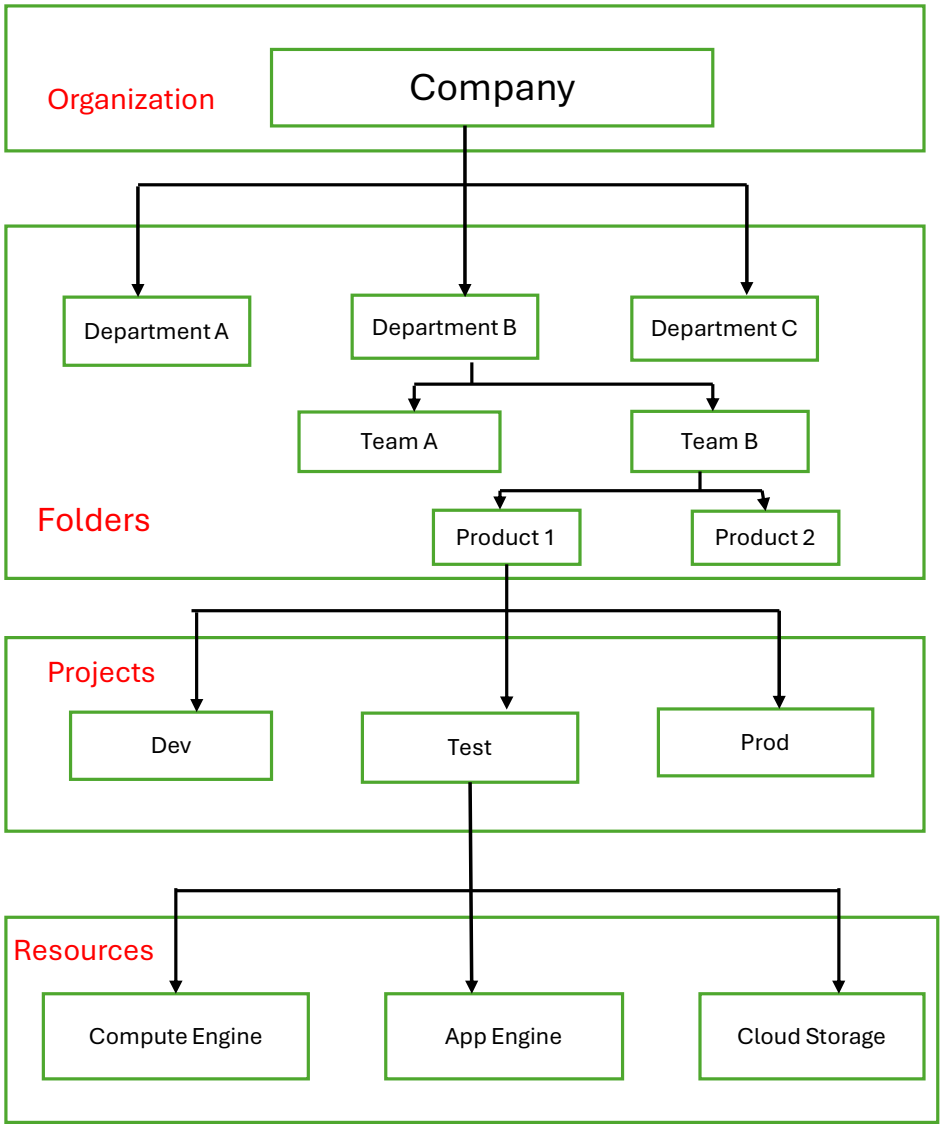
GCP Certification Full course in Hindi

Associate Cloud Engineer

DEVOPS MADE SIMPLE

Youtube channel link- Google Cloud Platform (GCP) Full Course 2025 - Hindi | Associate Cloud Engineer Certification - YouTube



Policy inheritance

Google Cloud has container resources—such as **projects, folders,** and **organizations**—that let you organize your resources in a parent-child hierarchy. This hierarchy is called the resource hierarchy.

The Google Cloud resource hierarchy has the following structure:

- The **organization** is the root node in the hierarchy.
- **Folders** are children of the organization, or of another folder.
- **Projects** are children of the organization, or of a folder.
- **Resources** for each service are descendants of projects.

Identity and Access Management (IAM)

IAM lets you control who can do what on which resources.

Giving someone permissions in IAM involves the following three components:

- **Principal:** The identity of the person or system that you want to give permissions to
- **Role:** The collection of permissions that you want to give the principal
- **Resource:** The Google Cloud resource that you want to let the principal access

To give the principal permission to access the resource, you grant them the role on the resource. You grant these roles using an allow policy.

Principals - Principals represent one or more identities that have authenticated to Google Cloud.

Principal types-

- Google Accounts
- Service accounts
- Google groups
- Google Workspace accounts

Permissions and roles

- Permissions determine what operations are allowed on a resource.
- You can't directly grant permissions to a principal. Instead, you give principals permissions by granting them roles.
- Roles are collections of permissions. When you grant a role to a principal, you give that principal all of the permissions in that role.

There are three types of roles:

1. **Predefined roles:** Roles that are managed by Google Cloud services. These roles contain the permissions needed to perform common tasks for each given service
2. **Custom roles:** Roles that you create that contain only the permissions that you specify. You have complete control over the permissions in these roles.
3. **Basic roles:** Highly permissive roles that provide broad access to Google Cloud services. These roles can be useful for testing purposes, but shouldn't be used in production environments

Resources

Most Google Cloud services have their own resources. For example, Compute Engine has resources like instances, disks, and subnetworks.

In IAM, you grant roles on a resource. Granting a principal a role on a resource means that the principal can use the permissions in that role to access the resource.

To create a custom role using a YAML file:

title: "custom role 1"
description: "role to test gcloud command"
stage: "ALPHA"
includedPermissions:
- compute.instances.create
- compute.instances.delete

- save this as YAML file and upload it to cloudshell

*gcloud iam roles create ROLE_ID --project= PROJECT_ID *
--file=YAML_FILE_PATH

To create a custom role using flags:

gcloud iam roles create ROLE_ID --project=PROJECT_ID \
--title=ROLE_TITLE \
--description=ROLE_DESCRIPTION \
--permissions="PERMISSIONS_LIST" \
--stage=LAUNCH_STAGE

gcloud iam roles create vmdeleteflag --project=PROJECT_ID \
--title="gcloud command flag“ \
--description="creating role with gcloud command" \
--permissions="compute.instances.delete" \
--stage="ALPHA"

To update a role using YAML file -

Add/remove permissions and save the YAML file, and then execute following command-

*gcloud iam roles update ROLE_ID --project=PROJECT_ID *
--file=YAML_FILE_PATH

To update a custom role using flags:

*gcloud iam roles update Role_ID *
--add-permissions="compute.instances.suspend"

*gcloud iam roles update Role_ID *
--remove-permissions="compute.instances.suspend"

IAM Policy

- Policy defines and enforces what roles are granted to which principals.
- You can grant roles to the users by creating an allow policy, which is a collection of statements that define who has what type of access.
- A policy is a collection of bindings.
- A binding associates one or more members with a single role and any context-specific conditions that change how and when role is granted.

Each binding includes the following fields-

1.Member- known as identity or Principal

- User account
- Service Account
- Google Group

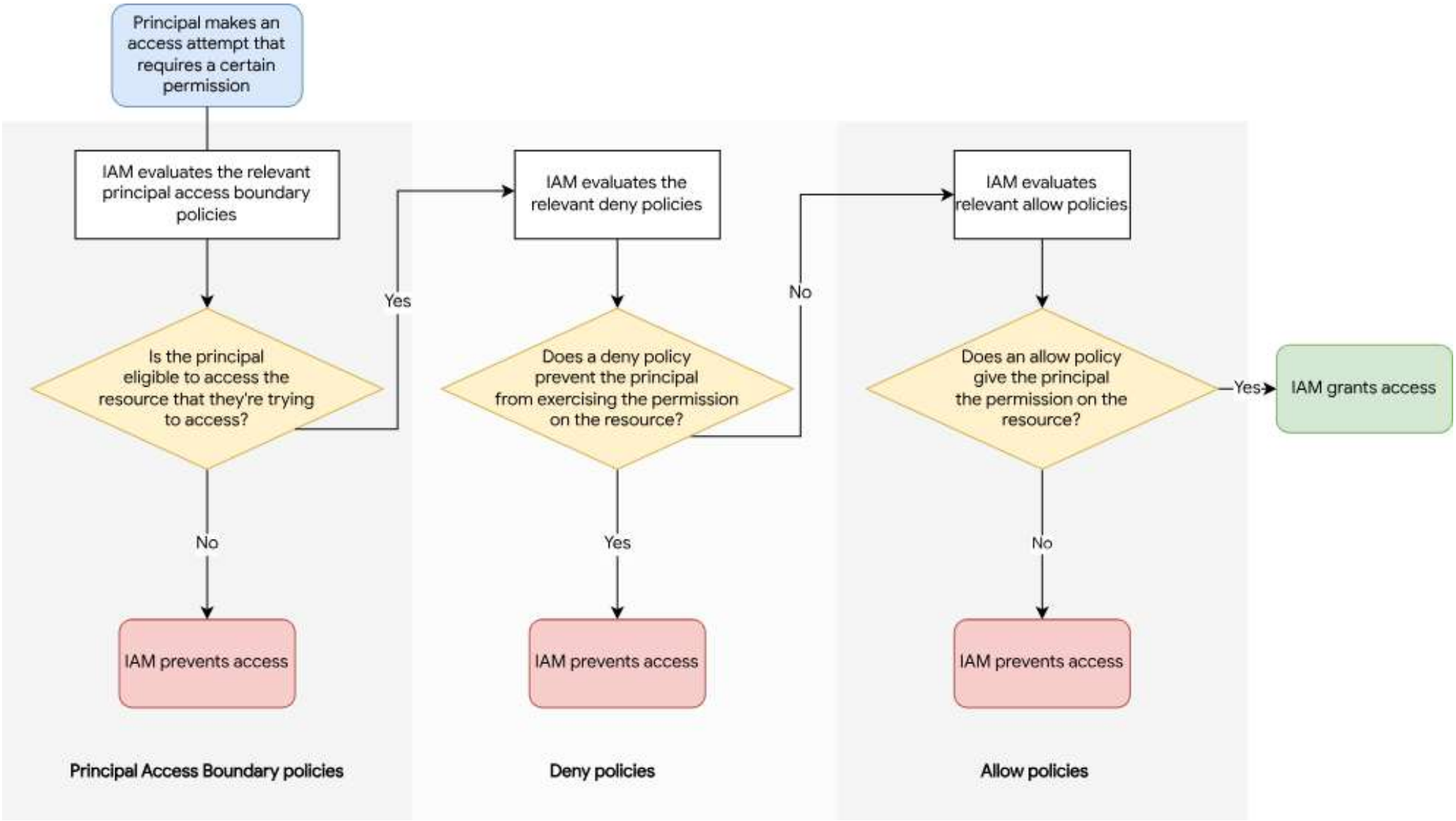
2.Role - named collection of permissions that grant access to perform actions on Google Cloud Resources

3.Condition - a logical expression that further constraints the role binding based on attributes about the request, such as its origin, the target resource etc.

Policy Types-

IAM offers the following types of policies:

- **Allow policies -** allow policy contains a list of *role bindings* that associate IAM roles with the principals who are granted those roles
- **Deny policies -** Deny policies prevent principals from using certain permissions, even if they're granted a role with the permission.
- **Principal access boundary (PAB) policies -** Principal access boundary policies define and enforce the resources a principal is eligible to access. Principals can't access resources that they're not eligible to access, even if they've been granted a role on the resource.


Principal makes an access attempt that requires a certain permission
IAM evaluates the relevant principal access boundary policies
Is the principal eligible to access the resource that they're trying to access?
Yes
No
IAM prevents access
Principal Access Boundary policies
IAM evaluates the relevant deny policies
Does a deny policy prevent the principal from exercising the permission on the resource?
Yes
No
IAM prevents access
Deny policies
IAM evaluates relevant allow policies
Does an allow policy give the principal the permission on the resource?
Yes
IAM grants access
No
IAM prevents access
Allow policies

Add a role binding to a policy:-

- **get-iam-policy -** to get the current allow policy for the resource

gcloud RESOURCE_TYPE get-iam-policy RESOURCE_ID --format json/yaml/text

- **add-iam-policy-binding -** to grant a role to a principal

*gcloud RESOURCE_TYPE add-iam-policy-binding RESOURCE_ID *
*--member=PRINCIPAL --role=ROLE_NAME *
--condition=CONDITION

- **remove-iam-policy-binding** - to revoke a role from a user

gcloud RESOURCE_TYPE remove-iam-policy-binding RESOURCE_ID
--member=PRINCIPAL --role=ROLE_NAME

Service Accounts

- A service account is a special kind of account typically used by an application or compute workload, such as a Compute Engine instance, rather than a person.
- A service account is identified by its email address, which is unique to the account.
- When an application authenticates as a service account, it has access to all resources that the service account has permission to access.
- The most common way to let an application authenticate as a service account is to attach a service account to the resource running the application.

Where to create service accounts

- Each service account is located in a project. After you create a service account, you cannot move it to a different project.

- There are a few ways to organize your service accounts into projects:

Create service accounts and resources in the same project-

- This approach makes it easier to get started with service accounts. However, it can be difficult to keep track of your service accounts when they are spread across many projects.

Centralize service accounts in separate projects

- This approach puts all of the service accounts for your organization in a small number of projects, which can make the service accounts easier to manage. However, it requires extra setup if you attach service accounts to resources in other projects, which allows those resources to use the service account as their identity.

- When a service account is in one project, and it accesses a resource in another project, you usually must enable the API for that resource in both projects.

- **For example-** if you have a service account in the project my-service-accounts and a Cloud SQL instance in the project my-application, you must enable the Cloud SQL API in both my-service-accounts and my-application.

Types of service accounts

In Google Cloud, there are several different types of service accounts:

- **User-managed service accounts:** Service accounts that you create and manage. These service accounts are often used as identities for workloads.
- **Default service accounts:** User-managed service accounts that are created automatically when you enable certain Google Cloud services. You are responsible for managing these service accounts.
- **Service agents:** Service accounts that are created and managed by Google Cloud, and that allow services to access resources on your behalf.

Create service accounts with gcloud commands-

To create the service account

```
gcloud iam service-accounts create SERVICE_ACCOUNT_NAME \
  --description="DESCRIPTION" \
  --display-name="DISPLAY_NAME"
```

To grant your service account an IAM role on your project

```
gcloud projects add-iam-policy-binding PROJECT_ID \
  --member="serviceAccount:SERVICE_ACCOUNT_NAME@PROJECT_ID.iam.gserviceaccount.com" \
  --role="ROLE_NAME"
```

Assign Service Account to VM instance

```
gcloud compute instances set-service-account VM_NAME \
    --zone=ZONE-NAME \
    --service-account=SERVICE_ACCOUNT_EMAIL \
    --scopes=https://www.googleapis.com/auth/cloud-platform
```

Service account impersonation

- When an authenticated principal, such as a user or another service account, authenticates as a service account to gain the service account's permissions, it's called impersonating the service account.
- Impersonating a service account lets an authenticated principal access whatever the service account can access.
- Impersonation is useful when you want to change a user's permissions without changing your Identity and Access Management (IAM) policies.

You can use impersonation to-

- temporarily grant a user elevated access
- to test whether a specific set of permissions is sufficient for a task
- locally develop applications that can only run as a service account
- to authenticate applications that run outside of Google Cloud.
- To impersonate a service account: Set the --impersonate-service-account flag when running a Google Cloud CLI command.
- To impersonate a Service Account, User should have *Service Account Token Creator* IAM Role.