Logging, Tracing, Debugging, Metrics, Dashboards)
 - Being Prepared (Game days)
- **Post-mortem:**
 - Goal: How to prevent a repeat of the incident?
 - Goal is NOT "Whom to blame?"



Google Cloud

Architect Responsibilities: Data Management

- **Managing data and its flow** is a very important responsibility
 - **How does data come in?** (stream or batch or transactional application)
 - If you have plans to move from batch to stream, prefer Cloud Dataflow
 - **What rate will we receive data?**
 - Plan for scaling storage (prefer auto scaling storage like Cloud Storage)
 - **What kind of data?** (archive or ..)
 - **How much data?** (GB, TB or PB)
 - **For how long?**
 - Cloud Storage Lifecycle Policies and Retention Policies
 - Table Expiration
 - **Who will have access?**
 - **How will you use the data?**
 - Ideally keep data and data processing system in the same zone (or atleast the same region)
 - If you are not planning to use the data for a long time, use Cloud Storage Archive Storage class

Other Important Google Cloud Services

Cloud Scheduler

In 28
Minutes

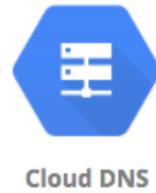
- **Fully managed, enterprise-grade scheduler**
 - Schedule all kinds of jobs
 - Batch, big data jobs, cloud infrastructure operations etc
 - Uses Unix cron format
- Integrates with App Engine, Cloud Pub/Sub, Cloud Logging and any HTTP endpoint
- Manage all your automation tasks from one place
- Provides **automated retries**
- **Use Case:** Schedule tasks across a fleet of Compute Engine instances
 - Use Cloud Scheduler for scheduling a message on Pub/Sub
 - Compute Engine instances can process messages from Pub/Sub
- **(REMEMBER)** Needs an App Engine App in the Project
 - **Earlier Alternative:** App Engine Cron Service

Cloud Emulators

- How do you **develop GCP applications in your local machine without connecting to GCP?**
 - Setup local development environment with Cloud Emulators
- **Supports emulation of:**
 - Cloud Bigtable
 - Cloud Datastore
 - Cloud Firestore
 - Cloud Pub Sub
 - Cloud Spanner
- You can **develop your applications locally using emulators!**

Cloud DNS

- What would be the **steps in setting up a website** with a domain name (for example, in28minutes.com)?
 - **Step I :** Buy the domain name in28minutes.com (Domain Registrar)
 - **Step II :** Setup your website content (Website Hosting)
 - **Step III :** Route requests to in28minutes.com to the my website host server (DNS)
- **Cloud DNS = Global Domain Name System (Step III)**
 - Setup your DNS routing for your website (in28minutes.com)
 - Route api.in28minutes.com to the IP address of api server
 - Route static.in28minutes.com to the IP address of http server
 - Route email (ranga@in28minutes.com) to the mail server(mail.in28minutes.com)
 - Public and private managed DNS zones (container for records)

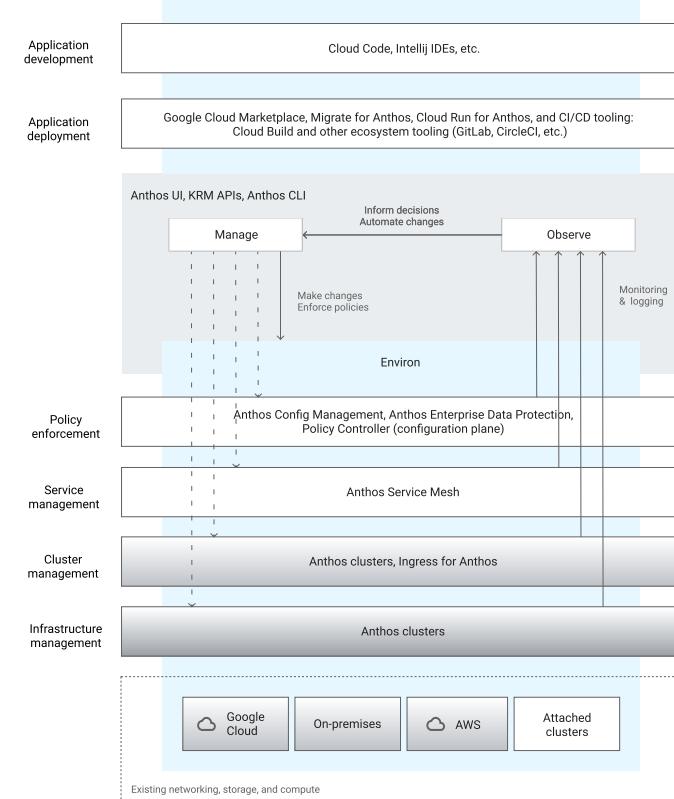


Pricing Calculator

- Estimating the cost of a Google Cloud solution is NOT easy
- You would need to take a number of factors into account
- How do you estimate the cost of your GCP solution?
 - Use Google Cloud Pricing Calculator
- Estimates for 40+ Services:
 - Compute Engine
 - Google Kubernetes Engine
 - Cloud Run
 - App Engine
 - Cloud Storage
 - etc
- (REMEMBER) These are Estimates! (NOT binding on GCP)

Anthos

- Run K8S clusters on cloud & on-premises
 - Multi-cluster management: Consistent managed K8S
 - Consistent development and operations experience
- Centralized config management (Git repo)
 - Logically group and normalize clusters as environs
 - Define policies - Kubernetes API, Access control
 - Deploy to clusters on new commits
 - Use Namespaces, labels, and annotations to decide which clusters to apply changes on
- Provides Service Mesh (based on Istio)
 - Sidecar to implement common microservice features
 - Authentication & authorization (service accounts)
 - Distributed Tracing, Automatic metrics, logs & dashboards
 - A/B testing, canary rollouts (even track SLIs & error budgets)
 - Cloud Logging & Cloud Monitoring Support



<https://cloud.google.com>

REST API Challenges

- Most applications today are built around REST API:
 - Resources (/todos, /todos/{id}, etc.)
 - Actions - HTTP Methods - GET, PUT, POST, DELETE etc.
- Management of REST API is not easy:
 - You've to take care of authentication and authorization
 - You've to be able to set limits (rate limiting, quotas) for your API consumers
 - You've to take care of implementing multiple versions of your API
 - You would want to implement monitoring, caching and a lot of other features..



Apigee API Management

- Design, secure, publish, analyze, monitor, monetize and scale APIs anywhere
 - On-premises, Google Cloud, or hybrid
- Manage Complete API life cycle
- Provides AI-powered API monitoring (Get actionable insights)
- Enable Caching with Cloud CDN
- Create Developer Portals :
 - Allow developers to easily explore the APIs, get API keys etc. Example: Apigee integrated portal
- Use Case: Abstraction layer on top of legacy services
- Use Case: Expose your assets (ML Models) as APIs



Machine Learning in Google Cloud

In 28
Minutes

Service	Discussion
Prebuilt APIs	Needs no in-house ML expertise. Easy to Use. Examples: Vision API, Video API, Natural Language API, Speech-to-Text API, Text-to-Speech API, and Translation API
Cloud AutoML	Build custom ML models with developers having limited ML expertise
AI Platform	Help data scientists build custom models (based on Tensorflow Enterprise) Serverless, scalable training and serving capabilities for custom ML models Take ML projects from concept to production quickly Explainable AI - interpret models with confidence
Data management	Cloud Storage and BigQuery BigQuery ML - Build ML models directly from data in BigQuery
Automation and instrumentation	AI Platform Pipelines & Cloud Composer - Orchestrate/automate data/ML pipelines Cloud Build and Container Registry - Build and deploy custom ML systems

Getting Started with Identity Platform

- **Identity Platform:** Customer identity and access management
- **What's the difference:** Cloud IAM vs Identity Platform
 - **Cloud IAM:** Employees and Partners Authorization
 - Control access to Google Cloud Resources
 - Member, Roles, Policy, Service Accounts
 - **Identity Platform:** Customer identity and access management (CIAM)
 - Authentication and Authorization for your applications and services
- **Identity Platform:** Key Features
 - Authentication & authorization for web & mobile apps (iOS, Android, ..)
 - Multiple authentication methods
 - SAML, OIDC, email/password, phone, social - Google/Facebook/Twitter/..
 - Features: User sign-up and sign-in, MFA etc.
 - An upgrade from Firebase Authentication Legacy
 - Integrates well with Identity-Aware Proxy

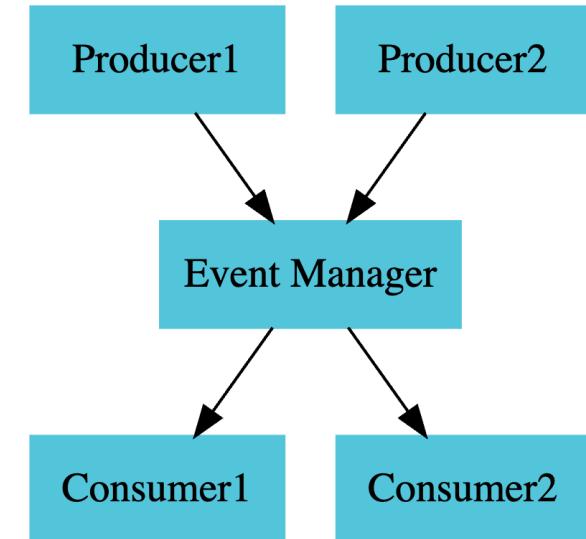


Cloud IAM vs Identity Platform - Scenarios

Scenario	Solution
An Application on a GCE VM needs access to cloud storage	Cloud IAM - Service Account
An enterprise user need access to upload objects to a Cloud Storage bucket	Cloud IAM
I want to manage end users for my application	Identity Platform
I want to enable "Login using facebook/twitter" for my application	Identity Platform
I want to create user sign-up and sign-in workflows for my application	Identity Platform

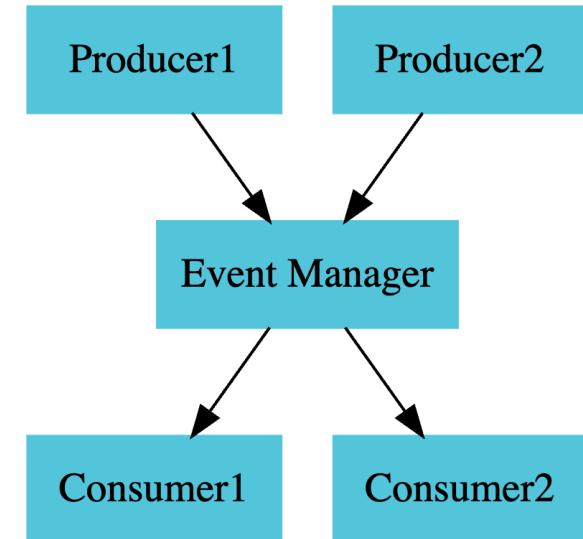
Getting Started with Event Driven Architecture

- Microservices calling each other => Tight Coupling
- **Event driven architectures:** Microservices reacting to changes in state (events)
 - **Example:**
 - 1: Order Service publishes an OrderReceived event
 - 2: Billing Service receives it and publishes an OrderBilled event
 - 3: Warehouse Service receives it & publishes an OrderReadyToShip event
 - 4: Shipping Service receives it and publishes an OrderShipped event
 - 5: Email Service receives it and sends an email to the user
 - **Advantages:**
 - **Loose Coupling:** Microservices do not know about each other
 - **Flexible Orchestration:** Same event can be processed by multiple services
 - **Resiliency:** Events can be easily retried in case of failures
 - **Asynchronous:** A microservice does not need to wait for the consumer to process the event



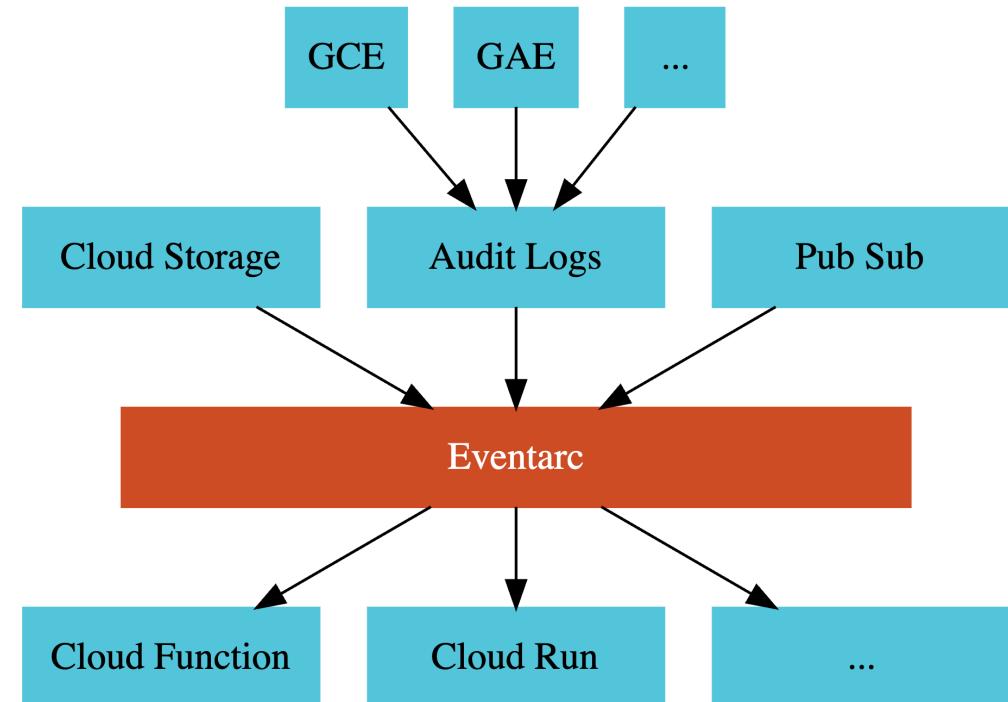
Getting Started with CloudEvents

- **CloudEvents:** Standard specification to describe events
 - Reference: <https://cloudevents.io>
 - **Challenge:** Different publishers use different formats for events
 - **Goal:** Describe event data in a standard way
 - A project under CNCF - Cloud Native Computing Foundation
 - **Advantages:**
 - **Consistency:** All events have the same structure
 - **Standard Libraries and Tooling:** Enables building common libraries and tooling across different types of infrastructure (AWS/Azure/Google Cloud/On-premise/..) and languages (Python, Go, NodeJS, Java, ..)
 - **Portability:** You are no longer tied to specific infrastructure or language
 - **Example Usecases:**
 - Get alerted when changes happen in storage buckets or database tables
 - Send one event to multiple consumers (fan out)



Getting Started with Eventarc

- **Eventarc:** Simplifies event driven architectures in Google Cloud
 - Adheres to the CloudEvents (cloudevents.io) specification
 - **Event provider:** Who can trigger events?
 - **Direct:** From Pub/Sub, Cloud Storage, Cloud Functions, Cloud IoT, Cloud Memorystore ..
 - **In-Direct:** From Cloud Audit Logs Entries
 - Huge variety of Google Cloud services. Ex: GCE, GAE, Artifact Registry, ...
 - **Event destination:** Who can process events?
 - Cloud Functions (2nd gen), Cloud Run & GKE services,..
 - (BACKGROUND) Uses Pub/Sub topics



Exploring Eventarc Event providers

```
gcloud eventarc triggers create my-pub-sub-trigger  
  --destination-run-service=$SERVICE_NAME --destination-run-region=$REGION  
  --event-filters="type=google.cloud.pubsub.topic.v1.messagePublished"

gcloud eventarc triggers create my-audit-log-trigger  
  --destination-run-service=$SERVICE_NAME --destination-run-region=$REGION  
  --event-filters="type=google.cloud.audit.log.v1.written"  
  --event-filters="serviceName=storage.googleapis.com"  
  --event-filters="methodName=storage.objects.create"
```

- **Two Types of Eventarc Event Providers:**

- **1: Directly from Google Cloud Services**
 - Pub/Sub, Cloud Storage, Cloud Functions, Cloud IoT, Cloud Memorystore ..
 - type: google.cloud.storage.object.v1.archived/deleted/finalized
 - type: google.cloud.pubsub.topic.v1.messagePublished
- **2: Indirectly from Cloud Audit Logs Entries**
 - type : google.cloud.audit.log.v1.written
 - serviceName:appengine.googleapis.com, methodName:google.appengine.v1.Applications.CreateApplication
 - serviceName:artifactregistry.googleapis.com, methodName:google.devtools.artifactregistry.v1.ArtifactRegistry.CreateRepository
 - serviceName:compute.googleapis.com, methodName:compute.instances.delete

Getting Started with Observability and OpenTelemetry

- **Observability:** "measure the internal state of a system by examining its outputs"
 - **Goal:** Proactively identify problems and fix them
 - **THREE PILLARS** of observability: logs, metrics and traces
 - Earlier we had different standards for logs, metrics and traces
 - We also had very different approaches across languages
- How about **ONE STANDARD ACROSS PLATFORMS?**
 - **OpenTelemetry:** Collection of technologies (tools, APIs, SDKs) to collect and export telemetry - metrics, traces, and logs (<https://opentelemetry.io>)
 - Open standard
 - A CNCF - Cloud Native Computing Foundation - Project (Kubernetes is another CNCF project)
 - Almost every cloud platform supports OpenTelemetry
 - Steps to use OpenTelemetry:
 - 1: Add OpenTelemetry libraries (for your specific language) to your project
 - 2: Instrument your code to export telemetry



Logging

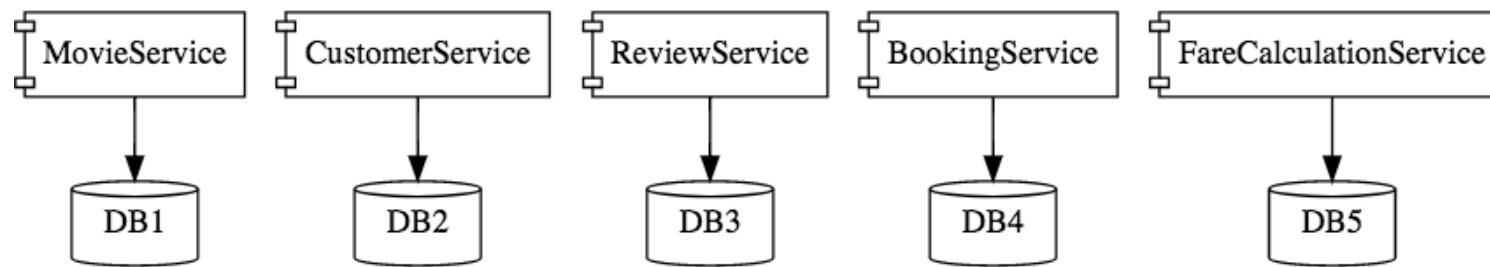


Trace



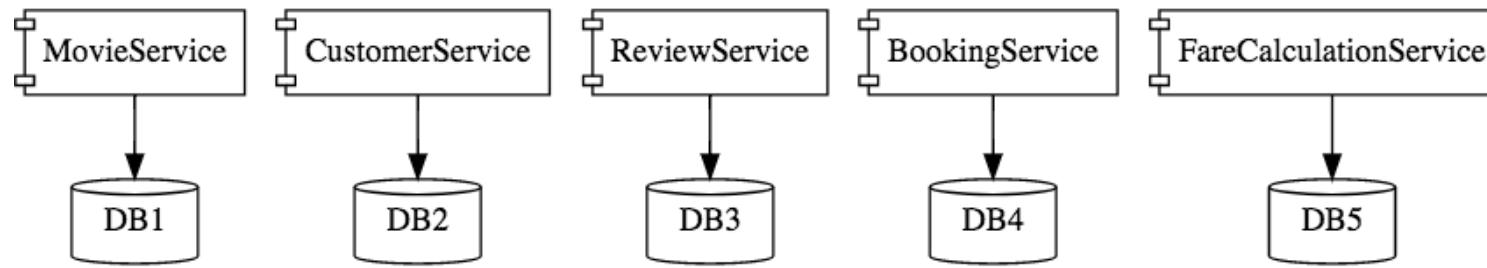
Monitoring

Using Service Directory



- **Service Discovery** - Help microservices find one another
- **Service Directory** - A single place to publish, discover, and connect services
- Your workloads can be running in:
 - Google Cloud
 - (Compute Engine VMs, Google Kubernetes Engine, ..)
 - On-prem
 - Other clouds - AWS, Azure, ..

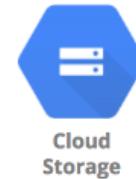
Using Service Directory - Features



- **Managed Service** (highly available and scalable)
- Register/resolve services using **DNS, HTTP, and gRPC**
- Service Directory **client libraries** are available for multiple languages (along with REST/RPC APIs)
 - You can use these libraries to register/resolve service location
- **Audit logging** (“Who did what, where, and when?” - Cloud Logging)
- **Request/response logs** (Cloud Logging)

Cloud Storage - Command Line - gcloud storage

- Earlier, **gsutil** was the recommended CLI for Cloud Storage
 - **GCLOUD STORAGE** is now the recommended CLI for Cloud Storage
 - Advantages:
 - Up to 94% faster storage transfers
 - Better parallel processing
 - Do NOT worry about options/parameters/flags
 - **gcloud storage** will decide the optimal storage transfer approach for you
 - Provides very simple to remember commands (consistent with gcloud):
 - **gcloud storage buckets create gs://BKT_NAME** (Create Cloud Storage bucket)
 - options: --default-encryption-key, --default-storage-class
 - **gcloud storage buckets delete gs://BKT_NAME**
 - **gcloud storage buckets list gs://B***
 - **gcloud storage buckets describe gs://BKT_NAME**
 - **gcloud storage buckets update gs://BKT_NAME**
 - options: --default-encryption-key, --default-storage-class, --[no-]versioning
 - If you have existing scripts that make use of gsutil commands AND
 - You do NOT want to change the scripts AND
 - You want the performance benefits offered by new features in **gcloud storage**
 - Check out **shim** (In boto configuration file, configure use_gcloud_storage=True under GSUtil section)



Google Cloud Architecture Framework

Google Cloud Architecture Framework

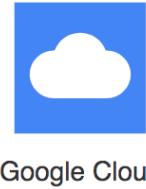
- Best practices and implementation recommendations to help you design your Google Cloud deployment
- First Step: Focus on designing robust, secure, and scalable systems
- **4 Principles:**
 - Operational excellence
 - Security, privacy, and compliance
 - Reliability
 - Performance and cost optimization



Google Cloud

Principle 1: Operational Excellence

- "Efficiently running, managing & monitoring systems that deliver business value"
- Strategies:
 - Automate build, test, and deploy
 - Monitor business objectives metrics
 - Conduct disaster recovery testing



Google Cloud

Operational Excellence - Best practices

- Increase software development and release velocity
 - **Release Engineering**
 - Frequent small releases
 - **Automation**
 - Static code analysis and security scans
 - Automate Testing (Unit, Integration, System, Load, Security Testing etc..)
 - Automate build and release pipeline
 - A/B or canary testing
 - **Services:**
 - **Cloud Source Repositories:** Fully-featured, private Git repository
 - Similar to Github
 - **Container Registry:** Store your Docker images
 - **Cloud Build:** Build deployable artifacts (jars or docker images) from your source code and configuration



Container Registry



Cloud Source Repositories

Operational Excellence - Best practices - 2

- Monitor system health and business health
 - Understand four golden signals
 - Latency - How fast are you responding to your users?
 - Traffic - How does the load on your system look?
 - Errors - Are any requests failing?
 - Saturation - What's the utilization levels of your resources?
 - Logging: Use Cloud Logging and export to Cloud Storage, BigQuery, and Pub/Sub depending on your needs
 - Metrics: Define and capture metrics SLIs, SLOs etc
 - Monitoring: Cloud Monitoring can aggregate metrics, logs, and events
 - Define alerts and custom metrics to identify problems and resolve them quickly
 - You can create visual dashboards as well
 - Generate actionable alerts
 - Key Services: Cloud Monitoring, Cloud Logging, Cloud Debugger, Error Reporting, Cloud Trace and Cloud Profiler



Monitoring



Logging



Debugger



Trace

Operational Excellence - Best practices - 3

- Design for disaster recovery
 - Create well-defined disaster recovery (DR) plan
 - Define Recovery time objective (RTO) and recovery point objective (RPO)
 - Consider everything in your DR plan: Applications (compute etc), Data(databases etc), Network infrastructure, Bandwidth, Facilities
 - Use features provided by Google Cloud:
 - Use Global network to build Redundancy
 - Managed services make Scalability easy
 - Regularly test your DR plan
 - Example features you can make use of:
 - Schedule Persistent Disk snapshots for your VMs and copy them across regions
 - Enable Live Migration to keep your VMs running even when there is software or hardware maintenance
 - Use Cloud DNS to switch from primary to backup



Google Cloud

Principle 2: Security, Privacy and Compliance

- Plan your security controls, privacy, and meet your compliance needs
- **Strategies**
 - Implement least privilege with identity and authorization controls
 - Build a layered security approach
 - Automate deployment of sensitive tasks
 - Implement security monitoring



Google Cloud

Security, Privacy and Compliance - Best practices

In 28
Minutes

- **Manage Authentication and Authorization**
 - **Follow Identity and Access Management (IAM) Best Practices**
 - Grant appropriate roles
 - Understand when to use service accounts
 - Treat each app component as a separate trust boundary
 - Create separate service account for each component/service with the minimum permissions needed
 - Use Organization Policy Service (what is allowed in your organization?)
 - Use Cloud Asset Inventory (Track your inventory)
 - Use Cloud Audit Logs to audit IAM policy changes and service accounts



Google Cloud

- **Implement Compute Security Controls**
 - Use Private IPs as much as possible
 - Create hardened VM images (OS + minimum software)
 - Use Shielded VMs to prevent remote attacks, privilege escalation, and malicious insiders
 - Enable node auto-upgrades for GKE clusters

Security, Privacy and Compliance - Best practices - 2

In 28
Minutes

- **Secure the network**

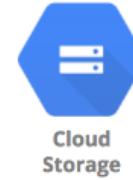
- Use a carefully designed custom VPC (DO NOT use the default VPC)
- Isolates workloads into individual projects by creating one VPC per project
- **Control ingress and egress network traffic using firewall rules**
- Run advanced security and traffic inspection tools
- Use **Security Command Center** to analyze the security of your infrastructure
- Use **Network Intelligence Center** to evaluate your network topology and architecture



Cloud Virtual Network

Security, Privacy and Compliance - Best practices - 3

- **Implement data security controls**
 - (Default) Google Cloud encrypts customer data stored at rest
 - Use customer managed or customer supplied keys based on your needs
 - **Use Object Versioning** in Cloud Storage for sensitive data
 - Use Object Lifecycle Management to reduce costs
 - Achieve compliance with retention policies using Bucket Lock
 - Use Signed URLs to give time-limited read or write access to an object
 - **Implement data security**
 - Use Cloud Data Loss Prevention (DLP) API to classify, mask, tokenize, and transform sensitive data (ensure data privacy for sensitive data elements like credit card numbers, names, SSNs, phone numbers etc..)
- **Audit your infrastructure with audit logs**
 - Use Cloud Audit Logs
 - Export to Cloud Storage, BigQuery, or Pub/Sub based on your needs
 - Enable Access Transparency logs (trace Google support team actions)



Principle 3: Reliability

- Most important feature of any application/service:

- Keys:

- Measurable reliability goals
 - Architect for scalability, high availability, and automated change management
 - Self-healing and observability are important
 - Quick recovery from failures using automation as much as possible

- Strategies:

- Define KPIs, SLOs and SLIs
 - Create redundancy (Prefer horizontal scalability)
 - Make incremental changes
 - Include rollback capability
 - Instrument systems for observability
 - Automate detection of failure
 - Document and automate emergency responses
 - Test failure recovery
 - Reduce toil (Eliminate manual work)



Google Cloud

Reliability - Best practices

- **Define reliability goals:** Service Level Objectives & Error Budgets
 - Clearly define SLIs, SLOs, SLAs and Error budgets
- **Build observability** into your infrastructure and applications
 - Monitoring, logging, tracing, profiling, debugging etc.
- **Design for scale and high availability**
 - Design a multi-region architecture with failover
 - Eliminate scalability bottlenecks
 - Prefer horizontal scalability
 - Degrade service levels gracefully
 - Static web pages when dynamic web site is down
 - Temporarily disabling data updates
 - Implement exponential backoff with jitter
 - Exponentially increasing wait time before retry
 - Predict peak traffic events and plan for them



Google Cloud

Reliability - Best practices - 2

- **Build flexible and automated deployment capabilities**
 - Ensure that every change can be rolled back
 - Spread out traffic for timed promotions and launches
 - Implement progressive rollouts with canary testing
 - Automate build, test, and deploy
- **Build efficient alerting**
- **Build a collaborative process for incident management**
 - Reduce Mean Time to Detect (MTTD): Alert teams at the right time
 - Reduce Mean Time to Mitigate (MTTM) and Mean Time to Recovery (MTTR): Properly documented and well-exercised incident management plan
 - Increase Mean Time Between Failures (MTBF): Build reliable systems
 - Blameless postmortem culture



Google Cloud

Principle 4: Performance & Cost Optimization

- **Strategies**

- Evaluate performance requirements
- Use scalable design patterns
- Identify and implement cost-saving approaches



Google Cloud

Performance and Cost Optimization - Best practices

- **Use autoscaling and data processing**
 - Autoscale Compute Engine VMs with Managed instance groups (MIGs) - Uses an instance template
 - Enable Cluster autoscaler and Pod autoscaling (Horizontal Pod Autoscaler) on Google Kubernetes Engine Clusters
 - Try Serverless options: Cloud Run, App Engine, Cloud Functions, Dataproc and Dataflow
 - Use Google Cloud Load Balancers to provide a global endpoint.
- **Use GPUs and TPUs to increase performance**
 - Use GPUs to accelerate machine learning and data processing workloads
 - Use TPUs for massive matrix operations performed in your machine learning workloads



Compute Engine



Cloud Functions



App Engine



Container Engine

Performance and Cost Optimization - Best practices - 2

- **Identify apps to tune**

- Use Cloud Trace, Cloud Debugger, and Cloud Profiler to gain insights into your apps
- Instrument apps to identify inter-service communication latencies or identify bottlenecks



BigQuery

- **Analyze your costs and optimize**

- Export Billing to BigQuery to analyze your billing data
 - Use Google Data Studio to visualize
- Understand Sustained use discounts
- Make use of Committed use discounts for predictable long term workloads
- Use Preemptible VMs for non critical fault tolerant workloads
- Use Cloud Storage Object Lifecycle Management to reduce storage costs



Debugger



Trace

Getting Started with AI and ML

Getting Started with AI and ML

- Please Download Updated Slides:
 - AI and ML - [*https://www.in28minutes.com/downloads/99-for-beginners/ai-with-google-cloud.pdf*](https://www.in28minutes.com/downloads/99-for-beginners/ai-with-google-cloud.pdf)
 - Generative AI Based Solutions - [*https://www.in28minutes.com/downloads/99-for-beginners/machine-learning-google-cloud-020-generative-ai-based-solutions.pdf*](https://www.in28minutes.com/downloads/99-for-beginners/machine-learning-google-cloud-020-generative-ai-based-solutions.pdf)

Case Studies

Case Studies

- Please Download Updated Slides for Case Studies -
<https://www.in28minutes.com/downloads/99-for-beginners/architecture-google-cloud-030-case-studies.pdf>

Get Ready

Google Cloud Architecture Resources

Title	Link
Cloud Architecture Center	https://cloud.google.com/architecture/
Google Cloud Solutions	https://cloud.google.com/solutions/
Best Practices for Enterprise Applications	https://cloud.google.com/docs/enterprise/best-practices-for-enterprise-organizations
Building Scalable and Resilient Web Applications on Google Cloud Platform	https://cloud.google.com/solutions/scalable-and-resilient-apps
Disaster recovery planning guide	https://cloud.google.com/solutions/dr-scenarios-planning-guide

Certification Resources

Title	Link
Home Page	https://cloud.google.com/certification/cloud-architect
Exam Guide	https://cloud.google.com/certification/guides/professional-cloud-architect
Case Studies	https://cloud.google.com/certification/guides/professional-cloud-architect
Sample Questions	https://cloud.google.com/certification/sample-questions/cloud-architect
Registering For Exam	https://support.google.com/cloud-certification/#topic=9433215

Certification Exam

- **50 questions and Two hours**
 - No penalty for wrong answers
- **Questions:**
 - Type 1 : Multiple Choice - 4 options and 1 right answer
 - Type 2 : Multiple Select - 5 options and 2 right answers
- Result immediately shown after exam completion
 - Email (a couple of days later)
- **My Recommendations**
 - Read the **entire question**
 - Identify and write down the **key parts of the question**
 - Focus on the constraints in the question:
 - Example: Most cost-effective way - Prefer regional to multi-regional if it satisfies all requirements
 - **TIP: Answer by Elimination!**
 - **Flag questions for future consideration (Review before final submission)**

You are all set!

Let's clap for you!

- You have a lot of patience! **Congratulations**
- You have put your best foot forward to be an Google Cloud Certified Professional Cloud Architect
- Make sure you prepare well
- Good Luck!

Do Not Forget!

- Recommend the course to your friends!
 - Do not forget to review!
- **Your Success = My Success**
 - Share your success story with me on LinkedIn (Ranga Karanam)
 - Share your success story and lessons learnt in Q&A with other learners!

What Next?

FASTEST ROADMAPS

in28minutes.com

