INDEX

Name _____ Std _____ Sec _____

Roll No. _____ Subject _____ School/College _____

School/College Tel. No. _____ Parents Tel. No. _____

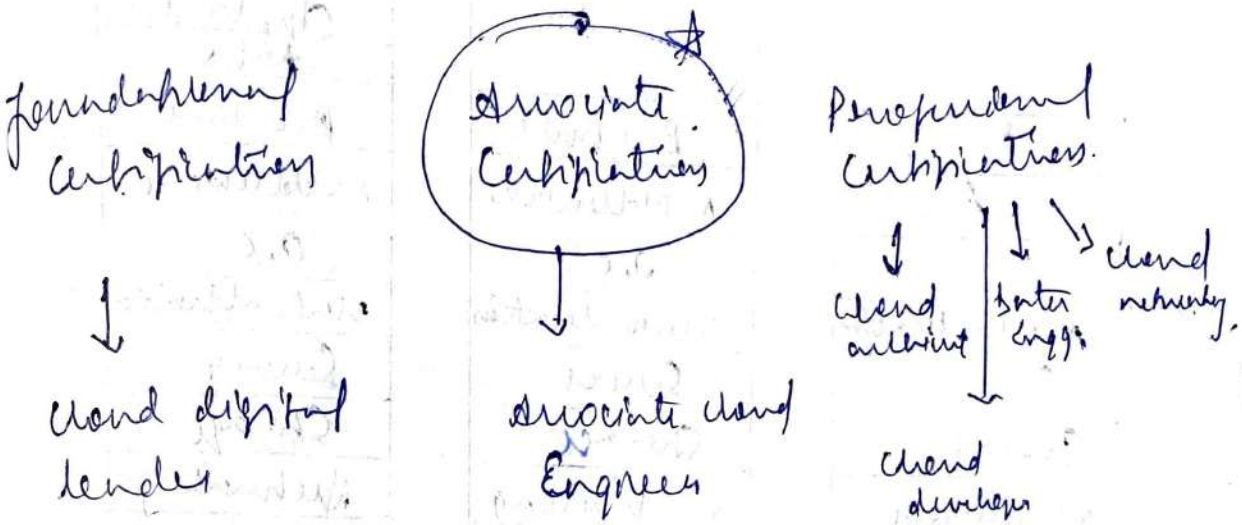
Sl. No.	Date	Title	Page No.	Teacher Sign / Remarks
		Teacher(Incharge) : BHUPINDER RAJPUT		
		TECHNICAL GUFTGU		
		Notes : Mohammad Sarfaraz. S		

Lec-01

17/1/23

GCP – Associate Cloud Engineer

Certification path.



Lec-02 Basics of cloud computing

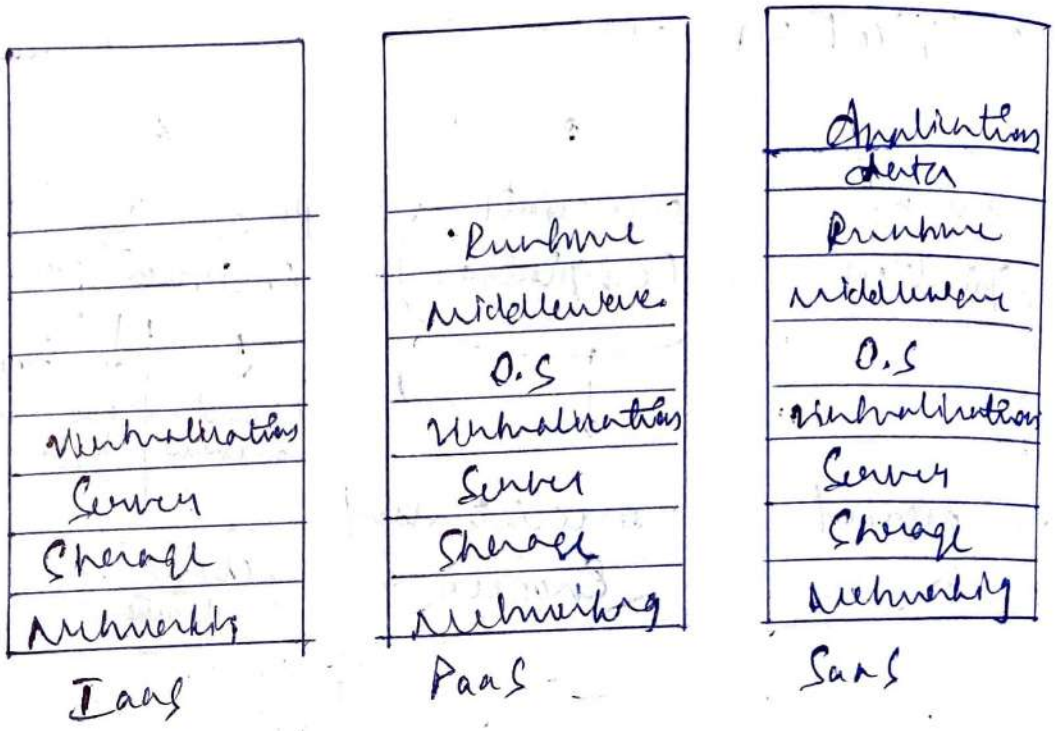
Cloud computing is the delivery of computing services – servers, storage, databases, networking, tools & software over the internet.

Characteristics of cloud computing

1. On-demand self service
2. Broad network access
3. Resource pooling
4. Rapid Elasticity
5. Measured service

Lec-03

What is IaaS, PaaS & SaaS



Cloud Deployment Models / Types of cloud

1. Public cloud
2. Private cloud
3. Hybrid cloud

Lec-04

GCP Console Overview

login

↓

GCP console

Lec-04 Concept of Zones, Regions & Multi-regions

Regions are independent geographic area that consists of collection of Zones. They affect pricing, reliability, networking & performance

29 Regions

Zone -> A zone is a deployment area of google cloud Resources within a region. Zones should be considered a single failure domain within a region. To deploy fault tolerant applications with high availability & help protect against unexpected failures, deploy your applications across multiple zones in a region.

Regional Resource -> app engine

88 Zones

Multi-regional services are designed to be able to function following the loss of a single region.

Multi-regional Resources - BigQuery, Bigtable, Cloud storage, Spanner, Datastore, firestore, Artifact Registry

Note: If a single region fails, only customers solely in that region are impacted, customers who have multi-region products are not impacted.

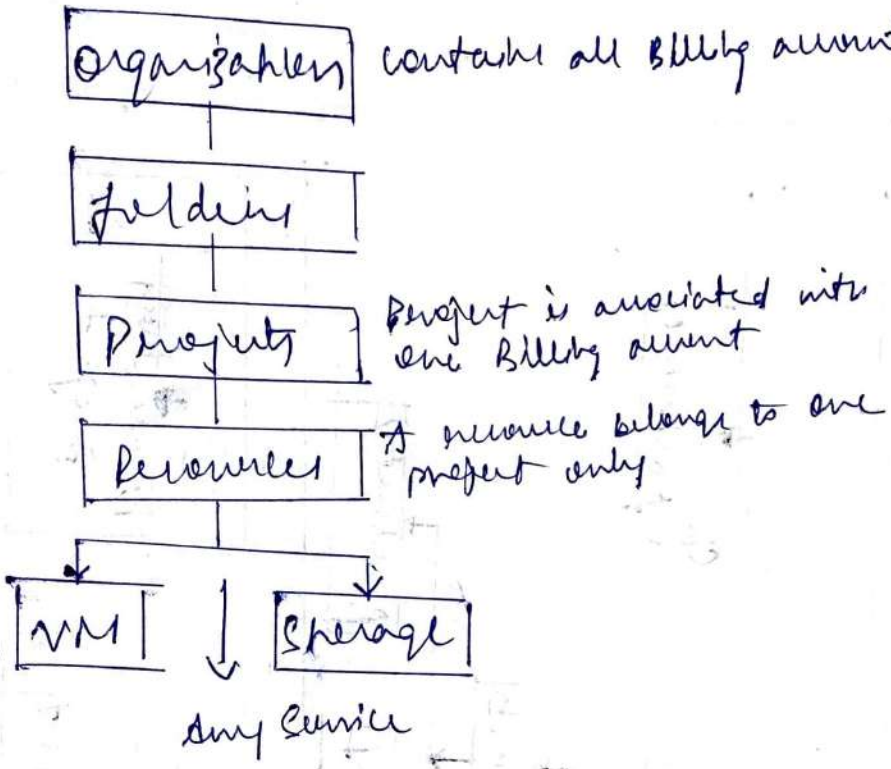
The fully qualified name for a zone is made up of <region> <zone>

for eg → zone-a in region us-central1 is us-central1-a

<u>When selecting zones generally, here are something to keep in mind.</u>

1. Communications within & across regions will incur diff. cost.
2. Design important systems with redundancy across multiple zones or regions → At some point in time, your instances might experience an unexpected failure. To mitigate the effect of these possible events, you should duplicate important systems in multiple zones & regions

GCP Resource Hierarchy & Concept of Project in Detail



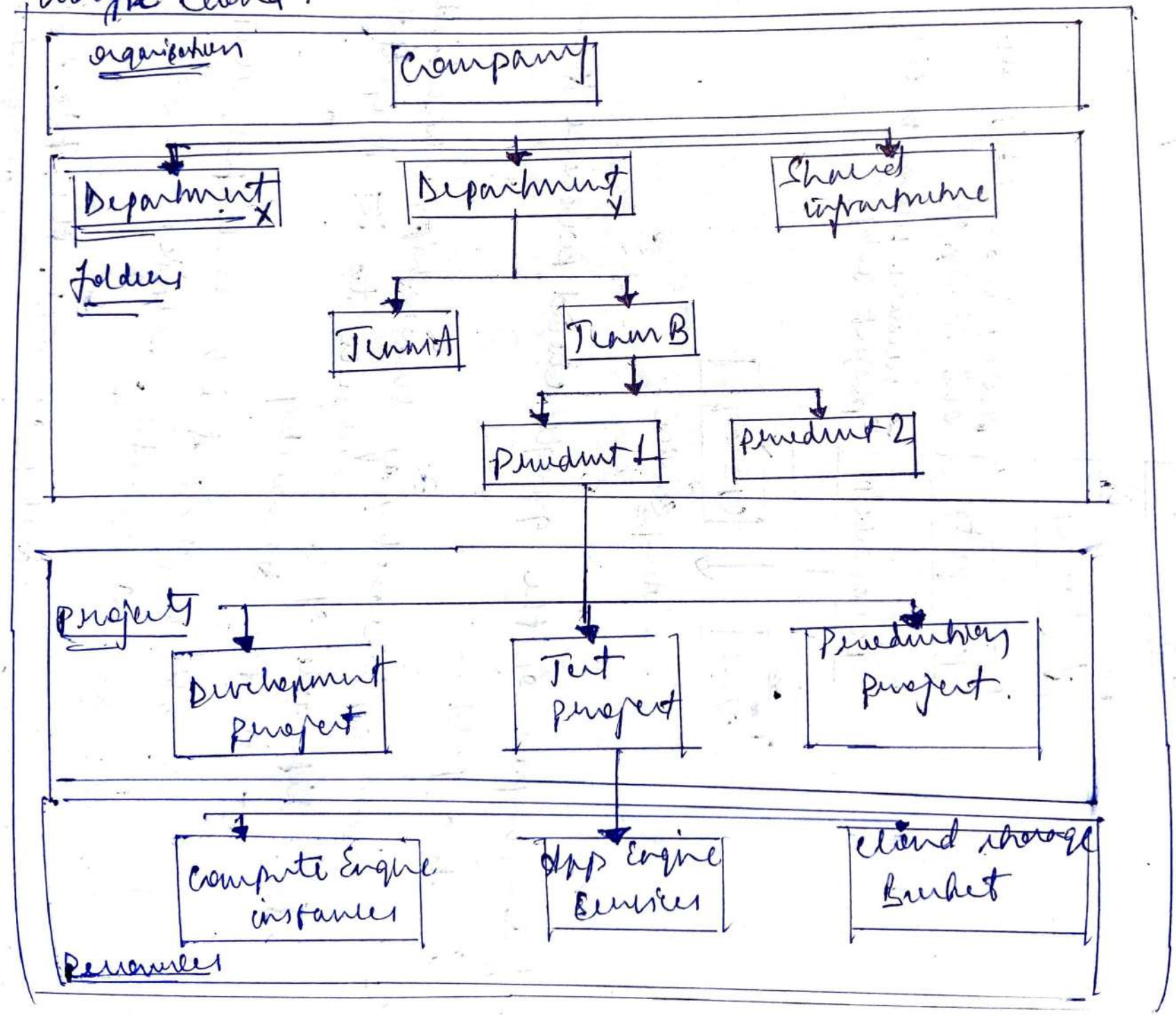
The purpose of the google cloud Resource Hierarchy is two fold :-

① Provide a hierarchy of ownership, which binds the lifecycle of a resource to its immediate parent in the Hierarchy.

② Provide attach points & inheritance for access control and organizations policies.

GCP Resource hierarchy resembles the file system found in traditional O.S as a way of organizing & managing entities hierarchially. Each resource has exactly one parent.

Google Cloud:



Benefits of the organization Resource

with an organization Resource, projects belongs to your organization instead of the employee who created the project. This means that the projects are no longer deleted when an employee leaves the company.

-> You can grant roles at the organization level which are inherited by all projects and folders under the organization resource.

for eg:
You can grant the network admin role to your networking team at the organization level, allowing them to manage all the networks in all projects in your company, instead of granting them the role for all individual projects.

Folders

-> Folders resources provides an additional grouping mechanism & isolation boundaries b/w projects.

-> Folders can contain other folders

-> It is optional

☆☆ Projects

- → All GCP resources you use are associated to one specific project
- → You can track resources & quota usage
- → Enable Billing & set Budget
- → Manage permissions & credentials
- → Project is a global entity
- → Enable Services & API
- → Equivalent to "account" in AWS & "Subscription" in Microsoft Azure.

☆☆ Project has three components

- Project ID
- Project Name
- Project Number (12 digit)

Project ID	Globally unique	Chosen by you	Immutable (can be only when creating)
Project Name	Uniqueness not req.	Chosen by you	Mutable (can be changed)
Project Number	Globally unique	Assigned by GCP	Immutable

Lec-06 How to Create GCP free Tier Account

Lec-07 Google cloud compute Engine Part-1
or virtual Machines:

Compute Engine is a customizable compute service that lets you create and run virtual machines on Google's Infrastructure.

There are two types of machine in GCP :-

Custom machine

you can take vCPU and memory as per your need, and create your own custom machine.

Pre-defined machines

Predefined machine types are pre-built and ready to go configurations of VMs with specific amounts of vCPU and memory to start running app quickly.

→ Google compute engine is comes under IaaS.
→ A virtual machine is like a physical hardware server, but running in the cloud.
→ GCP isolates resources and sells them on a per server basis with different combinations.
→ Google provides VM of all sizes → micro to 160vCPU and 3.75 TB memory. (RAM)
→ You can attach local SSD upto 3TB and persistent disk upto size of 64 TB, that too encrypted as well. (DAS - Direct attach storage; NAS - network attached storage)

→ Google cloud uses open source KVM Hypervisor at Backend.

→ Fast Booting time amongst all cloud providers.

→ All vCPU, GCP & GB of memory are charged a minimum of 1 min. After 1 minute, instances are charged in 1 sec increments.

Compute (Server Based) (IaaS)

- Compute Engine
- EC2's Engine
- VMware Engine

Serverless (PaaS)

- App Engine
- Cloud Run
- Cloud functions

Note

GCP → KVM Hypervisor (Kernel based VM)

AWS → Nitro

Azure → hyper-V

Types of Predefined VM/Machine families.

1. General purpose
2. Compute Optimized
3. Memory optimized
4. Accelerator optimized / GPU
5. Shared core
6. Sole Tenant nodes

→ A machine type is a set of virtualized hardware resources available to a virtual machine including the system memory size, vCPU & persistent disk limits.

→ In compute Engine, machine types are grouped and put inside families on their behavior & workload.

Compute Engine has 3 terms:

Machine family – Set of processors and hardware configuration.

Series → Machine families are further classified by series and generation.

Machine type → Every machine series has pre-defined machine types that provide a set of resource

Customers using GCE → Paypal, Twitter, P&G, Airbus, Etsy, SKY

General Purpose workloads			Optimized workloads		
Cost-optimized	Balanced	Scale-out optimized	Memory optimized	Compute optimized	Accelerated optimized
EC2	N1, N2, N2D D -> AMD	T2U, T2D	M1, M2.	C2, C2D	A2
Day to day computing at a lower cost	Balanced price / performance across a wide range of VM shapes	Best performance & cost for scale out workloads	Ultra high memory workloads. offers highest mem. capacity across VM families	Ultra high performance for compute intensive workload	Optimized for high performance computing workloads. Best for parallel computing workload
-> Web serving App serving -> Back office apps -> Small & mid databases -> Microservices -> Virtual desktops	-> Web serving -> App serving -> Back office apps -> Med. large databases -> Cache -> Media (streaming)	-> Scale-out workloads -> Web serving -> Containerized microservices -> Media Transcoding -> Large scale Java app.	-> Medium-large in-memory data - based such as SAP HANA -> In memory database & in mem. analytics -> Microsoft SQL Server & similar databases	-> Compute bound workloads -> High performance web serving -> Gaming -> AI serving -> High performance computing -> Media Transcoding	-> CUDA enabled ML Training & Inference -> HPC -> massively parallelized computations

Shared-core Machines

This type of VMs intends to share the physical core. Shared core machine types are a cost effective option that work well with small or batch workloads that only needs to run for a short time.

-> It uses partial vCPU that runs on one hyper thread of the host CPU running your instance.

Sole-Tenant Nodes:

Sole tenant nodes are the servers for compute engine that are <u>dedicated</u> for the use on priority. The purpose of these nodes is to serve the deployment for BYOL (Bring your own License) applications.

-> The sole tenant nodes provides you access to similar VM configurations & machine types just like regular compute instances.

Customs machine type

If none of the pre-defined machine types matches your needs, you can independently specify the number of CPUs & the amount of memory for your instance.

-> The memory per vCPU of a customs machine type must be b/w 0.9GB, 3.75 & 6.5GB per vCPU.

-> Total memory of a custom machine type must be a multiple of 256MB.

-> Above 1 i.e the instance vCPU count must be even. Such as 2, 4, 6, 8, 10 & so on

Lec - 08

Google cloud compute Engine Pricing & VM Disk Types

Compute Engine Pricing

1. Sustained use discount (AWS -> on demand)
2. Committed use Discount → One year (Reserved) → Three year → Sole-tenant node (dedicated host)
3. Pre-emptible VM (or) Spot VM

1. Sustained use discounts:

-> Sustained use discounts are applied automatically to usage within a project seperately per each region. So, there is no action required on your part to enable these discounts.

-> If you run a VM for more than 25% of a month, you are eligible for discount b/w 20-30%.

-> Discount is given on vCPU & memory usage.

-> for general-purpose N2 & N2D predefined & custom machine types & for compute optimized machine types, you can receive a discount of upto 20%.

-> for general-purpose N1 predefined & custom machine-types, sole-tenant nodes & GPU, you can get a discount of upto 30%.

Limitations

-> Sustained use discounts automatically apply to VM created by GCE or GKE

-> Sustained use discounts do not apply to VMs created using the app-engine environment & Dataflow.

-> Sustained use discounts do not apply to B2, A2 or Tau T2D machine types

2 Committed use discount:

Ideal for workload with predictable resource need

-> When you purchase a committed use contract, you purchase compute engine resources - such as vCPUs, memory, GPUs, local SSDs and Sole-tenant nodes - at a discounted price in return for committing to paying for those resources for 1 year to 3 years.

-> Discount is upto 57% to 70% ✓

-> After purchase, you are billed monthly whether you are using it or not.

-> No upfront cost is required.

How it works?

for example, if you purchase a commitment for 8 cores and you runs 24 cores during the month, you will receive committed use discount for 8 cores. The other 16 cores will be billed at standard, non-committed use rates.

Note

Keep in mind that after you purchase a set amount of commitments, you are billed for these commitments monthly, even if you don't use them.

→ You can't combine committed use discounts & sustained use discounts for the same resources.

Committed use discounts (Limitations)

→ Projects that are in the free-tier paired with free-tier credits & projects that do not have any payment history do not qualify for committed use discount.

→ Committed use discounts apply to usage in the project or cloud Billing account in which those discounts are purchased.

→ Committed use discounts do not apply to pre-emptible VM instances.

→ If a reservation is attached to a committed use discount, the reservation can't be deleted for the duration of the commitment.

3. **Preemptible instances (PVM)** (only test / never for prod)

Preemptible instances are available at much lower price – 60-91% discount, compared to the price of std. VMs.

However google cloud might stop these instances if it needs to re-claim the compute capacity for allocation to other VMs.

→ Compute engine always stops pre-emptible instances after they run for 24 hrs.

→ The GCP free-tier credits for compute engine do not apply to pre-emptible instances.

★★→ Pre-emptible instances cannot live migrate to a regular VM instances or vice-versa.

→ Not covered under SLA (Service level agreement)

→ GCP gives only 30 sec. time before reclaiming it.

→ Best suited for batch processing jobs of short duration.

Storage options for Compute Engine

1 * **Zonal Persistent Disk**
(Block storage type, NAS, Bootable)

Server storage
- a. Standard PD — HDD — min 10GB [4th] Performance
- b. SSD PD — min 10GB [2nd]
- c. Extreme PD → min 500GB - SSD [1st]
- d. Balanced PD — SSD — min 10GB [3rd]

** [can be Zonal / Regional → if Regional we get replica]

2. **Regional Persistent Disk** (non Bootable)
Data is Replicated between two zones in the same Region. Block storage type. Bootable disk.

3. **Local SSD** (non Bootable) (*SWS → Instance store, *Sme - Temp disk)
physically attach to Host server, non-persistent, Direct attach storage, Not Bootable.

4. **Cloud Storage Bucket** (non Bootable)
- a. Std. Storage
- b. Nearline
- c. Cold line
- d. Archive

5. **File store** (non Bootable)
Shared storage for instance

☆☆☆

Note! Increase but not shrink the disk size

-> Number of disk is dependent on vCPU

-> Size of disk inc. -> IOPS inc.

-> Can't take snapshot of local disk.

Lec-09 GCP console overview demo

Lec-10 How to create virtual machine

Login to GCP account -> [Compute Engine]
-> VM instances

(2) [Create instance] ->

Name -> xyz
Labels -> key-value
Region -> US-west4 Zone -> US-west4-b

[Machine family]

a. General Purpose -> Series -> E2, N2, N1, N2D, T2D

b. Compute optimized

c. Memory optimized

d. GPU

[Machine type]
-> Custom **
-> Shared core
-> Standard
-> High memory, High CPU

[Boot Disk]
-> Public images -> O.S, version, disk type
-> Custom images
-> Snapshots
-> Existing Disk

BPD
EPD
Std. PD
SSD P.D

Identity & API access

Service account

Firewall

- [x] Allow HTTP traffic
- [x] Allow HTTPS traffic

Networking, Disks, Security, Management, Sole tenancy

CREATE

Equivalent command line

within 30 sec

Lec 11 How to install apache on Linux Machine

① go to GCP console -> Compute -> VM instances
-> create instance (Debian 11)

② Take SSH of Linux machine
Run following commands

-> sudo su
-> apt update -y
-> apt install apache2 -y
-> apache2 -version
-> systemctl status apache2
-> hostname -I

copy public IP of Linux machine from GCP console & put in Browser.

To create a custom webpage or modifying existing

-> vi /var/www/html/index.html
i -> new edit

esc -> :wq

Now check in google chrome Browser, you will get edited page

Lec-12 Create & manage VM instance using cloudshell

(1) Login to GCP console -> Select your project -> Activate cloudshell

-> gcloud compute instances create myvm01 --machine-type n1-standard-2 --zone asia-south1-a

It will ask to authorize the credentials, click on authorize

-> gcloud compute ssh myvm01 --zone=asia-south1-a -> exit

-> gcloud compute images list

-> gcloud compute instances create myvm02 --image-family ubuntu-2004-lts --image-project ubuntu-os-cloud --zone=us-central1-a

-> gcloud compute instances describe myvm02

-> gcloud compute instances stop myvm01

-> gcloud compute instances delete myvm01

-> gcloud compute instances list

Lec-13

Changing the Machine type of a VM instance

-> If your existing machine is not a good fit for the workloads you run on your VM, change the machine type of that VM. (vertical scaling)

-> If your VM does not have a local SSD and is not part of a Managed Instance Group (MIG), you can change the machine type of your VM after stopping it.

-> If your VM has a local SSD, you cannot stop the VM to change its machine type unless you force it to stop, which causes you lose all of the data on the local SSD.

-> Changing a machine type might also affect the sustained use discounts for that VM.

-> You can only change the machine type of a stopped VM. You cannot change the machine type of a running VM.

** -> Take snapshot before updating size

-> For VMs that do not have a local SSD and are not part of a MIG, you can change the machine type without affecting the following resources:-

* (a) The VM's SSH keys
* (1) The VM configurations, such as metadata

(c) Persistent disk data including installed applications and app data.

Steps

(1) go to google console -> Compute -> VM instances -> Select the VM -> Create one VM

(2) now to modify this VM, stop it

(3) go to edit -> machine configurations -> Select the machine type you want -> Save

Refresh your console -> you will get updated VM.

Demo using Cloudshell

(1) stop the VM first

-> gcloud compute instances stop cmyVM01

(2) change the machine type

-> gcloud compute instances set-machine-type myVM01 --machine-type e2-standard-2

for custom machine type, you can use

-> gcloud compute instances set-machine-type

myvm01 --machine-type e2-custom-4-2048

vcpu memory

2x1024 -> 1GiB

Lec-14

1) Demo - Resize your Persistent disk of VM

*-> You can increase the size of your Persistent disk when your VM instances requires additional storage space or higher performance limits.

*-> You can increase the size of P.D but you cannot reduce

-> VM using public images automatically resize the boot disk. In all other cases, you need to manually resize the boot disk or non-boot disk.

-> No need to stop the VM, to resize

Resize PD using Console

① login to GCP console -> Compute Engine -> Create one VM with 10GB size of Boot volume

Now, to modify this volume,

go to disk -> Select disk -> edit -> modify size = 30GB -> Save

Now, check on VM Detail Page, it shows 30GB storage.

Cloudshell Commands

-> gcloud compute disks resize diskname --zone
-- size diksize

Now, to **Resize in Linux VM** ☆

-> Take SSH
-> sudo su (root user)
-> df -Th (volume types) -> boot (sda) / -> non boot
-> lsblk
-> Parted /dev/sda
-> resizepart
-> 1 (Partition number)
-> yes
97 G ? End -> 100%

-> quit
Now check again,

-> lsblk

For windows after adding req. volume

-> Disk management
-> Action -> Refresh (unallocated disk is seen)
-> click on c: drive -> extend volume

Now check again

Lec-15 What is Public Image, Custom Image and Machine Image

Images

-> you can use O.S images to create boot disk for your instance. You can use one of the following image type -

Public Image -> Public image are provided and maintained by google, Open source communities & third party vendors.
By default, all google cloud projects have access to these images & can use them to create instances.

Custom Image -> custom images are available only to your cloud Project. You can create a custom image from Boot disk & other images. Then, use the custom image to create an Instance

-> custom images are free of cost but do incur an image storage charge.

you can use custom images for following purposes :-

-> Import a virtual disk to compute Engine from your on-premises Environment or from VMs that are running on your local workstation or on another cloud platform.

-> Create an image from boot disk of your existing VM. Then use that image to create new boot disks for your instance.

-> Copy one image to another image by using either the gcloud CLI or the API. You can also create an image from a custom image in a different project

Machine Image

-> A machine image is a compute engine resource that stores all the configuration, metadata, permissions and data from multiple disks of a VM instance. You can use a machine image in many system maintenance, backup and Recovery and instance cloning scenarios.

Machine Image are the most ideal resources for the following cases:

* (1) Multiple disk Backups
* (2) Instance Cloning

-> Machine Images are suitable for creating Backup of all disks that are attached to the VM instances.

-> Machine Image can be used to clone instances. You can create multiple copies of an instance.

Information stored in a Machine Image:

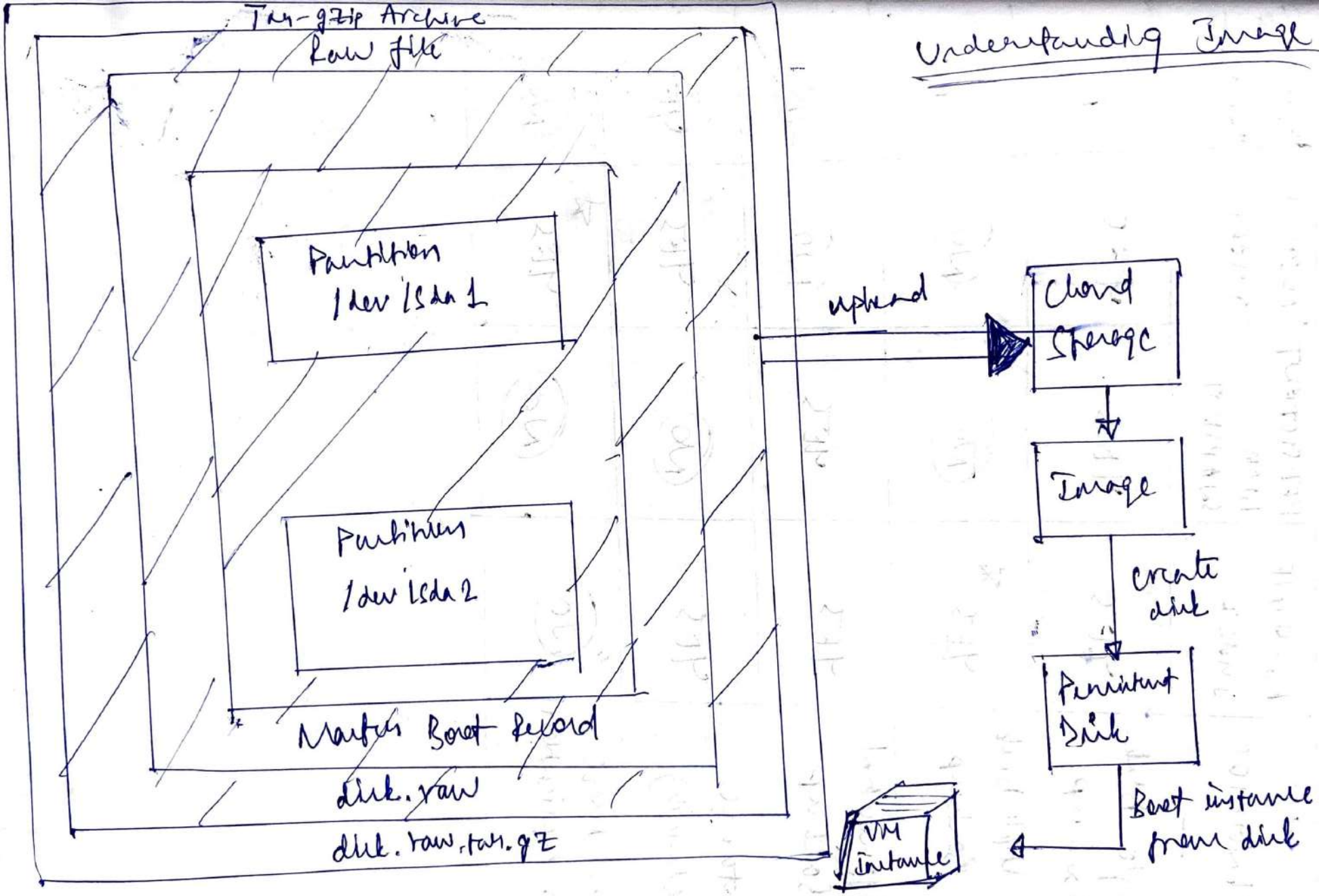
-> Description, Machine type, labels, Network tags, Maintenance policy & Instance metadata.

-> Machine Image stored in cloud storage bucket.

-> Regardless of where the machine image is stored, you can still access the machine image globally.

-> Machine Image is unchangeable but you can over-ride few properties.

Understanding Image



When to use which image?

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	YES	NO	YES	YES
Base image for Replication	NO	NO	YES ☆	NO

Lec-16 How to create Custom Image

① go to GCP Console -> Compute -> VM instance -> Create instance

② Go to Compute -> VM instances -> Image -> create Image

Name - instance-1
Source - disk
Location - Multi-regional [asia]

☑ Keep instance running

-> [Create]

③ go to compute -> create instance
Name, region - asia-south1
Boot disk -> Custom images -> Select your image -> create instance

Using Cloudshell

Disable auto-delete option for the disk
-> gcloud compute instances set-disk-auto-delete VM-name --no-auto-delete --disk=sourcedisk name

-> gcloud compute images create image-name --source-disk = source-disk --source-disk-zone = zone --storage-location = US --force

→ gcloud compute images describe image-name.

→ gcloud compute images delete image-name

Lec-17 Demo - Creating a Machine image from a VM

① go to GCP console -> Create one VM instance1

② Now go to 'Machine Images' -> Create Machine Image

Name -> mymachine-img
Source-VM instance -> instance-1
Location -> Regional

-> [Create]

③ Now go to the Machine image -> create instance

④ Test

[Using cloudshell]

-> gcloud compute machine-images create machine-image-name --source-instance = Source VM-name --source-instance-zone = us-central-1a

Now create instance from Machine image

-> gcloud beta compute instances create VM-name --zone = zonename --source-machine-image = machine-image-name --machine-type = e2-std-2 (optional)

Lec-18 Cloud storage & its Types

Storage Types in Google Cloud:

1. Cloud Storage (S3)
2. Persistent Disk (EBS)
3. Filestore (EFS).

Google Cloud Database Store:

1. Bigtable
2. Firestore (Backend as a service)
3. Cloud SQL
4. Cloud Datastore
5. Cloud Spanner
6. Memory Store

→ Cloud storage is a service for storing your objects in Google Cloud. An object is an immutable piece of data consisting a file of any size and format.

→ You store objects in containers called buckets. (Folder/Box)

→ Ways to access / interact with cloud storage

(a) Console

(b) gsutil → cloudshell

(c) Client libraries

(d) Rest APIs

Cloud Storage classes :

(1) Standard storage → Hot storage

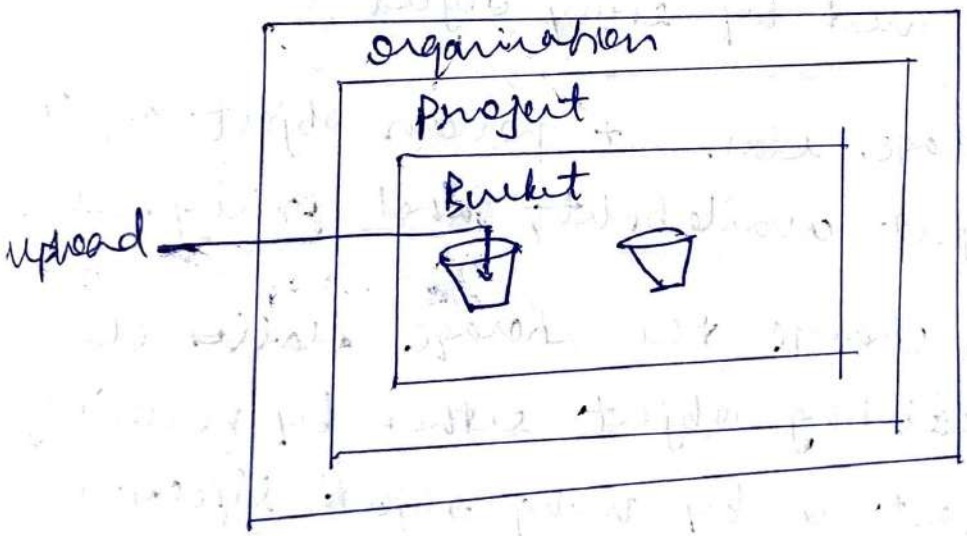
(2) Nearline storage (once in a month) → (4) Warm storage

(3) Coldline storage (once in 3-4 months) → (2) Cold storage

(4) Archive storage → (1) Cold storage

Data retrieval ↑ ↓

GB size Reduce ↓



Location type

↳ Regional

↳ Dual Region

↳ Multi-region

US → region 1, region 2

US, asia

Lec-19 Cloud Storage types and its characteristics

Storage classes

1. Standard storage → costliest
2. Nearline storage
3. Coldline storage
4. Archive storage → cheapest

→ A storage class is a piece of metadata that is used by every object.

→ The storage class set for an object affects the object availability and pricing model.

→ You can change the storage ~~class~~ class of an existing object either by rewriting the object or by using object lifecycle management.

→ When you create a Bucket, you can specify a default storage class for the Bucket. If you don't specify, then default storage is standard.

All storage classes provides the following:

- Unlimited storage with no minimum object size ✓
- Worldwide accessibility and worldwide storage locations ✓
- Low latency ✓
- High Durability (99.999999999)
- Zero redundancy, if the data is stored in a multi-region or dual-region ✓
- A uniform experience with cloud storage features, security, tools and APIs ✓
- Object written to cloud storage must be redundantly stored in at least two different availability zones before the write is acknowledged as successful. ✓
- Max size of an object stored in cloud bucket is 5 TiB. ✓
- There is a 100 ACL limit per object.
- A multiple upload can have upto 10,000 parts. An individual part has a max. size limit of 5 GiB.

Object assembled from a multipart upload are subject to the overall 5 TiB size limit.

① Standard Storage

Standard storage is best for data that is frequently accessed (Hot Data) or stored for only brief periods of time.

→ When used for a region, standard storage is appropriate for storing data in the same location as GKE Clusters or Compute Engine instances that use the data.

→ Regional storage is redundant within a Region.

→ When used in dual-region, improved availability

→ When used in multi-region, ideal for storing data that is accessed around the world, such as serving website content, streaming video, executing interactive workloads or serving gaming data or apps.

→ Multi-regional is Geo-redundant stored across 2 or more regions which are 500 miles apart.

★ Availability of Standard storage data is:

Location type	Availability SLA → (Service level agreement)
Multi-region	99.95 % ✓
Dual-region	99.95 % ✓
Regional	99.90 % ✓

(2) Nearline Storage

Nearline storage is a low-cost, highly durable storage service for storing infrequently accessed data.

-> Nearline storage is better choice than standard storage in scenarios where slightly lower availability, a 30-day minimum storage duration and cost per data access are acceptable trade-offs for lowered at-rest storage costs.

-> Nearline storage is ideal for data you plan to read or modify on average once per month.

* Availability of nearline storage data is

Location-type	Availability SLA
Multi-regions	99.9 % ✓
Dual-regions	99.9 % ✓
Regions	99.0 % ✓

③ Coldline Storage

Coldline storage provides very low-cost highly durable storage service for storing infrequently accessed data (cold data)

→ Coldline storage is ideal for data you plan to access or modify at most once a quarter

☆ Availability is same as 'nearline'

④ Archive Storage

Archive storage is the cheapest, highly durable storage service for data archiving, Online Backup and disaster recovery.

→ Unlike the 'coldest' storage service offered by other cloud providers, your data is available within milliseconds, not hours or days.

→ Archive storage has higher cost per data access, as well as 365 days min. storage durations.

→ Archive storage is the Best choice for data that you plan to access less than once a year

☆ Availability is same as "nearline or coldline"

To maximize the availability of your data

-> Consider storing your data in a multi-region or dual-region bucket locations if high availability is a top requirement.

-> All data is stored geo-redundantly in these locations, which means your data is stored in at least two geographically separated regions.
In the unlikely event like regions wide outage, buckets in geo-redundant locations remains available, with no need to change storage path.

*-> You cannot change Bucket location or project after creation.

Lec-20 Cloud storage pricing, Concept of Bucket and object

Cloud storage pricing is based on the following components :-

1. Data storage
2. Data Processing
3. Network usage

1. Data storage

The amount of data stored in your Buckets storage rates vary depending on the storage class of your data and locations of your buckets.

Data Processing

The processing done by cloud storage, which includes operations charges any applicable retrieval fees and inter-region replication.

Network usage

The amount of data read from or moved between your buckets.

for eg- Data storage cost in Iowa

Location ↓ / Types →	Standard	Nearline	Coldline	Archive
Regional	$ 0.020	$ 0.010	$ 0.004	$ 0.0012
Dual-Region	$0.022	$ 0.011	$ 0.0044	$ 0.0014
multi-regions	$ 0.026	$ 0.010	$ 0.007	$ 0.004

Note: Dual regions are billed to both underlying regions at the above prices.

for eg:- Standard storage in a dual-region comprised of Iowa & Oregon will be billed at $0.22 per GB per month for IOWA and $0.22 per GB per month for Oregon.

Data processing fees :-

Data processing costs consists of operations charges, which apply to all request made to cloud storage, retrieval fees, which apply to reading data stored in certain storage classes and Inter-regions replication charges, which apply to data written to dual regions and multi-regions.

Price is $0.05/10,000 operations in standard storage & $0.10/10,000 operations in nearline and coldline.

3. Retrieval/Network usage fees -

Applies when you read, copy, move or rewrite object data or metadata that is stored using nearline storage, coldline storage or archive storage.

Standard	Nearline	Coldline	Archive
$0/GB	$0.01/GB	$0.02/GB	$0.05/GB

-> Default inter-region replication is free. Turbo replication is $0.04/GB chargeable.

-> Data move within the same location is free.

-> Data moves different locations on the same continent is chargeable.

-> Data moves between different continents is chargeable.

Cheap storage = High Retrieval cost

Bucket

-> Bucket is like a folder for your objects.
-> Using buckets - you can organise and maintain access permissions.
-> you cannot create nested Buckets.
-> Bucket has name, default storage class, Geographic location.
-> Name and location can only be changed by deleting and Recreating.
-> There is only one Google Cloud Storage namespace, which means every bucket must have a unique name across the entire google Cloud Storage Namespace.
-> Object names must be unique only within a given bucket.
-> Size is unlimited

Object

-> object is file that you want to store.
-> Can be upto 5 TiB.
-> Object has data & Metadata
-> No limit on number of objects in Buckets.
-> Objects are immutable and you can overwrite objects.

Turbo Replication

Turbo replication is a cloud storage premium feature designed to asynchronously replicate newly written cloud storage objects associated with any insert, rewrite, copy or compose operation - regardless of object size to a separate region within a target of 15 minutes.

W-21 Complete Demo on Cloud Storage using GUI & Cloudshell

-> Login to GCP console -> go to Cloud storage
-> Bucket -> Create Bucket

① Name your bucket (unique)
Labels (Optional) 3-63 characteristics

② Choose where to store your data
- Multi-region <= US, Europe, Asia
- Dual-region
- Region

③ Choose a default storage class for your data
Standard, Nearline, Coldline, Archive

④ Choose how to control access to object
- uniform
- fine-grained

⑤ Choose how to protect object data protection tools
- ☐ None
- ☐ object versioning
- ☐ Retention Policy

Data Encryption

- Google Managed encryption key.
- Customer Managed encryption key (CMEK)

-> [create]

[Cloudstorage demo using Cloudshell]

-> open cloudshell -> select your project

-> ls -a $Home

-> sudo rm -rf $Home -> Restart cloudshell

① Create a bucket

-> ~~gcloud~~ gsutil mb -b on -l us-east1 gs://tgbucket454/
(make bucket) (location)

② upload an object into your bucket

cloudshell -> : -> upload -> ⊙ file -> upload file

-> ☆ gsutil cp Sarfaraz.JPG gs://tgbucket454
(file) (destination)

③ Download the object from your bucket

-> gsutil cp gs://tgbucket454/Sarfaraz.JPG /home/awscsmay/download.JPG
source destination

④ Copy the object to a folder in the bucket

-> gsutil cp gs://tgbucket454/Sarfaraz.JPG gs://tgbucket454/myfolder/new.JPG
new name

(5) List contents of a bucket or folder

-> gsutil ls gs://tgbucket454

(6) List details per an object

-> gsutil ls -l gs://tgbucket454/sarlanz.JPG

(7) make the object publicly accessible

-> gsutil iam ch allUsers:objectViewer gs://tgbucket454
(small o)

(8) Delete an object or bucket

-> gsutil rm gs://tgbucket454/sarlanz.JPG

(9) Delete entire bucket & its objects

-> gsutil rm -r gs://tgbucket454

Lec-22

Demo - object versioning and Lifecycle rules per Bucket

To support the Retrieval of objects that are deleted or Replaced, Cloud storage offers the object versioning feature.

One enabled object versioning:

-> Cloud storage retains a non-current object version. Each time you replace or delete a Live object version, as long as you do not specify the generation number of the live version.

-> Non-current versions retain the name of the object, but are uniquely identified by their generation number.

-> Non-current versions only appear in request that explicitly call for them to be included.

If you disable object versioning:

-> The bucket no longer accumulates new non-current versions of object.

-> Object versions that already exist in the bucket are unaffected.

To set on/off of versioning:

-> gsutil versioning set on gs://sarlaing123

-> gsutil versioning set off gs://sarlaing123

To check status

-> gsutil versioning get gs://sarlaing123

To check versions available

-> gsutil ls -a gs://sarlaing123

To make particular version as live object

-> ~~gsutil cp gs://sarlaing123~~

-> gsutil cp gs://<source Id> gs://<dest Id>:txt>

To delete version

-> gsutil rm gs://<Id>

Object Lifecycle management:

To support common use cases like setting a TTL (time to live) for objects, retaining non-current versions of objects, or downgrading storage classes of objects to help manage costs, cloud storage offers the object Lifecycle Management features.

In order to use object Lifecycle Management, you define a lifecycle configuration, which must be set on a bucket. The configuration contains a set of rules which apply to current & future objects in the bucket. When an object meets the criteria of one of the rules, cloud storage automatically performs a specified action on the object.

Here are some of the use cases:-

-> Downgrade the storage class of objects older than 365 days to coldline storage.

-> Delete objects created before 1 Jan, 2022

-> Keep only the 3 most recent versions of each object in a bucket with versioning enabled.

Object Lifecycle configuration:

Each Lifecycle management configuration contains a set of rules. Each rule contains one action and one or more conditions.

-> An object has to match all of the conditions specified in a rule for the action in the rule

to be taken.

-> If you specify multiple rules that contain the same action, the action is taken on an object when that object matches the conditions in any of the rules.

-> If multiple rules have their conditions simultaneously for a single object, Cloud storage performs the actions associated with only one of the rules, based on the following considerations,

-> The delete action takes precedence over any SetStorageClass action.

-> The SetStorageClass action that switches the object to the storage class with the lowest at-rest storage pricing takes precedence.

Standard ↓
Nearline ↓
Coldline ↓
Archive

* Use documentation of Object Lifecycle Management for CLI and configuration examples for object lifecycle management.

Lec-23 Demo - How to create Signed URL

A signed URL is a URL that provides limited permission & time to make a request.

-> Signed URLs contain authentication information in their query string, allowing users without credentials to perform specific actions on a resource.

Key parameters in Sign URL

① X-Goog-Algorithm -> used to sign the URL

② X-Goog-Credentials -> Information about the credentials used to create the signed URL.

③ X-Goog-Date -> The date & time the signed URL became usable, YYYYMMDD 'T' HHMMSS 'Z'

④ X-Goog-Expires -> The length of time the signed URL remained valid, measured in sec.

In our lab, we will take 60 sec.

Max duration is 604800 sec i.e 7 days

⑤ X-Goog-SignedHeaders -> Header that had to be included as part of any request that used the signed URL.

⑥ X-Goog-Signature -> The authentication string that allowed request using this signed URL to access resources.

Create Bucket
↓
upload file
↓
APIs & Services
↓
credentials
(service acc. & acct)
↓
Key
↓
Add key
↓
Create new key
○ JSON (Recommended)
○ P12 (for backward compatibility with code using the P12 format)
↓
Download key
↓
gsutil signurl -d 60ps sec key.json gs://(bucketname)/objectname.png

---

CloudShell
↓
sudo rm -rf $Home
↓
upload key file
↓
sudo su
↓
pip install pyopenssl

Lec-24 Demo - Static Website Hosting using Cloud Storage

-> Login to GCP console -> create a Bucket
(Bucket Name: bucket5543)

-> go to Cloudshell

-> wget "paste link" (html5webtemplates.co.uk)

-> ls

O/P sample style_banner.zip

Now create one directory & do all things in it.

-> mkdir website

-> mv sample style_banner.zip website/

-> cd website/

-> unzip sample style_banner.zip

-> gsutil -m cp -r sample style=banner gs://bucket5543
(source) (destination)

Go to console -> bucket -> folder -> select all files
or object -> Permission -> + Add

Principal -> All users

Select a Role -> Cloud storage -> viewer

Now, test your static website

Copy URL of index.html file & check in Browser

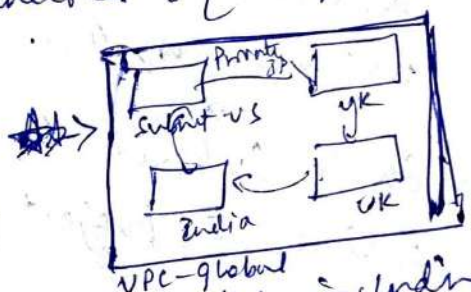
LU-25 Concept of VPC Networks

[VPC -> Global, Subnet -> Regional]

LAN -> Private IP
WAN -> Public IP

-> VPC Network is just like your physical network except that it is virtualized within google cloud.

-> A VPC Network is a global resource that consist of a list of regional virtual subnetworks in data centers, all connected by a google wide area network.



A VPC Network provides:

1. Provides connectivity for your VMs, including GKE & app engine instances.
2. Offers native Internal TCP/UDP Load Balancing.
3. Connects to on-premises network using cloud VPN Tunnels & cloud Interconnect attachments.
4. Distributes traffic from google cloud external Load balancers to Backends.
5. VPC always lies inside a project. One project can contain multiple VPC n/w.
6. New project start with a default VPC that has one subnet in each region.

VPC Networks have following properties:

-> VPC networks, including their associated routes & firewall rules are global resources. They are not associated with any particular region or zone.

-> Subnets are regional resources. Each subnet is associated with a region. You can create more than one subnet per region.

-> Each subnet is having unique network range inside a project.

-> Inside and outside traffic can be controlled by firewall rules.

-> Resources within a VPC, can communicate with one another by using IPv4 and IPv6 addresses.

-> An organization can use shared VPC to keep a VPC network in a common host project.

-> VPC networks can be connected to other VPC networks in different projects or organizations by using VPC Network Peering.

-> VPC Network do not support broadcast or Multicast addresses within the network.

-> Auto mode VPC networks create subnets in each region automatically.

-> You cannot connect directly Auto mode VPC network to one another because they uses same ~~segments~~ Ranges 10.128.0.0/9

W2.6 Creating Automatic Mode VPC using Console & Cloudshell

(1) Login to GCP console -> VPC Network ->
Create VPC Network
Name - my-auto-VPC , VPC ULA IPv6 -> Disabled

Subnet -> ○ custom
◉ Automatic

IPv4 firewall Rules -> allow TCP:22

Dynamic Routing mode -> ◉ Regional
○ Global

MTU -> 1460 / 1500 / 8896
(max. Transmission unit) -> [Create]

(2) go to compute engine -> Create two VMs in different regions (for eg -> us-central1 & asia-southeast1)

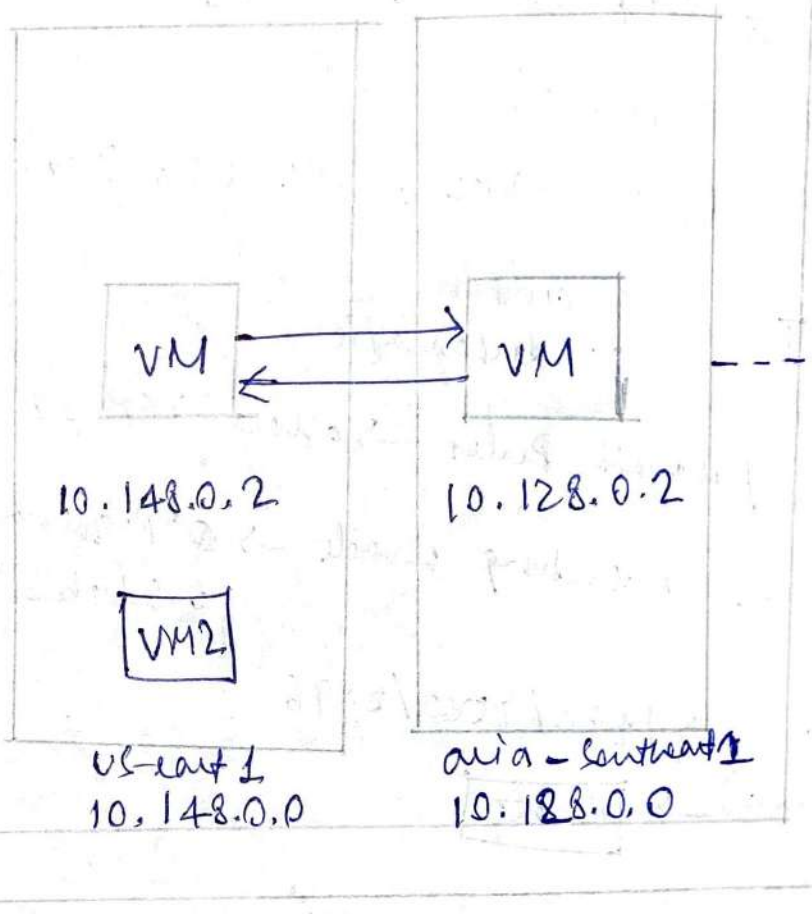
(3) Take SSH of Both VMs and Ping each other with their Private IPs.

MTU is the size in Bytes, of the largest packet supported by a network layer protocol, including both headers & Data.
Use Higher MTU (>1600) can be used only if the source & Dest. interfaces are in the same subnet and use private IPs.

GCP account

Project - Sunlaroz

VPC - Network



Firewall - ICMP - allow for testing Ping

Lu 27

Creating Custom VPC using GUI & Cloudshell

① Login to GCP console -> Select your project

② VPC Networks -> Create VPC Network.

Name - myvpcnetwork-VPC

Subnet creation Mode - Custom

New subnet - Subnet-asia-east1

IPv4 Range - 10.0.1.0/24

Add subnet

Name - subnet-us-east1

IPv4 Range - 10.0.2.0/24

-> Done

③ IPv4 Firewall Rules

☑ allow-icmp

☑ allow-~~tcp~~ ssh

④ Dynamic Routing Mode -> Regional

-> CREATE

Cloudshell commands :

-> gcloud compute networks create VPC-name
-- project = project-id -- Subnet-mode = custom

To create a Subnet

-> gcloud compute networks subnets create
Subnet-name --network = VPC-name --region = region
-- range = 10.0.3.0/24

To check available networks

-> gcloud compute network list

To create firewall Rule

-> gcloud compute firewall-rules create
fwrule-allow-icmp-ssh-rdp -- direction = INGRESS
-- priority = 1000 -- network = VPC-name -- action = ALLOW
-- rules = icmp, tcp:22, tcp:3389 -- source-ranges =
0.0.0.0/0

To get VPC details:
-> gcloud compute networks describe <mycli-VPC>

To change routing mode:
-> gcloud compute networks update <mycli-VPC> --bgp-
routing-mode = REGIONAL / GLOBAL

To create instance:
-> gcloud compute instances create myvm -- zone = <avia-east1>
-- machine-type = n1-std-1 -- Subnet = <Subnet-avia-east-1>

Lec-28

VPC Network Peering

-> Google Cloud Network peering allows Internal IP address connectivity across two VPC network regardless of whether they belong to the same project / organization or not.

-> In VPC Network Peering, traffic stays within Google Network & doesn't traverse the public ~~internet~~.

KEY FEATURES

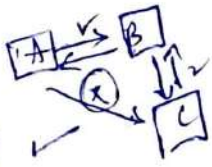
-> VPC network peering works with compute engine, GKE and app Engine flexible environment.

-> Each side of a peering association is set up independently. Peering will be active only when the configuration from both sides matches.

Either side can choose to delete peering association at any time.

-> VPC Network peering supports IPv4 connectivity only.

-> CIDR Range cannot overlap in two peered Networks.



-> Transitive peering is not supported.

-> One VPC can directly peered with max 25 VPC. Peering is allowed with shared VPC.

VPC 1

↓

Subnet: 10.0.0.0/24
us east 1

↓

All icmp
& ssh

↓

Create instance

↓

VPC network peering
create connections

VPC 2

Subnet : 172.16.0.0/16
asia east 1

↓

All icmp
ssh

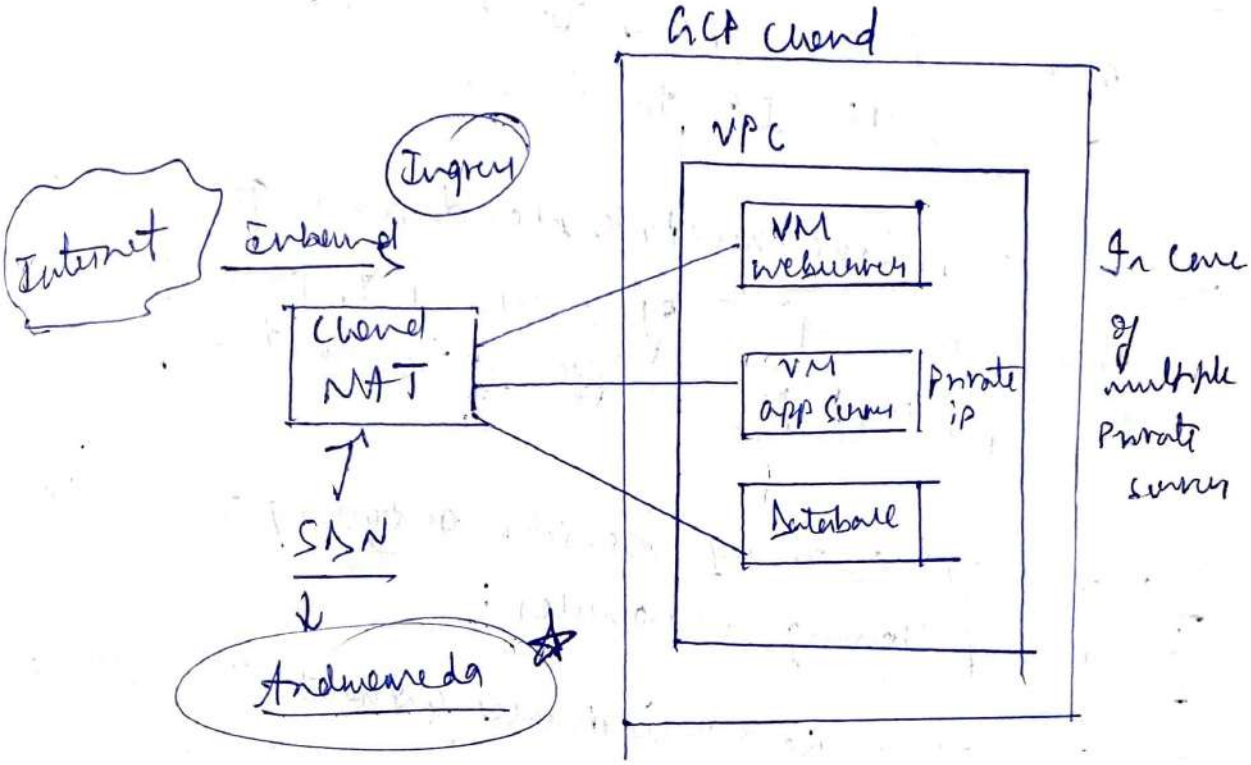
↓

create instance

[☆ Peering must be from B.S]

LL-29

Cloud NAT Gateway Demo



Network Address Translation (NAT) is a method of remapping one IP address space into another by modifying network address information in the IP Header of Packet. Usually this is done in a NAT Device.

IP Masquerading or NAT is a technique that hides an entire IP address. Usually the Private IP address behind a single public IP. The hidden addresses are changed into a single public IP as the source of the outgoing IP packet so they appear as originating from the routing device & not from the hidden host.

→ Cloud NAT allows outbound traffic but ☆ Restrict inbound traffic from external resources.

→ Cloud NAT allows your resources to create outbound connections to the internet without giving public IP to them.

→ You can have multiple cloud NAT gateways in the same region of a VPC but, Each gateway is configured for a different subnet. ☆

Cloud NAT Gateway provides outgoing connectivity for the following resources:

1. Compute engine VM instances without IP addresses.
2. Private GKE Clusters
3. Cloud Run instances through serverless VPC access.
4. Cloud functions instances through Serverless VPC access.
5. App Engine standard environment instances through serverless VPC access.

→ Cloud NAT cannot apply to any custom route where next hop (nextHop) is not the default internet Gateway.

→ Google cloud NAT is based on Andromeda ☆ i.e Google Network Virtualization stack.

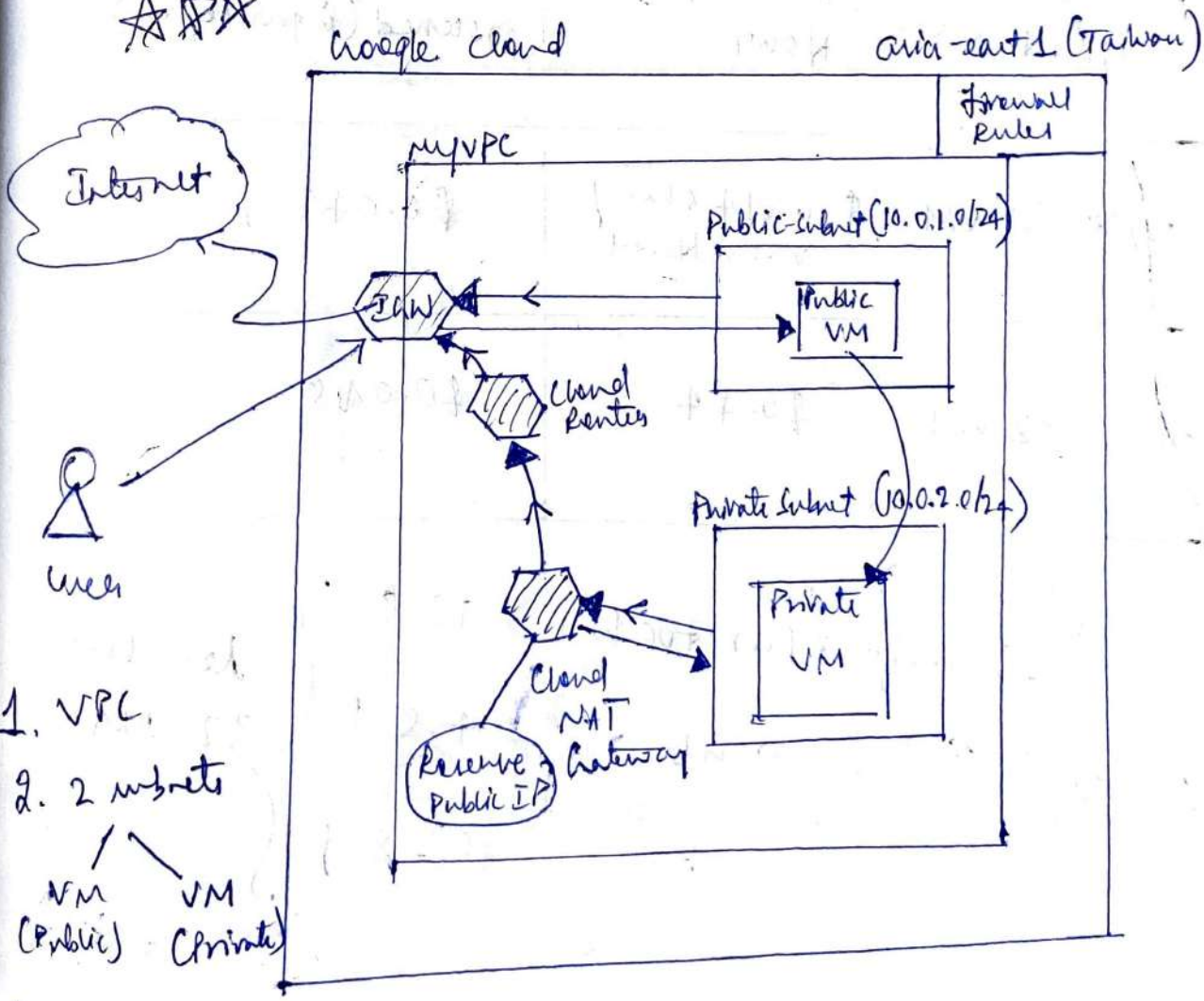
→ There is no intermediate NAT proxies in the data path. Each VM is programmed by GCP to NAT using assign ports. ✓

→ Each cloud NAT gateway is associated with a single VPC network, region and cloud Router.

☆ Cannot use same cloud NAT in two or more regions.

→ You cannot use cloud NAT for communication between VM's in a VPC network. It is designed to provide connectivity to the internet.

☆ ☆ ☆



1. VPC.
2. 2 subnets
   - VM (Public)
   - VM (Private)
3. Public reserve
4. Cloud router
5. NAT (Connect with router & allocate Public IP)
6.

☆ Cloud NAT Pricing

Cloud NAT Pricing is based on following mage.

① A hourly price of NAT Gateway that is based on the number of VM instances that are using the gateway.

② Per-GB cost for ingress (inbound) & egress (outbound) data that is processed by the gateway.

No. of assigned VM	Price per Hour	Price per GB processed (ingress/egress)
upto 32 VM	$ 0.014 x no. of VM x Hour	$0.045
above 32 VM	$0.44	$0.045

$ 0.014 x 10 x 700 hrs = 98 $

0.045 x $\frac{GB}{100}$ = $ 4.5 = 100 GB = 4.5 $

102.5 $

less than 32 VM

Networking

(1) VPC

↓

Subnet (custom) public

↓

10.0.1.0/24

aws-east 1

↓

allow-custom

icmp

rdp

Subnet (custom) Private

↓

10.0.2.0/24

aws-east 1

(2) VM instances

Public VM

↓ ~~Subnet~~ public

~~Windows~~ (desktop)

External IP/Address → Ephemeral

Private VM

↓ ~~Subnet~~ private

windows (desk)

External IP → none

↓

(3) R→P

Show external

→ private IP

→ awscli

→ p/w

$ip config

$ ping 8.8.8.8

no reply

networking

(4) VPC network
↓
IP addresses
↓
External IP
(Reserve)
↓
Specify same region

(5) Hybrid connectivity
↓
Cloud Router
↓
Specify same region

(6) Network Services
↓
Cloud NAT
↓
Select cloud router
↓
VPC - Sregion
↓
Cloud NAT mapping

Cloud NAT IP → manual → Select created one

☆☆
Delete

1. NAT
2. Router
3. IP address (Release)
4. VM
5. VPC

Lec-30 Theory on Load Balancers - Part - 1

→ A Load Balancer distributes user traffic across multiple instances of your application. By spreading the load, Load Balancing reduces the risk that your applications experience performance issues.

→ Cloud Load Balancer is built on the same front-end-serving infrastructure that powers Google. It supports 1 million+ queries per second with consistent high performance & low latency. Traffic enters cloud load Balancing through 80+ distinct Global load Balancing locations, maximizing the distance traveled on Google's fast private new backbone.

Google Cloud offers the following Load Balancing features:

1. Single anycast IP address.
2. Software-defined Load Balancing
3. Seamless autoscaling. No prewarming
4. Layer-4 & Layer-7 (HTTP/HTTPS) LB (TCP/UDP/ICMP)
5. External & Internal LB
6. Global (External) & Regional LB
7. Enable Cloud CDN for HTTP(S) Load Balancing to optimize application delivery for users near with a single checkbox

⑧ Manage SSL certification & Decryption.

★ Load Balancer can be opted on the Basis of your Requirement. In Google, you have various aspects:

① External versus Internal LB

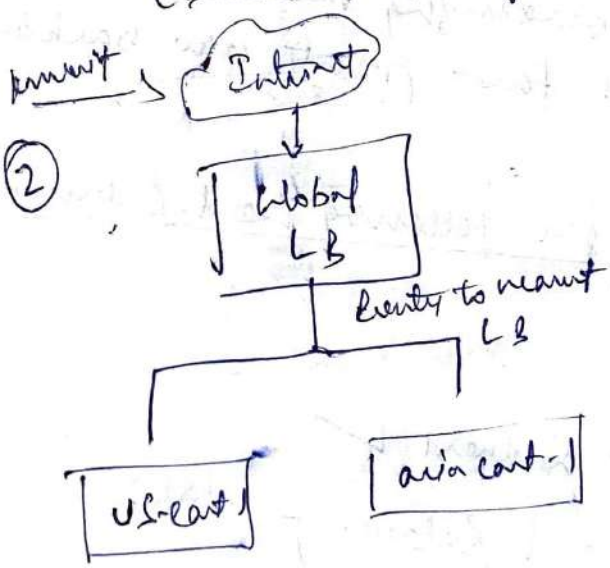
② Global versus Regional

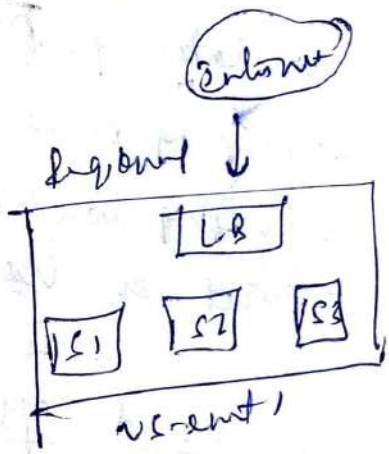
★ ③ Premium (Cloud Interconnect) v/s Standard Tier (By default → Premium)

④ Proxy (with SSL offloading) versus Pass-through

⑤ Traffic - Type (TCP/UDP/~~HTTP~~/~~HTTPS~~)

⑥ DDoS protection (Distributed Denial of Service)



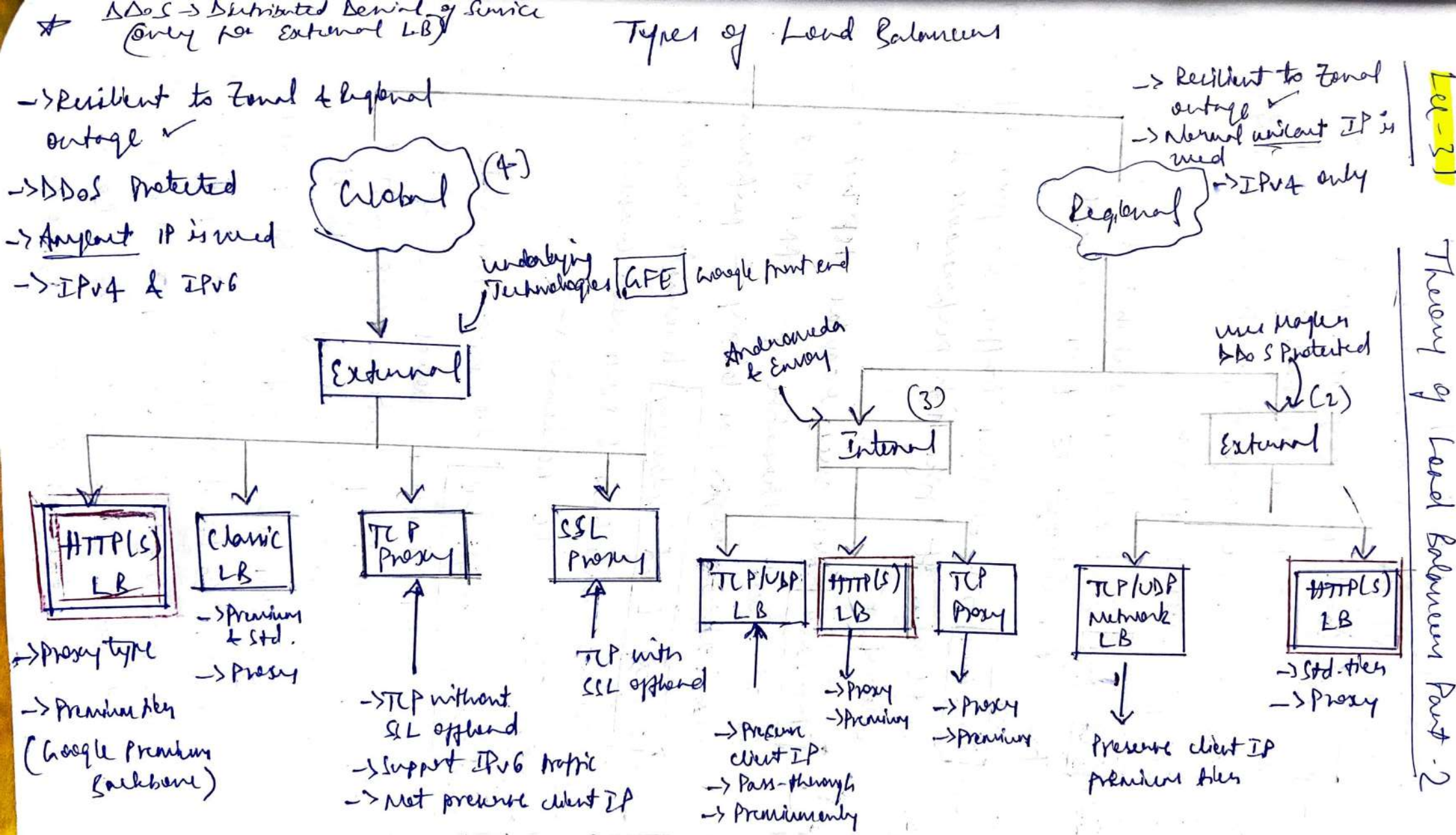


④ ★ Proxy → decrypted to encrypted from LB & send to servers

Pass-through → Server decrypts directly (direct connection) (or) return

Theory of Load Balancer Part-2

Types of Load Balancers



-> **Global** Load Balancing when your backends are distributed across multiple regions.

-> **Regional** Load Balancing when your backends are in one region or when you have jurisdictional compliance requirements for traffic to stay in a particular region.

-> **External** Load Balancers distribute traffic coming from the internet to your Google Cloud VPC.

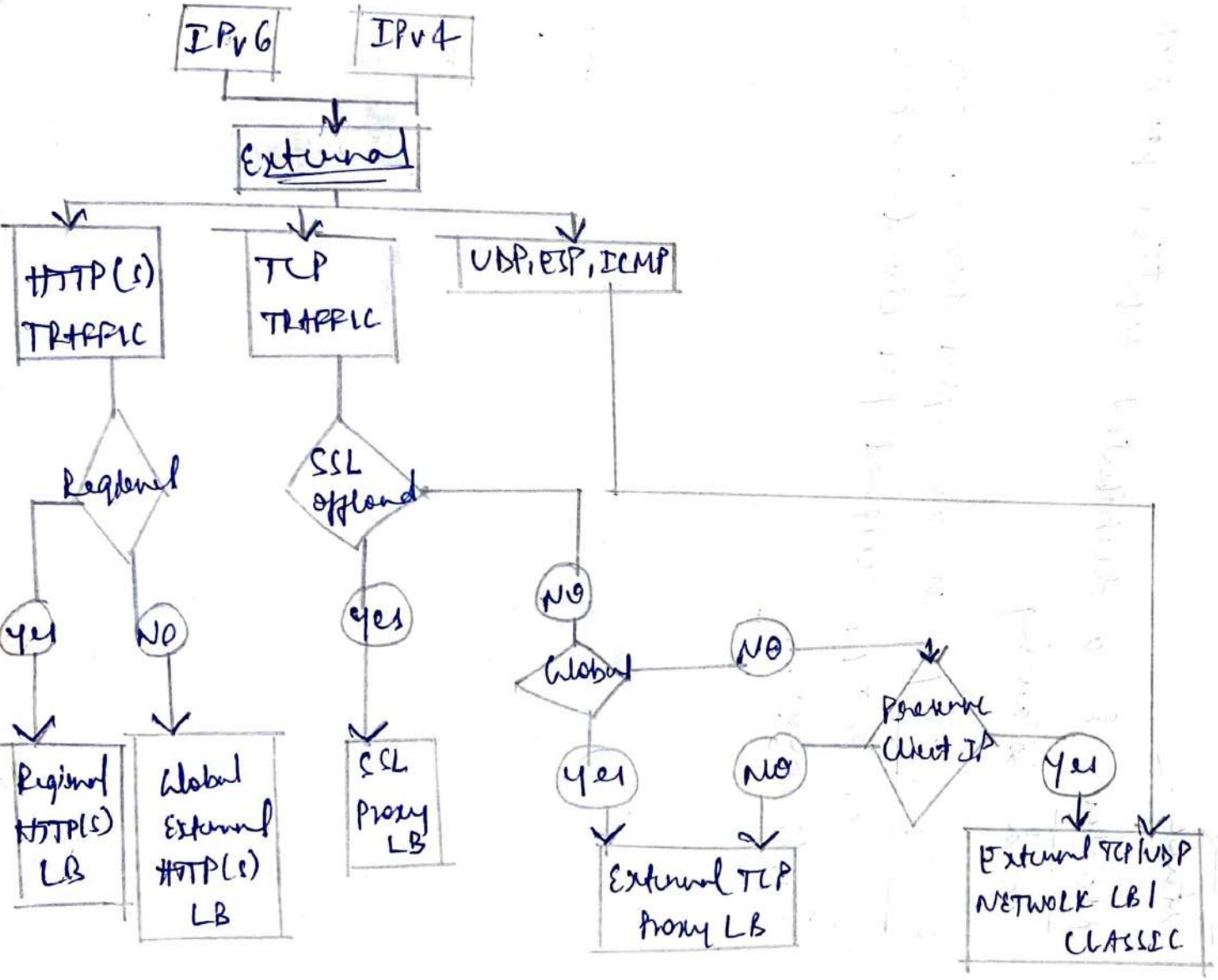
-> **Premium** tier delivers traffic on google's premium Backbone. It gives high performance & low latency.

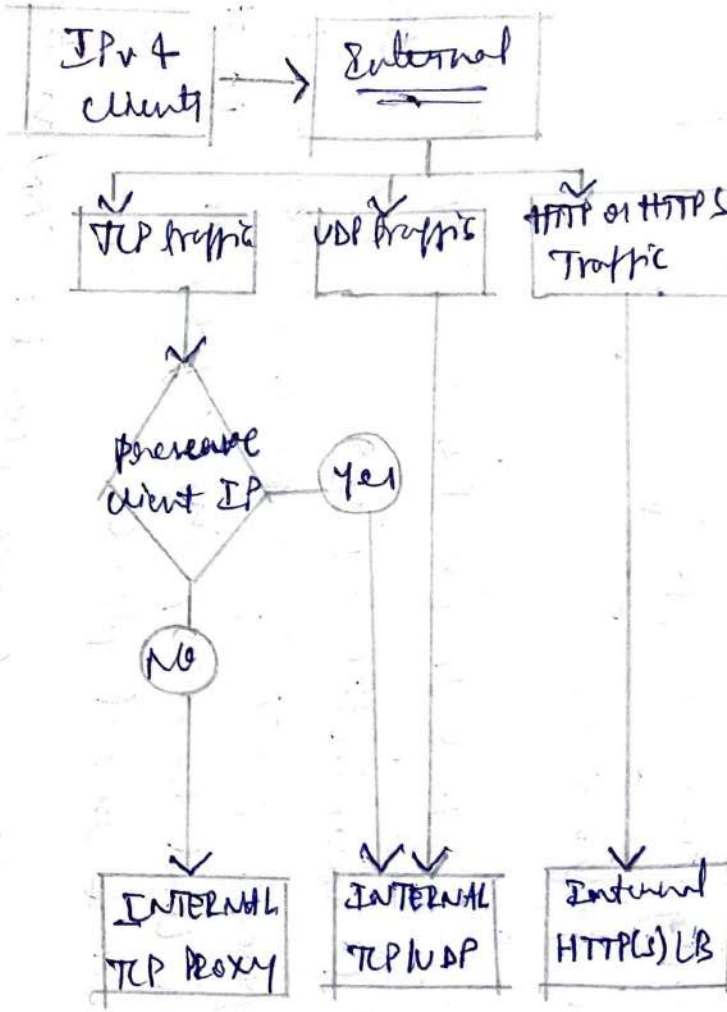
-> **Standard** tier uses regular ISP network. It is low-cost alternative for apps that don't have strict req. for latency & performance.

-> **Proxy Load Balancers** terminate incoming client connections & open new connections from the LB to the backends.

-> **Pass-through LB** do not terminate client connections, instead, load Balancers packets are received by backend VMs with the packet source, destination, & if applicable, port information unchanged connections are terminated by the Backend VM's. Responses from the Backend VMs go directly to the clients, not back through the load Balancer. The term for this is direct server return.

Decision Tree for choosing a Load Balancer:





Underlying Technology of Google cloud Load Balancers.

① Google Front Ends (GFEs) are software defined, distributed systems that are located in google points of presence (PoPs) & perform Global Load Balancing in conjunction with other systems & control plane.

② Andromeda is Google cloud's SDN virtualization stack.

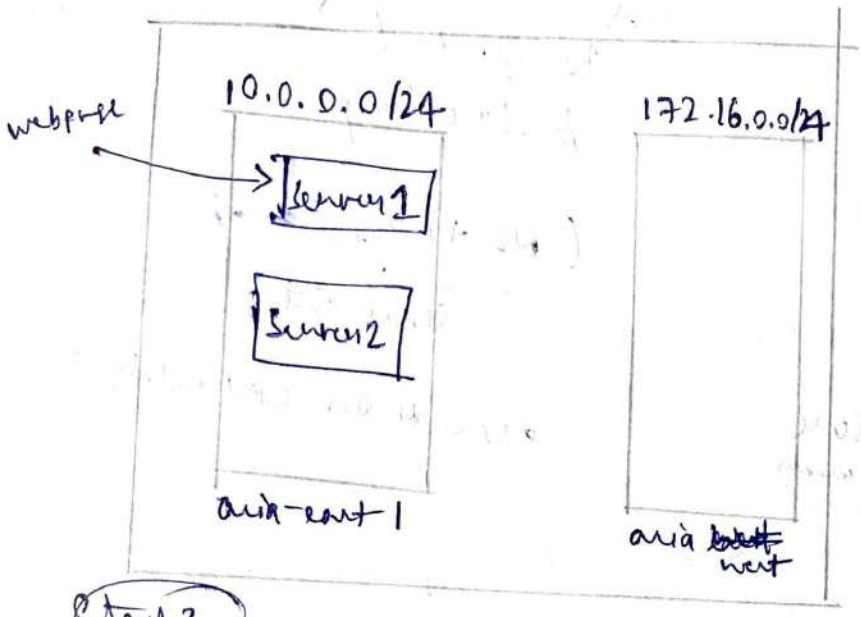
③ Maglev is a distributed system for Network Load Balancing

④ Envoy proxy is an open source edge & service proxy, designed for cloud-native applications.

Lec-32 Demo of HTTP(S) Load Balancer

→ GCP → Project

Step 1 MyVPC



Step 2

→ Instance Group

```
-> yum install httpd -y
-> service httpd start
cd /var/www/html
vi index.html
```

Step 3 Load Balancer

Activity:

~~① Instance group~~

① Network Services → Load Balancers
Backend
Frontend

② Compute Engine → Instance group

③ Health checks

④ VPC & Subnets

Lec-33 Autoscaling - Demo

Autoscaling

- Vertical Autoscaling (manual) (from N2 → E2)
- Horizontal Autoscaling (N2 + N2) → Scale out.

Scale up ↑ 2GB → 2GB ↓ Scale down

depends on CPU utilization

Demo

1. Instance Group
   a. Min Serving : 1
   b. Max Serving : 10
   c. CPU utilization : 60%.

Compute Engine
↓
Instance Group
↓
Create IG

→ Create new instance template
- → N1
- → Boot disk
  a. windows
  b. desktop exp.
- → Firewall
  HTTP ☑
- → VPC
- → Locations
  + Region (Zones)
  + Min Instance + Max Instance
  + CPU - 60%.
  + Create Health check

Window RDP
↓
- Text
- a.bat

Save as → All files

Deleting

1. Instance Groups

Lec-34

Google Cloud IAM Part-II

-> IAM lets you grant granular access to specific Google cloud resources & helps prevent access to other resources.

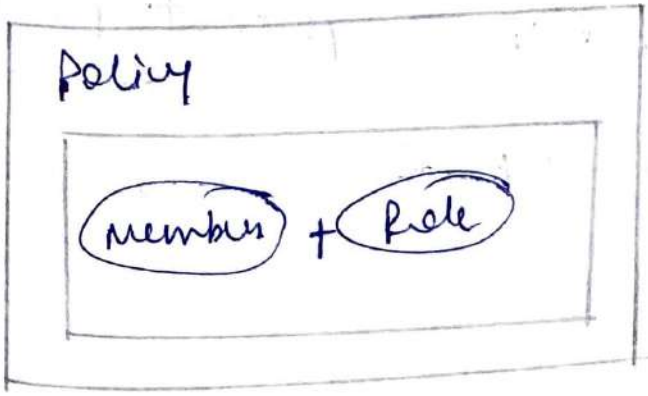
-> IAM lets you adopt the security principle of least Priviledge, which states that nobody should have more permissions than they actually need

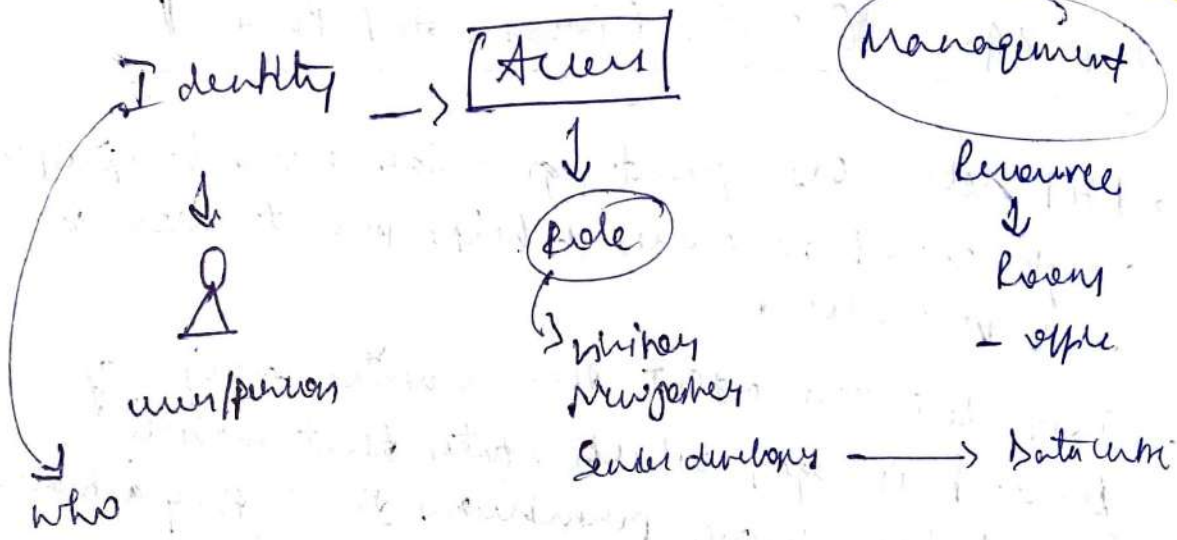
-> with IAM, you manage access control by defining 'who' (Identity) has what access (role) for which resources.

-> IAM, permissions to access a resource is not granted directly to the end user Instead, permissions are grouped into roles, and roles are granted to authenticated Principals (members)

-> Policy defines & enforces what roles are granted to which principals. Each allow policy is attached to a resource.

-> When an authenticated principal attempts to access a resource, IAM checks the resource's allow policy to determine whether the action is permitted.





Features of IAM

→ Lets you authorize who can take specific actions on resources to give you full control & visibility on your google cloud services centrally.

→ Permissions are represented in the form of "service.resource.verb".

→ Can map job functions into groups & roles.

→ with IAM, users only get access to what they need to get the job done.

→ Cloud IAM enables you to grant access to cloud resources at fine-grained levels, well beyond project level access.

Basically IAM has three important parts:

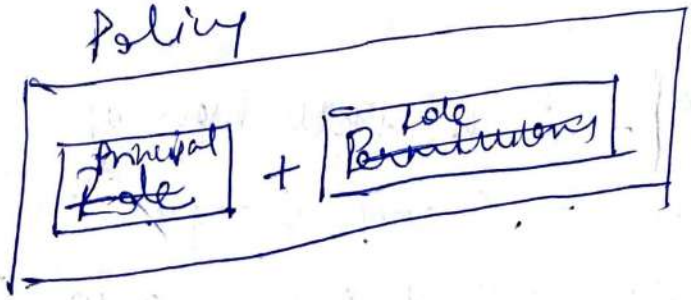
1. Principal/members.
2. Role
3. Policy

Principal / Members: A principal can be a

(a) Google account
(b) Service account
(c) Google Group
(d) Google workspace
(e) Cloud Identity domain
(f) Authenticated users
(g) All users.

Role A role is a collection of Permissions. Permissions determine what operations are allowed on a Resource. When you grant a role to a principal, you grant all the permissions that the role contains.

Policy policy is a collection of role bindings that bind one or more principals to individual roles. When you want to define who has what type of access on a resource, you create allow policy & attach to resource.



Lec-35 Google Cloud IAM Part-2

In IAM, you grant access to principals.

☆ **Principals** can be of the following types.

1. Google a/c
2. Service a/c
3. Google Group
4. Google workspace a/c
5. Cloud Identity domain
6. All authenticated users
7. All users.

Google account → per person – Any email address that is associated with a google account can be an identity, including gmail.com or other domains.

Service account → per service – A service account is account for an application or compute workload instead of an individual end user.

Google Group – is a collection of Google accounts & service accounts. Every Google Group has a unique email address that's associated with the group.

You can grant & change access controls for a whole group at once instead of granting or changing access controls one at a time for individual users or service accounts.

virtual group

Google workspace account - Google workspace accounts are associated with your organization's internet domain name, such as example.com. When you create a google account for a new user, such as username@example.com, that google account is added to the virtual group for your Google workspace account.

Cloud Identity Domain -> A cloud Identity domain is like a google workspace account, because it represents a virtual group of all Google accounts in an organization. However, cloud Identity domain users don't have access to Google workspace applications & features.

All authenticated users - The value all authenticated users is a special identifier that represents all service accounts and all users on the internet who have authenticated with a Google account.

This identifier includes accounts that are not connected to a google workspace account or cloud Identity domain, such as personal Gmail accounts. Users who are not authenticated, such as anonymous visitors, are not included.

All users

The value All users is a special identifier that represents anyone who is on the internet including authenticated & unauthorized users.

★ Roles

- → Basic ⇒ Project level
- → pre-defined ⇒ Service level
- → custom-role

→ A role contains a set of permissions that allows you to perform specific actions on Google cloud Resources.

→ You don't directly grant users permissions in IAM. Instead you grant them roles, which bundle one or more permissions.

→ To make permissions available to members, including users, groups & service accounts, you grant roles to the members.

There are three types of roles in Google cloud IAM:

① **Basic / Primitive roles:**

IAM Basic roles offer fixed, coarse-grained levels of access.

Owner	Editor	Viewer	Billing Admin	Browser (Same as viewer)
-> Supervises at project level ✓ -> Add/remove members ✓ -> Delete & create projects ✓ -> Setup Billing for project ✓	-> Modify code -> Deploy apps. -> Configure service -> Stop & start service	-> Read only access	-> Manage billings -> Add or remove administrators.	Browser ↓ - Same as viewer - doesnot have cloud storage access

② Pre-defined Roles

-> Provides granular access per a specific service & is managed & defined by Google cloud. ✓

-> Prevents unwanted access to other resources ✓

-> Google is responsible for updating & adding permissions as necessary. ✓

-> You can grant multiple roles to the same user. ✓

③ Custom Roles

-> Provide granular access according to a user-defined list of permissions. ✓

-> You can create a custom IAM role with one or more permissions & then grant that custom role to users or groups. ✓

-> Custom roles are not maintained by Google.

-> You can grant multiple roles to a user or

Policy → (Principal + Role) + Conditions

→ A policy is a collection of bindings and configurations & metadata.

→ A binding associates one or more members with a single role & any context-specific conditions that change how & when the role is granted.

→ Each binding includes the following fields:

→ A member, known as an identity or principal, can be a

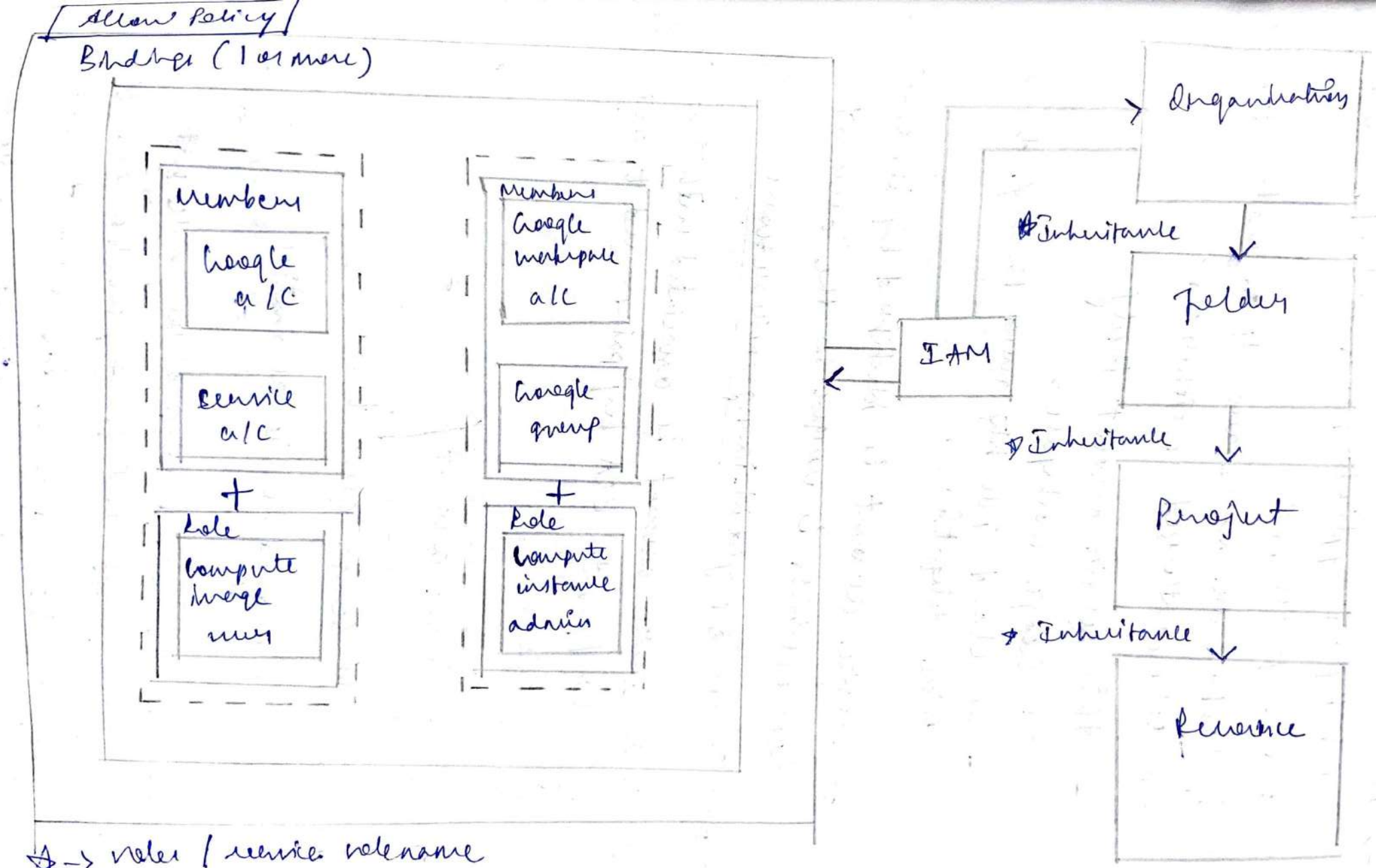
- Management
- Service account
- Google Group
- Domains

→ A role which is named collection of permissions that grant access to perform actions on Google Cloud Resources.

→ A conditions, which is a logical expression that further constraints the role binding based on attributes about the request, such as its origin, the target resource etc.

Definition

You can grant roles to users by creating an Allow policy, which is a collection of statements that defines who has what type of access.


Allow Policy
Binding (1 or more)
Members
Google a/c
Service a/c
+
Role
Compute image user
Members
Google workspace a/c
Google group
+
Role
Compute instance admin
IAM
Organization
Inheritance
Folder
Inheritance
Project
Inheritance
Resource
* -> roles / service role name

Service Account

→ A service account is a special kind of account used by an application or a virtual machine (VM) not a person.

→ Applications use service accounts to make authorized API calls, authorized as either:

- this service account itself
- as Google workspace
- as cloud Identity users through domain-wide delegation.

→ A service account is identified by its email address, which is unique to the account

→ Sname@project id.iam.gserviceac.com

→ Each service account is associated with two sets of public / private RSA key pairs used to authenticate to google.

Types of Service account :-

1. Default Service account
2. User-managed Service accounts

→ A service acc. is also a resource with IAM policies attached to it, which means you can define who can use, the account and who can perform specific actions on the service account.

Lec-36

Google Cloud IAM Demo Part-1

IAM -> [Grant access]
↓
Principal
(gmail.com) -> user
↓
Role -> Basic
Predefined

Basic - viewer
- Browser
- Editor
- owner

IAM & Admin
↓
IAM
↓
Grant Access (to create new user/principal)
↓
+ Create principal (email only)
+ Role -> owner
↓
[Save]

* Change Roles & check for practice

{ In viewer bucket is seen
Ex Browser bucket is not seen }

Lec-37 Google Cloud IAM Demo Part-2.

→ A Service account is a special kind of account used by an applications or compute workload, such as a compute engine VM instance, rather than a person.

for eg. → A service a/c can be attached to a compute engine VM, so that applications running on that VM can authenticate as the service account. In addition, the service account can be granted IAM roles that let it access resources.

Difference between Service account & User account :-

① Service a/c donot have passwords, and cannot log in via browsers or cookies.

② Service accounts are associated with public/private RSA key pairs that are used for authentication to Google and for signing data.

③ You can let other users or service a/c impersonate a service a/c.

④ Service a/c. do not belongs to your Google workspace domain, unlike user accounts.

☆ Types of Service account :

1. Default Service account ✓
2. User managed Service account ✓
3. Google managed Service account. ✓ (Backend)

Default Service account :- When you enable or use some google cloud services, they create default service accounts that enable the service to deploy jobs that access other Google cloud services. These accounts are known as Default service account. These two services create service account "App Engine", "Compute Engine" & any google cloud service that uses App Engine or Compute Engine.

User managed Service account :- You can create user-managed S.A in your project using IAM API, console & CLI. ✓

→ By default you can create 100 user managed S.A in a project. ✓

→ When you create a user managed S.A in your project you choose a name for the service account. which uses the following format :- ✓

S.A.name@project-id.iam.gserviceaccount.com

constant

Google Managed Service account

(Google may think in backend)

-> Some google cloud services needs access to your resources so that they can act on your behalf

for Eg. when you use cloud Run to Run a containers, the service needs access to any pub/sub topics that can trigger the container.

To meet this need google create Google managed S.A. 'Not listed in S.A page'.

Types of keys for service account

1. Google managed key pairs
2. User-managed key pairs

Google managed key pairs are used by the service account credentials API, and by Google cloud services such as App Engine & Compute Engine.

User-managed key pairs you can create user-managed key pairs for a S.A, then use the private key from each key pair to authenticate with Google API. This private key is known as a Service account key.

- Each service acc. can have upto 10 S.A keys.
- User is responsible for the security of private key.

IAM & Admin

↓

Service account

↓

① Create S.A

+ name: mysservice

+ Service account ID:

+ description: < >

↓

② Grant this service acc. access to project

+ Role: Compute admin

+ IAM Conditions: optional

↓

③ Grant users access to this S.A (optional)

+ user role

+ Service acc. admin role

Compute Engine

↓

Create instance

+ Region + Zone

+ N1

↓

☆ Identity & API access

Select created service account

↓

Firewall

+ HTTP

+ HTTPS

Lec-38 Google Cloud IAM Demo Part-3

Topic - Predefined & custom Role

Basic
↓
Admin ✓
viewer ✓
Browser ✓
Editor ✓

Predefined
↓
Compute admin ✓
Service account admin ✓
Cloud Storage admin ✓

Custom Role

★ Pre-defined Roles

① IAM & Admin
↓
IAM
↓
Grant access
↓
→ Add. Principals
+ sulmaz@gmail.com

→ Role
+ Compute instance Admin (v1)

Save

② Incognito
GCP
↓
Gmail
↓
Console
↓
Select project
↓
only instances can be accessed
↓
cannot access Identity & API access hence, VM can't be created.

③ Edit roles in IAM
+ Service account
→ Service account admin

Save

④ V.M can be created.
(★ wait for 5 mins)

Custom Roles

(1) IAM
↓ drop down
Roles
↓
+ Create Role
+ mynole (Title)
Description Class delete instance

+ ID : customrole123
↓
Role launch stage
+ Alpha
↓
Add Permissions
+ Compute instance delete

Add
↓
Re delete instances ; instances have to be seen
+ Edit → Add permissions → Compute instance list

(2) IAM
↓
users
↓
Edit
↓
~~del~~ keep only customrole (created)

Save

(3) Check per VM instances

(4) Delete users

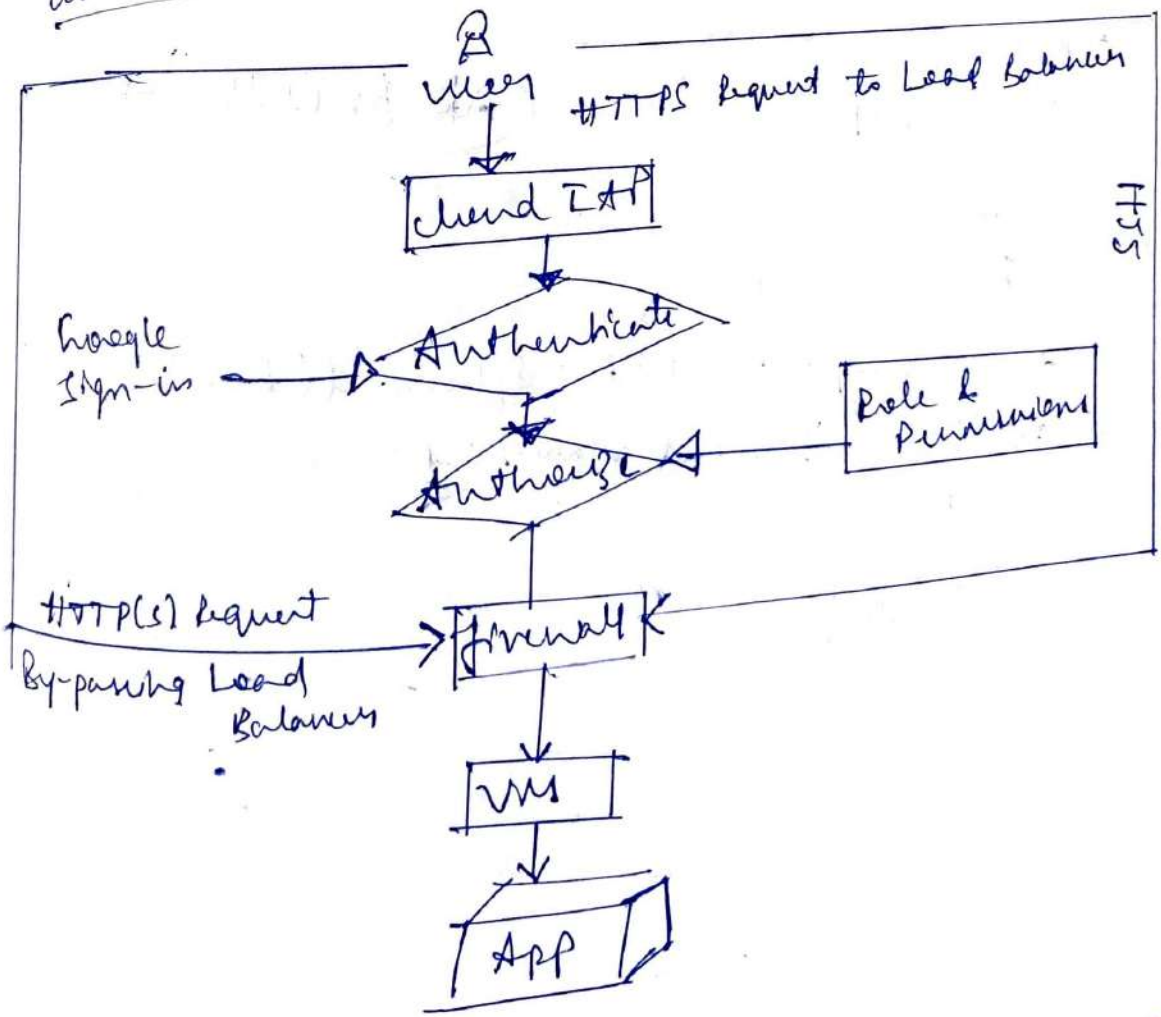
Lec-39

Identity Aware Proxy - Demo

IAP lets you establish a central authorization layer for applications accessed by HTTPS, so you can use application-level access control model instead of relying on network-level firewalls.

How IAP works:

* When an application or resource is protected by IAP, it can only be accessed through the proxy by principals also known as users, who have the correct identity & access management role.
* When a user tries to access an IAP-secured resource, IAP performs authentication & authorization checks.



→ IAP does not support (or) use VPN ✓

→ IAP allows access to webapps from anywhere

→ IAP follows the Zero-trust Security model ✓

→ IAP supports Google workspace account. ✓

* → IAP works for app engine, cloud run, compute engine, GKE & on-premises.

Demo

(1) Compute Engine

↓

VM instances

↓

Create instances

+ Name : Private VM

+ Region :

+ Series : N1

Allow firewall *

↓

Networking

+ Primary internal IP : Ephemeral (Auto)

* → + External IPv4 address : None

↓

[Create]

(2) VPC network

↓

firewall

↓

Create firewall Rule

+ Name : mypule

+ ↓

+ Target : All instances in the network :

~~+ All~~

+ Source IPv4 ranges : 0.0.0.0/0

+ Protocols & Ports : Specified protocols & ports

[✓] TCP

(Port : 22)

↓

[Create]

+ ~~Compute. instances. set metadata~~

~~+ service acc. user role~~

(3) Take SSH of created instance

~~Auto through web!~~

(4) Install GCP CLI

(5) IAM

↓

Create users

↓

Grant access

+ new principal : <email id>

+ role : Basic viewer

↓

[Create]

+Compute. instances. set metadata

+Service acc. user role

(6) IAM

↓

Identity aware Proxy

↓

SSH & TCP resources

↓

Copy Source IP (notepad)

↓

Edit firewall

↓

Create firewall rule

+ name : IAP rule

+ Targets : All instances

+ Source IP ranges : <IP>

+ Protocols & Ports

[✓] TCP → SSH 22 → [Create]

*

~~Lec 40~~ ~~App Engine - Standard & flexible~~

(7) IAM

↓

Identity Aware Proxy

↓

SSH & TCP resources

+ Select < Private-VM >

+ Add Principal,

→ New principal: Gmail ID

Role → Cloud IAP → IAP-Secured Tunnel User

↓

Save

(8) Compute Engine

↓

VM instance

↓

copy External IP

↓

Laptop → Google Cloud SDK Shell

↓

$ gcloud auth login

↓

Console → gcloud CLI → copy Code/Shell command & paste in CLI

$ sudo su

$ lsblk

$ ip addr

→ * Server can be accessed through Private IP

Lec 10

App Engine - Standard & flexible

-> The App Engine is Google's PAAS offering. It is a compute service that provides a managed platform for running applications.

-> As this is a managed service, your focus should be on the application only and google will manage the resources needed to run the application.

-> It was GCP earliest services, introduced in 2008.

-> Applications in app engine run across multiple servers.

-> No usage charges - Pay for resources provisioned

Google -> App Engine
AWS -> Elastic Beanstalk
Azure -> Azure app service

Features of App Engine :-

1 Project = 1 App Engine

-> App Engine makes deployment, maintenance, & scalability easy - so that the user may focus on innovation.

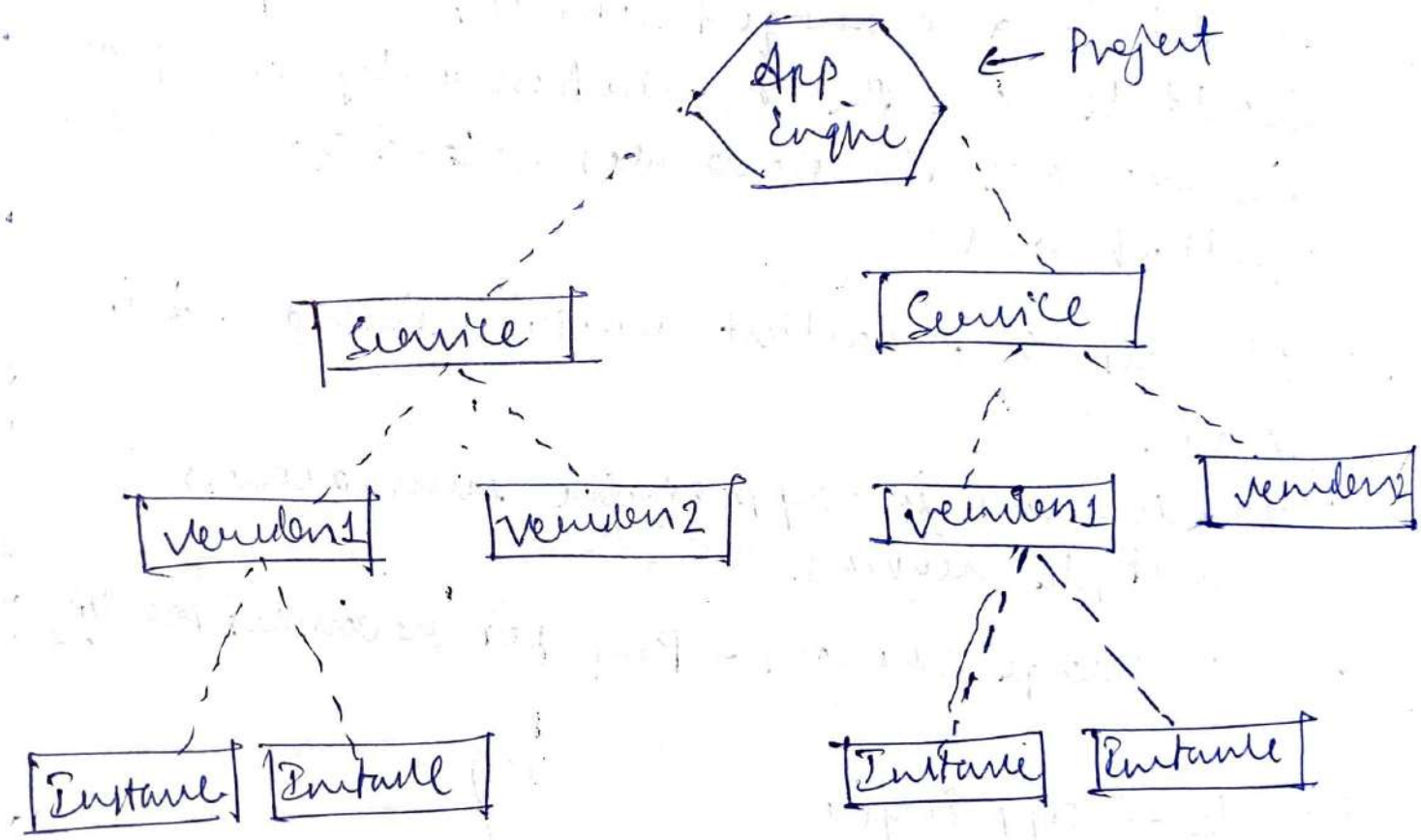
-> Automatic Load Balancing & Scaling

-> Applications Versioning & Traffic Splitting.

-> Supports several Programming Languages like Go, Python, Java, NodeJS, .Net & Ruby.

→ Suitable for web applications.

Services & versions



☆ https://[version-ID]-dot-[service-ID]-dot-[Project-ID].appspot.com

Features	Standard	Flexible
Instance startup	Seconds	minutes
SSH Access	NO	YES
write to local disk	NO (Some runtimes have read & write access to the /tmp/ directory)	Yes, Ephemeral (disk initialized on each VM startup), can allow app engine resources
Need of VPC	NO	Yes
Pricing model	After free daily use, pay per instance class, into automatic shutdown	Pay per Resource allocations per hours (VPC, storage etc)
Scaling to zero	Yes, high latency	min. 1 instance
Scaling	manual, Basic, Automatic (B) Backend Frontend	manual & automatic
	Sandbox Environment	App run in Docker containers, support more language

Difference b/w Compute Engine & App Engine:

	Compute Engine	App Engine
Service model	IaaS	PaaS
Type of Service	Unmanaged	managed by Google
Control over Resources	More control & flexibility over Resources	Less control over computing Resources
Cost	Less	High
	(Contradictory)	
Running Instance	Min 1	Scale to Zero
AutoScaling	Slower AutoScaling	Faster AutoScaling

Lec-41 App Engine Standard - Demo

-> Create new project
-> Open cloudshell
-> upload a folder in cloudshell
-> cd gae-file
-> ls
-> app.yaml package.json server.js
-> gcloud app create (if required)
-> gcloud app deploy

for Autoscaling

-> sudo apt install -y apache2-utils
-> ab -c 1000 -n 10000 "url of appengine"

To do versioning

do some changes in server.js.

-> gcloud app deploy -v version 2

To create a New Service

-> vi app.yaml

runtime : nodejs10
service : service2

-> gcloud app deploy -v service2 ver2

① Console

↓

New project

↓

< New project >

↓

[create]

② [Serverless]

↓

App Engine

↓

Dashboard

↓

Create application

↓

Select location

↓

Resources

Language : Python

Environment : Standard

Deploy with Google Cloud SDK

$ gcloud init

$ gcloud app deploy

with Cloud Shell

① Google → app engine GCP
↓
docs
↓
std
↓
node.js
↓
runtime

{ app.yaml
Package.jsons
Server.js }

② Cloud Shell
$ ls
$ sudo rm -rf $HOME (to clear everything in home)
↓
upload
↓
folder
↓
upload
↓
$ ls
Apps engine std files
$ cd Appsengine std files
$ node server.js

(Can preview using cloudshell)

* 2 Buckets are created by default by starting App engine

$ gcloud app describe (to check running apps)

$ gcloud app create (to initialize app)

* → $gcloud app deploy -v version1

(in another tab) $ gcloud app instances list

$ watch gcloud app instances list

In App Engine Dashboard

Services } are seen
Versions
Instance will be created once url is 'hit'

↓

~~$ ab~~ check url in cloudshell

↓ no runs more apps

$ sudo apt install -y apache2-utils

$ ab

★ → $ ab -c 1000 -n 10000 <url>

App Engine

↓

Settings

↓

Custom Domains

↓

Add A Custom Domain

To deploy in other versions

$ vi server.js

version 2 (in msg)

~~$ vi app.yaml~~

$ gcloud app deploy -v version2

Can split [traffic] in versions

Can migrate [traffic] to particular 'version'

To create fresh service

$ ls

$ vi server.js ✓

change msg

$ vi app.yaml ✓

add

service: newservice

$ gcloud app deploy -v service2-ver1

(Cloud Storage has all the configurations)

(Continue lab per next topic)

Lec-42 App Engine flexible Demo

The flexible environment provides more options & control to the developers who want to use App Engine, but without the language constraints of the standard environment. It uses Docker containers as the basic building blocks. These containers can be auto-scaled according to load.

Demo [VPC is mandatory for flexible EAP]*

↓

Dashboard

↓

cloudshell

↓

upload folder <GCP-42>

↓

$ ls

↓

$ cd GCP-42

$ ls

$ time gcloud app deploy -v flex1

(one instance is created without any hit) or without traffic

$ gcloud app browse

$ ipconfig

$ docker --version (app engine runs on containers)

$ cat /etc/os-release

$ docker exec -it <container id> sh
(go get into container)

ls

exit

Delete

1. migrate traffic to 1 version -> delete
   ~~deleted~~
2. Instances -> deleted (auto)

Lec-43 Cloud Run - Theory

Cloud Run is a managed (google) compute platform that lets you run containers directly on top of Google's scalable infrastructure.

→ Launched in 2019 ✓

→ AWS fargate & Azure Container instances are Equivalent services to Cloud Run.

→ You can deploy code written in any programming language. ✓

→ Containers can be called through web requests (or) Pub/Sub Events.

→ Cloud Run offers automatic scaling

→ Cloud Run provides per an open API & runtime environment. ✓

→ Cloud Run comes with a free-tier with additional Billing done on a pay per use basis.

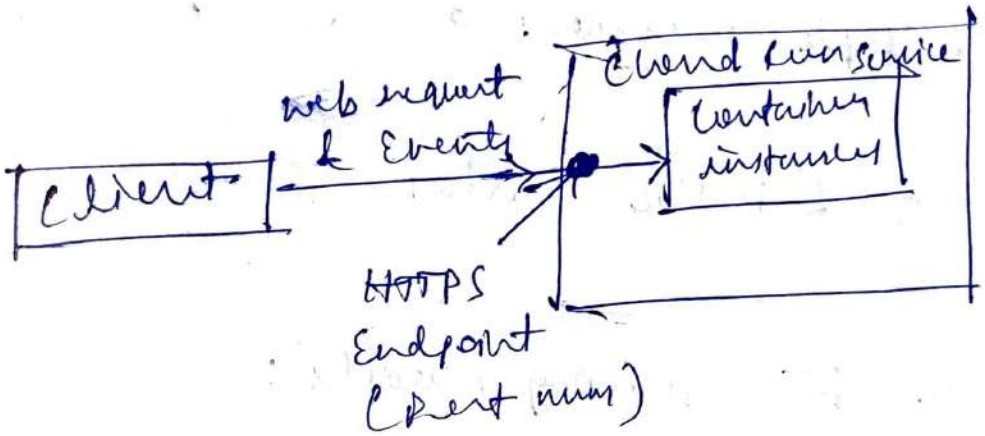
→ It is serverless & runs 'stateless' containers.

Two ways to run your code:

On cloud run, your code can either run continuously as a service or as a job

Cloud Run Services – used to run code that responds to web requests or events.

Cloud Run Jobs – used to run code that performs work (a job) & quits when the work is done.



A cloud Run service provides you with the infrastructure required to run a reliable HTTPS endpoint. Your responsibility is to make sure your code listens on a TCP port & handles HTTP Requests.

Features

① Unique HTTPS Endpoint per every Service

→ Every cloud Run Service is provided with an HTTPS Endpoint on a unique subdomain of the *.run.appdomain & you can configure custom domains as well.

② Fast Request-based autoscaling ← Horizontal

③ Built-in Traffic Management → Every deployment creates a new immutable revision. You can route incoming traffic to the latest version, roll back to previous versions, or split traffic to multiple revisions at the same time.

④ Private & Public Services

There are two pricing models:

1. Request Based – If a container instance is not processing requests, the CPU is not allocated & ~~you~~ you are not charged.

2. Instance-based – ~~you~~ are charged for the entire lifetime of a container instance & the CPU is always allocated. There is ~~not~~ no per-request fee.

Lec-44

Cloud Run Demo

(1) [Cloudshell]
↓
upload folder
Dockerfile, server.js

(2) [Serverless]
↓
Cloud Run
↓
Deploy one revision from existing container image
[Select image]
→ O continuously deploy new revisions from a source repository
↓
CPU allocation & Pricing (as required)
↓
Autoscaling (as required)

Push the file to Registry
$gcloud build submit -t gcr.io/$DEVSHELL_PROJECT_ID/<foldername>
folder name

~~$gcloud build submit -t gcr.io/$D~~

(3) [CI/CD]
↓
container registry

☆ <Image is run>

(4) To deploy new app

vi server.js
:
change <msg>

$gcloud builds submit -t gcr.io/$DEVSHELL_PROJECT_ID/<foldername>:version2 (or) ver2

(d) Cloud Run ☆ (no create version of same service)

↓

Service (existing)

↓

Edit & deploy new revision

☆ cloudshell

(6) [$ gcloud beta interactive]

$ gcloud run deploy <yyservice> --image=<imgname>

Vendor optional.

<-- allow-unauthenticated -- platform managed

-- region=<asia-southeast1> <-- project = <projectID>

> deleting

Cloud Run -> delete services

Storage -> delete buckets

Lec - 45

Google Cloud functions Serve [FaaS]

-> with cloud functions you write simple, single-purpose functions that are attached to Events emitted from your cloud infrastructure & services.

-> A pay-as-you-go functions as a Service

FaaS -> (FaaS) to run your code with zero server management.

-> cloud functions can be written using

(a) Node.js (b) Python 3

(c) Go (d) Java

-> Automatically scales based on load without thinking about the infrastructure.

-> Built-in security at scale & per function level based on the least privilege principle.

AWS - lambda
Azure - Azure functions

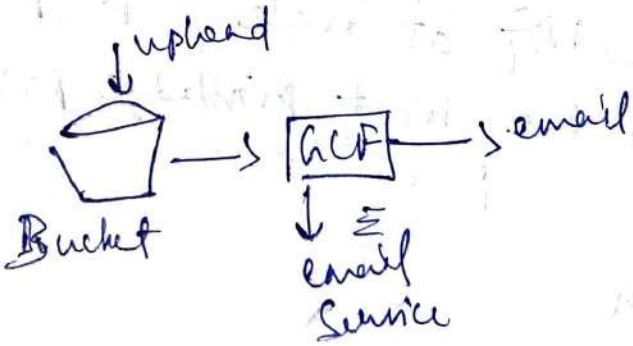
-> Allows you to trigger your code from Google cloud, Firebase, & Google assistant or call it directly from any web, mobile or backend applications via HTTP.

→ To act on events, you shall define a trigger. Binding a function to a trigger enables you to act on events. The most common triggers are :-

1. Google Cloud Storage
2. Pub/Sub events
3. HTTP

→ Pricing → GCF pricing acc. to

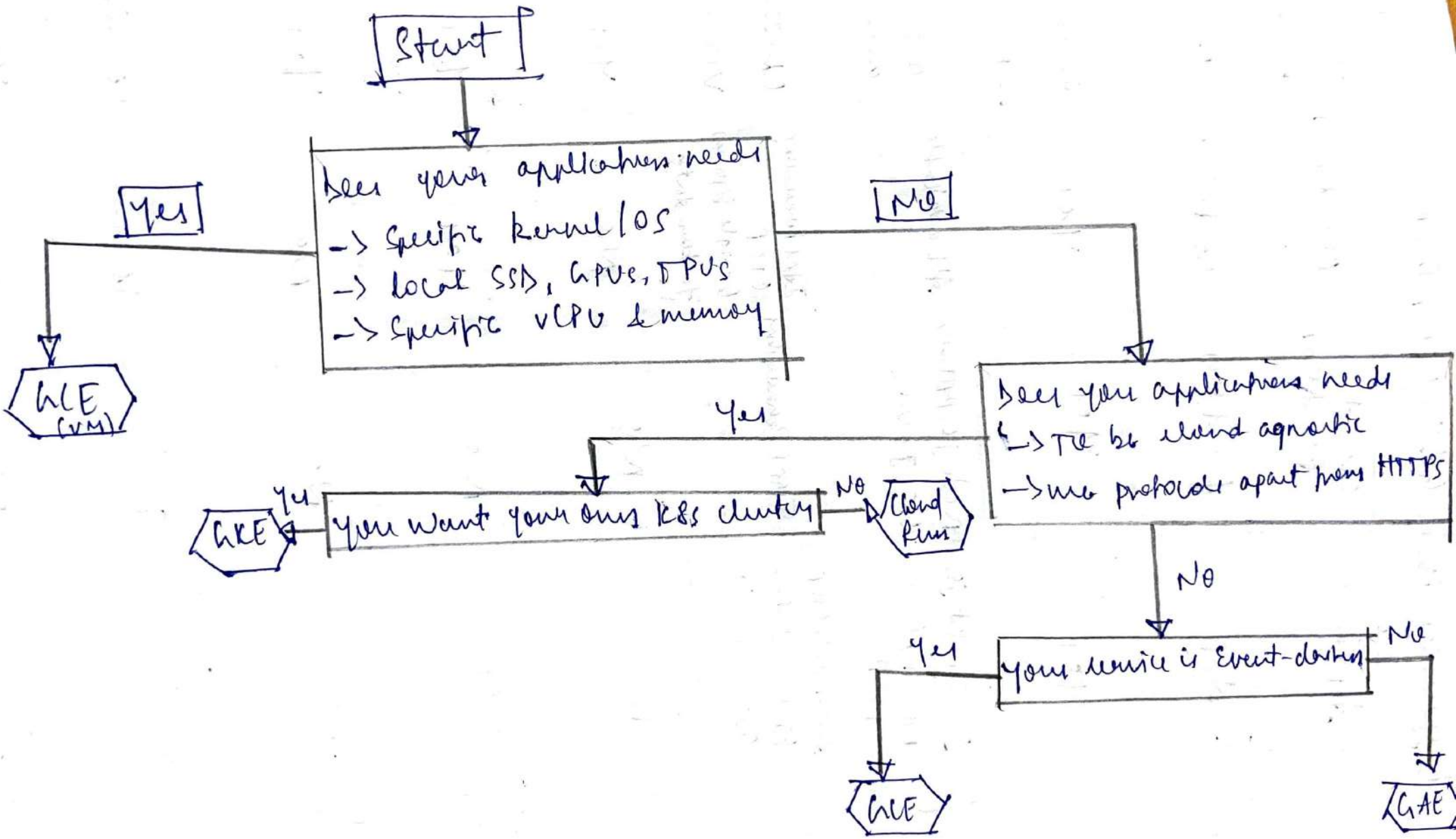
(a) How long your function runs
(b) How many times it is invoked
(c) How many resources you provision for the function.



(1 trigger = 1 invocation)

* (Only used for single micro-service processing)

How to Select Serverless Environment



Differences b/w GAE, GCF, GCR

Ex

	App Engine	Cloud functions	Cloud Run	Cloud Run for Anthos
Deployment type	App	functions	Containers	Containers
Scaling	Yes	Yes	Yes	only using Pods
Runtime Support	Java, Node.js, Python, Go, PHP	Node.js, Python, Go, Java	All available	All available
Access Control	Auth 2.0, CICP Firebase Auth, Google sign-ins	IAM Permissions	IAM Permissions CICP Google sign-ins Firebase Auth	Cluster only VPC-only
HTTP/gRPC	No	No	No	Yes
custom domains	Yes	No	Yes	Yes
GPU/TPU Support	No	No	No	Yes
Timeout	1 min	9 min	15 min	15 min
VPC networking	Yes (Std → No Flex → Yes)	Yes	No	Yes

Demo

[Serverless]

(1)

↓

Create functions

↓

Basics : 1st gen

function name : function-1

Region : US-central1

↓

Trigger

Trigger type : ~~HTTP~~

Authentication : ⊙ All unauthenticated invocations

○ Require authentication

↓

NEXT

↓

Code

↓

+ Runtime : <language>

+ Source code : < can upload > X

↓

[Deploy]

(2) function details

↓

Trigger

↓

URL

[Delete]

(a) Delete environment ✓

Lec - 46 Google Cloud SQL (Regional Service)

- Amazon - RDS
- Azure SQL Database

Database
- Relational
- Non-Relational

→ Cloud SQL is a fully managed Relational database service for MySQL, PostgreSQL & SQL server. ✓

→ Cloud SQL is fully hosted on Linux ✓

→ Cloud SQL is a webservice that allows you to create, configure, ~~configure~~ & use relational databases that live in Google's cloud. ✓

→ Any existing databases in MySQL, PostgreSQL or any other structured database can be migrated to cloud SQL. ✓

→ Google handles replication, patch management & database management to ensure availability & performance. ✓

→ fully managed MySQL (5.6, 5.7 & 8.0) ✓
PostgreSQL (9.6, 10, 11, 12, 13 & 14) & ✓
SQL server (2017, 2019) ✓

→ Easily scale upto 96 vCPU & 624 GB of
☆ RAM (vertical scalable) ✓

☆ → Upto 64 TB of storage available ✓

→ Support for private IP ✓

-> Data replication b/w multiple zones with automatic failover.

-> Import & export databases using mysqldump, or import and export CSV files.

-> Instance cloning.

* To use cloud SQL effectively, you should understand some of the concepts on which it is built.

① Cloud SQL Instance -> A cloud SQL instance corresponds to one VM. The VM includes the database instance & accompanying software (MySQL, PostgreSQL) containers to keep the database instance up & running. (DB instance + S/W)

② Database instance -> A database instance is the set of software & files that operate the database: MySQL, PostgreSQL or SQL server.

③ Public & private IP -> In cloud SQL, public IP means that the instance is accessible through the public internet, but are accessible through a VPC. Cloud SQL instances can have both public & a private IP.

④ High availability -> HA in cloud SQL works by having two sql instances: Primary & Standby.

* Each instance has exactly one VM. Each instance is in a different zone in the same region.

(5) Failover -> A failover is when cloud SQL switches from the original primary instance to the standby instance.

(6) Standby instance -> The standby instance is used in H.A to replace the primary instance when failover occurs. The standby instance doesn't appear in the Google Cloud console. When failover occurs, connections to the primary instance are automatically transferred to the standby instance.

Clone -> When you clone a cloud SQL instance, you create a new instance that is a copy of the source instance, but is completely independent. After cloning is complete, changes to the source instance are not reflected in the clone, & changes in the clone are not reflected in the source instance.

Replication -> Replication is the ability to create copies of a cloud SQL instance or an on-premises database, & offload work to the copies.

The main reasons for using replication is to scale the use of data in a database without degrading performance on the primary instance.

Database Administration

- Cloud SQL lets you create and delete databases & databases users, but it is not a database administration tool.
- There are many database administration tools you can choose from depending on your database engine, including the following :-

① Phpmyadmin ② Toad Edge
③ MySQL workbench ④ SQL server management Studio

Cloud SQL Pricing

-> How much storage you provision
-> How many vCPU you select
-> How much RAM you select
-> Where to choose to host your data (Regions)
-> How much network traffic leaves your instance
-> How many IP addresses you use.

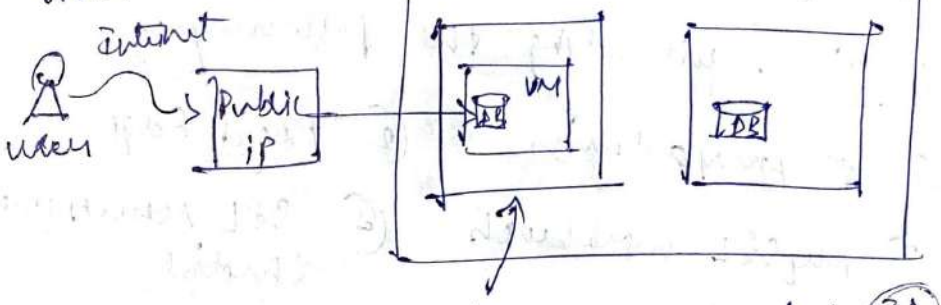
Cloud SQL updates

① Configuration updates -> Google
② System Updates -> user managed

[Note] Vertical Scaling is used in CloudSQL, we can scale storage, vCPU & RAM, but we cannot create a group of SQL instances, because Cloud SQL is regional service. Hence we create SQL instances in 1 region & as per google documentation.

Lec-47 Google Cloud SQL Demo

GCP console
↓
Databases
↓
SQL
↓
create → MySQL



Authorized network ← IP
Permission

Database
↓
SQL
↓
create instance
↓
choose database engine

+ MySQL ✓ + PostgreSQL + SQL Server
↓
Instance info
Instance ID : <Th>
Password : Generate
↓
+ Database version
MySQL 8.0 ✓
+ Minor version
Default ✓

↓

O Production

O Development ✓

↓

Choose region & zonal availability

O Single Zone

O Multiple Zone ✓

+ Region : <any>

Primary Zone

<any>

Secondary Zone

<any>

↓

Customize your instance

Machine type : Standard ✓

1 vCPU, 3.75 GB

↓

+ Storage : SSD

+ Storage capacity : 10 GB

↓

Connections

+ Private

+ Public ✓

↓

Keep everything default

↓

[Create]

wait upto 7 mins

↓

SQL

↓

connections

↓

Add Network

+ Cloudshell

+<ip>

Cloudshell

↓

☆

$ mysq --version

8.0.32

$mysql -h <public ip>

-u root -p

$ help

$ show databases;

$ use <databases>;

To know ip → ipify (google search) → code samples

→ Bash → run both commands

ip = $(curl -s https://api.ipify.org)

echo "My public IP address is: $ip"

↓

Save

Delete

SQL → Instance → delete

Lec-48

Cloud Spanner (Horizontal scaling)

-> A fully managed relational database service with unlimited scale, strong consistency (mission critical) & upto 99.999% availability.

-> Yearly downtime is 5 min/year

-> In AWS, AWS Aurora is available

-> Cloud spanner launched in 2017

-> Provides transparent, synchronous replication across region & multi region configuration.

-> Optimizes performance by automatically sharding the data based on request load & size of data so you can spend less time thinking about scaling your database & more time scaling your business.

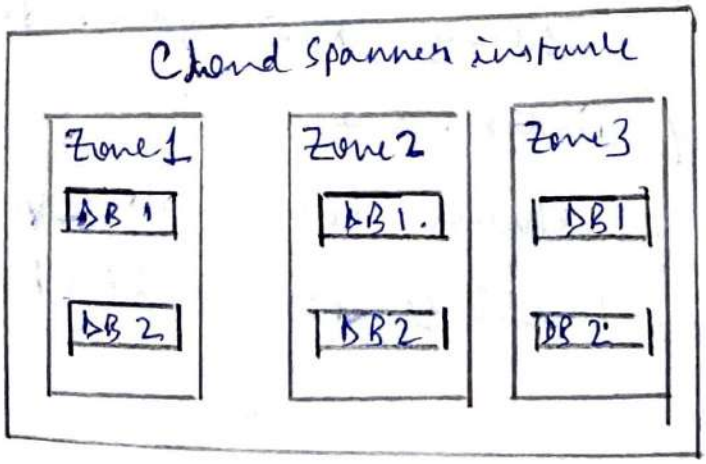
-> You can run instances on a regional scope (or) multi-regional where your database is able to survive regional failure.

-> Cloud spanner is highly recommended to use when data is 2TB

-> Used for Gaming, Logistics, e-commerce, Finance etc.

-> Billing hourly [Mission critical = Sync service]

Spanner Distributes:



	Cloud Spanner	Traditional Relational (or) SQL	Traditional Non-Relational (or) NO-SQL
Schema	Yes	Yes	No
SQL (Structured Query lang.)	Yes	Yes	No
Consistency	Strong	Strong	Eventual
Availability	High	Failover	High
Scalability	Horizontal	Vertical	Horizontal
Replication	Automatic	Configurable	Configurable

Cloud Spanner = ACID Compliant

A -> Atomicity
C -> Consistent
I -> Isolation
D -> Durability

☆

Pricing

-> Number of nodes in your instance
-> Amount of storage that your tables & secondary indexes use.
-> Amount of network bandwidth used
-> No additional charge for replication.

Limitations of Cloud Spanner:

1. Lift & Shift Scenario. (AWS ⇔ GCP)
2. Lots of in-database business logic (triggers & stored procedures)
3. When you need less latency.

Lec-49 Cloud Spanner - Demo

GCP console
↓
Databases
↓
Cloud Spanner → Regional $0.90/hr
→ multi Regional $3/hr
↓
free-tier - 90 days
Storage - 10GB

** 1 node = 4TB = 1000 processing units

* Globally consistent ✓

---

(1) Databases
↓
Spanner
↓
Start a free trial
↓

+ Instance Name : myspanner
+ Instance ID : —"—
+ choose a configuration : Delhi - asia-south2 ✓
Jakarta - asia-southeast2
Frankfurt - europe-west3
Columbus - us-east5

↓

Create instance

(2) create one more database

↓

create my own

↓

+ Database name : mydb1

+ Select database dialect

- (1) Google std. SQL ✓
- (0) Postgre SQL

↓

Define your schema (optional)

(ship)

↓

create

---

(3) Overview

↓

create Table → Google

↓

< Text > see documentation.

LL-50

Cloud Bigtable (No SQL non-relational)

* DynamoDB → No SQL in AWS

-> Cloud Bigtable is a fully managed, wide-column No-SQL database that offers low latency & replication for high availability.

-> Single digit millisecond latency, limitless scale & 99.999% availability for large analytical & operational workloads.

-> Easily migrate from Apache HBase & Cassandra to Bigtable with no downtime.

-> Bigtable is a sparsely populated table that can scale to billions of rows & thousands of columns.

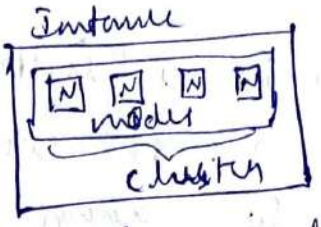
-> Bigtable supports storage of terabytes or even petabytes of data.

-> Bigtable is not a relational database. It does not support SQL queries, joins or multi-row transactions.

-> Data replication can be performed by simply adding a second cluster to the instance & replication starts automatically.

-> Bigtable integrates easily with big data tools like Hadoop, Dataflow, Dataproc & supports HBase APIs

-> Bigtable stores data in Colossus, Google's file system.



Instance :-

-> A Bigtable instance is a container for data.

-> Instance have one or more clusters located in a different zone & different region.

-> Each cluster has atleast 1 node

* -> A Table belongs to an instance & not to the cluster or node.

-> An instance also consists of the following properties :-

(a) Storage type - SSD or HDD

(b) Configuring cluster ID, selecting Region & Zone

(c) Node scaling mode - Manual or Auto

(d) Encryption -> GMEK
-> CMEK

(e) Add Replication Cluster.

CLUSTER

-> cluster handles the requests sent to a single Bigtable instance.

-> Each cluster belongs to a single bigtable instance & an instance can have upto 4 clusters.

-> Each cluster is located in a single zone

-> Bigtable instances with only 1 cluster do not use replication.

Nodes

-> Each cluster in an instance has 1 or more nodes, which are the compute resources that Bigtable uses to manage the data.

-> Each node in the cluster handles a subset of the request to the cluster.

-> All client requests go through a front-end servers before they are sent to a Bigtable node.

-> Bigtable separate the compute from the storage. Data is never stored in nodes themselves, each node has pointers to a set of tablets that are stored on colossus.

-> Recovery from the failure of a bigtable node is very fast as only the metadata needs to be migrated to the replacement node.

-> When a bigtable node fails, no data is lost.

Bigtable cluster can be scaled out by adding nodes which would increase:

- The no. of simultaneous req. that the cluster can handle.
- The max. throughput of the cluster

-> Bigtable nodes are sometime referred as tablet servers.

-> Bigtable has a limit of 1000 tables per instance.

Primary key

Singer ← Table

column family

row key	Personal-CF		album-CF	
	first name	last name	album name	year
Rafi mukesh	Mohammed Mukesh	Rafi	Nagmein Awara	1958

column qualifier

Lec-51 Bigtable - Demo

$468 per month
$0.65/hr with 0 hrs shared

console
↓
Databases
↓
Bigtable
↓
create instance

+ name :
+ id :

[Continue]

O SSD → real time
O HDD → machine learning / data mining

↓
Select a location

+ Region : ___ + Zone : ___

↓
Choose node scaling mode

O manual allocation
+ Quantity : 1
O Autoscaling
[1 node = 16 TB]

↓
Encryption
O GMCK
O CMEK

↓
Advanced options
+ Add replication
↓
Select a location
+ Region + Zone → [CREATE]

Instance
↓
Cloudshell
$ gcloud components list
(check Bigtable CLI)
$ ls -a
~~$ rm -rf .c~~
$ echo project = $DEVSHELL_PROJECT_ID >
- /.cbtrc

$ echo instance = instance-id >> -/.cbtrc

$ cat .cbtrc
$ cbt ls (to list table)
$ cbt createtable singer (create table)
$ cbt ls
$ cbt createfamily singer personal_cf
$ cbt createfamily singer albums_cf
$ cbt ls singer
→ family name GC-policy
albums_cf <never>
personal_cf <never>

adding data
$ cbt set singer rafi personal_cf:firstname
= mohammed
$ cbt set singer rafi personal_cf:lastname=
rafi
$ cbt set singer rafi albums_cf:albumname=
'Nagmein'

Delete Instance & Table
$ cbt read singer
— details are shown —

→ Can create another singer detail → change name & add entries

$ cbt read singer
— details of both singers is seen —
→ $ cbt deletetable <tablename>

Lec-52 Cloud Memorystore

→ Memorystore is a fully-managed database service that provides a managed version of two popular open source caching solutions ☆ Redis & Memcached

Memory store for Redis provides a fully-managed service that is powered by the Redis in-memory data store to build applications caches that provide sub-millisecond data access.

RAM = SSD / HDD

→ Memorystore for Redis offers several advantages over self-managed Redis:

① Deploy what fits your needs. → Basic (no replica) → Std. (replica) 99.9% avail.

② Easily scale to get blazing speed.

③ Highly available & more secure (IAM & in-transit encryption)

④ Focus on your application.

⑤ Redis protocol compatible.

→ It is good for caching, gaming & stream processing.

features of memorystore for Redis:

1. fully managed
2. Simple deployment
3. High availability (99.9%)
4. Enterprise Grade Security (no public ip)
5. Scale, as needed
6. Monitoring
7. On-demand billing (300GB/node)-RAM
8. Redis versions 6.x, 5.0, 4.0 & 3.2

You can use any standard Redis client on the following environments to connect to your memorystore for Redis Instance.

1. App engine flexible/Std. environment
2. Compute engine VM instance
3. Cloud functions
4. Cloud Run
5. GKE Clusters

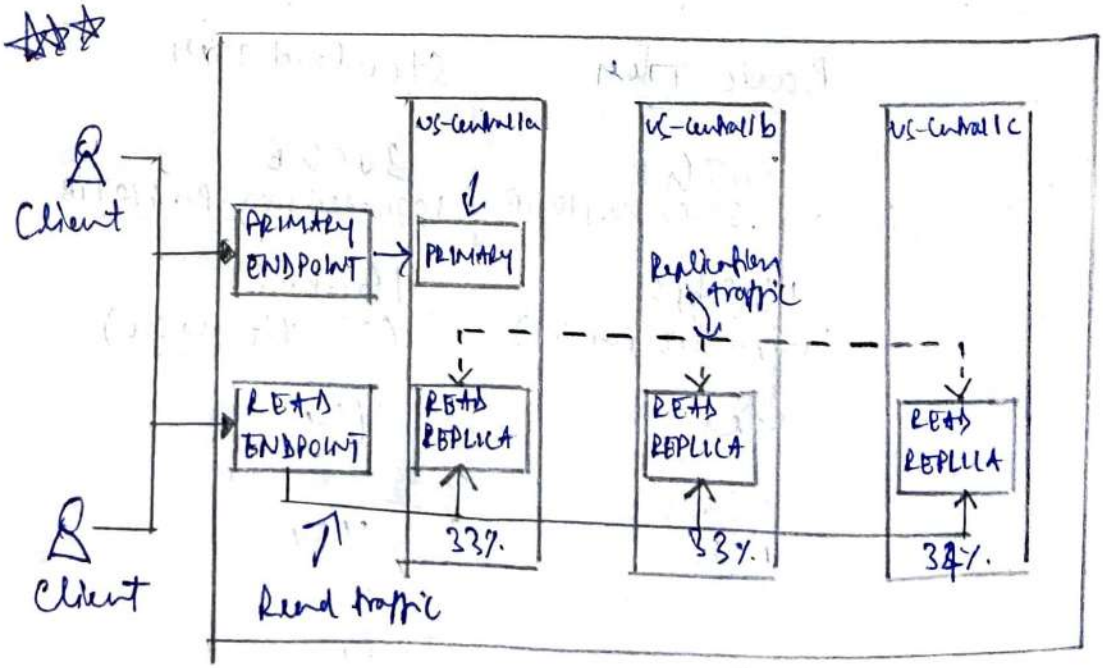
Spec.	Basic Tier	Standard Tier
max. Redis Node size	300GB 100nodes; 2000vCPUs; 10TiB	300GB 100nodes; 2000vCPUs; 10TiB
max. network Bandwidth	16Gbps (Giga bit per sec)	16Gbps (Giga bit per sec)
Scale node size	Yes	Yes
Cross-Zone Replication	No	Yes
Automatic failover	No	Yes
Redis-Replica	No	Yes

Read Replica Behaviour:

-> Read replica are not enabled on standard tier instances by default.
-> Once read replica is enabled, you can't disable it.
-> Std. tier instances can have 1 to 5 read replicas.

Limitations:

-> Read replicas are only supported for instances using with model >= 5 GB
-> Read replicas can only be enabled on instances that use Redis version 5 or higher.
-> Each instance has a primary endpoint & read endpoint. The primary endpoint always directs traffic to the primary node, while the read endpoint automatically load-balances read queries across available replicas.



Failover

-> When a primary fails, the memorystore health monitoring service initiates the failover & the new primary is made available for both read & writes. The failover usually complete in under 30 seconds.

-> When a failover happens the primary endpoint automatically redirect traffic to the new primary. However, all client connections to the primary endpoint are disconnected during a failover. Applications with connection retry logic will automatically reconnect once the new primary is online.

Zone failure

If the zone where the primary is located fails, the primary automatically fails over to a replica in another zone. If the instance has only one replica, then the read endpoint is unavailable for the duration of the zone outage. If the instance has more than one replica, replicas outside of the affected zone are available for reads.

-> If the zone where one or more of the replicas is located, fails, those replicas are unavailable for the duration of the zone failure. If there is a two zone failure & there are two or more replicas, the replica with least lag in the remaining zone is made primary.

Full Sync

→ Maintenance on Memorystore for Redis:

* Most of these updates are performed while your redis instance is up & running. However, certain system updates require a brief service interruption to be performed.
* These updates are called maintenance.

→ Maintenance typically falls into the following category:

① **Memorystore features** – To launch new features, memorystore requires a maintenance update.

② **Operating System Patches** – We continuously monitor for newly identified security vulnerability in the O.S, upon discovery, we patch the O.S to protect you from new risks.

③ **Database version upgrade** – Maintenance can include an OSS Redis patch minor version update which does not include breaking or incompatible changes.

→ The duration of the maintenance window is 1 hour. (Can be scheduled & re-scheduled)

→ Maintenance window is not specified for your instance, maintenance can begin at any time without advanced notification.

→ Version upgrade Behaviour

Data is preserved during a version upgrade operation for both Basic & standard tier instances. During an upgrade operation the instance undergoes a short connection reset of a couple minutes or less

+ The IP address & connection string remains the same.

Import & Export data:

Import & export uses Amazon Storage buckets to store RDB files. (.rdb) file.

→ An instance can only import backups from older Redis versions.

Import - from physical DB to ElastiRedis ; Export - from Redis to On-prem.

Import feature Behaviour:

→ For optimal performance, you should import from Amazon storage buckets located in the same region as your redis instance.

→ Your redis is unavailable during an import operation. Your instance become available after the import operation finishes.

→ If the import is successful, the existing data in the instance is overwritten by the data in the RDB file.

→ If the import fails, the instance is brought online, but the data may be fully flushed. You can retry the import using the same RDB file, or you may use another RDB file to restore data.

Resource Intensive workloads:

For standard Redis Enterprise, the following operations use extra memory for the duration of the operation.

1. Version upgrade
2. Scaling up/down
3. Manual failover
4. Import/Export

All these operations and scenarios break the network connection & requires connection retry.

LU-53 Demo – Memorystore for Redis 30/3/23

Login to GCP Console -> Database -> Redis -> enable API

Now [Create Instance]

Instance ID ->

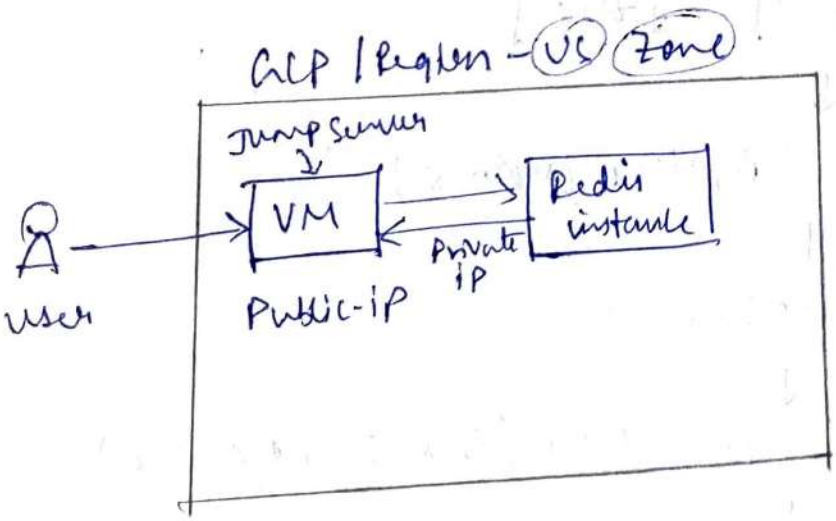
Display NAME ->

TIER SELECTION -> Basic
Std.

Capacity -> Standard (min - 5GB ; max - 300GB)
Basic (min - 1 ; max - 300)

Select Region & Zone [Zone specific]

Read Replica -> 0 to 5

Set up connection -> VPC

Maintenance -> Preferred window