Course Navigation

Course Intro
Section 1

IAM Big Picture
Section 2

Establishing Identity
Section 3

Granting Access to GCP Resources with Cloud IAM
Section 4

Service Accounts
Section 5

Next Sections

Back to Main

Course Introduction

Purpose of this IAM Deep Dive course

Core competency for properly managing access to your GCP environment

- Foundational skill set for properly managing who has access to what resources
  - Limit scope of access to only those who need it
- Essential building block for all GCP certifications

What will we cover?

- Managing organization accounts with G Suite and Cloud Identity
- Member access to GCP resources with IAM Roles within the Cloud Resource Hierarchy
- Manage service accounts, which are used by application and machines
- How to avoid unintended public exposure of Cloud Storage resources
- Keeping track of who did what, and when with Audit Logs
- And much more

Prerequisites

- Familiar with the 'basics' of working with GCP
  - Creating projects
  - Create and connect to GCE instances
  - Create and work with Cloud Storage buckets
- Recommend courses:
  - GCP Concepts
  - GCP Essentials

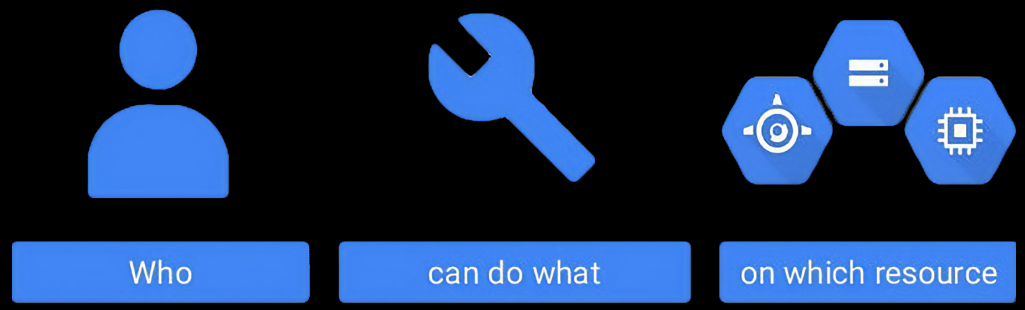
What is Identity & Access Management?

Technical Definition

With Cloud IAM, you can grant granular access to specific GCP resources and prevent unwanted access to other resources. Cloud IAM lets you adopt the security principle of least privilege, so you grant only the necessary access to your resources.

How it breaks down:

- **Member** = Who
- **Role** = Can do what
- On which **resource** = All resources across all levels of GCP (projects, organizations, cloud storage buckets, etc.)
- Combined, the **'who can do what on which GCP resource'** forms an **IAM Policy**



Course Navigation

Course Intro
Section 1

Member Types

IAM Big Picture
Section 2

Creating Identities with Cloud Identity

Establishing Identity
Section 3

Hands-On - Cloud Identity

Granting Access to GCP Resources with Cloud IAM
Section 4

Synchronize Cloud Identity with Active Directory

Service Accounts
Section 5

Google Groups

Configure Multi-Factor Authentication

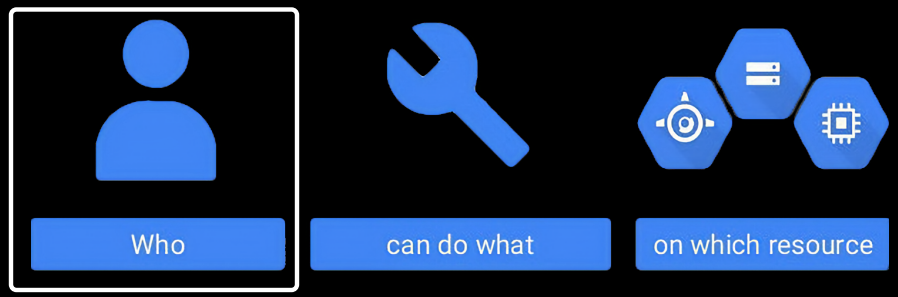
Next Sections

Back to Main

Member Types - the "Who"

Who = Member = Identity

- A person or a service account
- Identity format = email address
  - Log in with an email address (joe@linuxacademy.com)
- 'People' accounts authenticate via **Google account** (in the form of an email address)
  - E.g., matt@gmail.com, matt@linuxacademy.com (corporate Google identity/domain)
- Service account = application/server account
  - Not associated with a person
  - Also in email format, but not an actual email address



Account creation and management - two different sources

- **'People' accounts** = created and managed in **Google Admin Console** (for corporate domains)
  - Not part of GCP
- **Service accounts** (for applications) created and managed in GCP

'People' account types

Personal - joe@gmail.com
- Non-Gmail accounts associated with Google account (joe@yahoo.com)
- Unmanaged account

G Suite/Cloud Identity Domain - primary section focus
- **Managed** organizational Google accounts
- Both types managed in Google Admin Console
- Represents company Internet domain name (companyname.com)
- Represents a virtual group of all Google accounts in an organization
- **G Suite**
  - Uses company-wide Google apps (Gmail, Drive, etc.)
- **Cloud Identity**
  - No access to G Suite applications (Gmail, Drive, etc.)
  - Can sync existing directory service (e.g., Active Directory) to Cloud Identity
- Note: GCP IAM does not differentiate between G Suite and Cloud Identity accounts

Google Group - Collection of Google accounts
- Also (recently) managed in Admin Console as well as legacy groups.google.com page
- Also represented by email addresses (googlegroup@companyname.com)
- Apply a single policy to the collection of users in the group
- No login credentials—not used to establish an individual identity
- Can include Service Accounts

Special account types (varies by service)
- **allAuthenticatedUsers**
  - Special identifier to represent any Google account
  - Anonymous visitors not included
- **allUsers** (i.e., 'public')
  - Another special identifier
  - Anyone and everyone (including anonymous visitors)
  - Both types managed in GCP
  - Not represented as an email account

Cloud Identity/G Suite

What is it?

Identity the domain to establish Google managed, organizational accounts for GCP, and much more

- Example: matt@linuxacademy.com - linuxacademy.com is the organization domain for corporate Google accounts.

G Suite vs. Cloud Identity

- **G Suite:** Licenses for organization Google apps including Gmail, Google Drive, Google Calendar, and more
- **Cloud Identity:** Google identity only, no Google apps
- Both account types are created and managed from the same Google Admin Console (admin.google.com)
- Reminder: GCP does not care if an account is G Suite or Cloud Identity - they're just Google accounts

Cloud Identity Free vs. Premium

- Difference not relevant for working with IAM on GCP
- **Free account:** Provides core identity and management services
- **Premium account:** Additional enterprise security, application management, and device managed services
- Not a concern or focus for this course, we'll be using Free

Course Navigation

Course Intro
Section 1

Member Types

IAM Big Picture
Section 2

Creating Identities with Cloud Identity

Establishing Identity
Section 3

Hands-On - Cloud Identity

Granting Access to GCP Resources with Cloud IAM
Section 4

Synchronize Cloud Identity with Active Directory

Service Accounts
Section 5

Google Groups

Configure Multi-Factor Authentication

Next Sections

Back to Main

Why is this important?

Question: Where are you managing your users? What is your primary 'single source of truth'?

- **G Suite users:** Created and managed in Google Admin Console as the **primary** directory
- **Non-G Suite:** Typically **Active Directory** for Microsoft organizations

Problem:

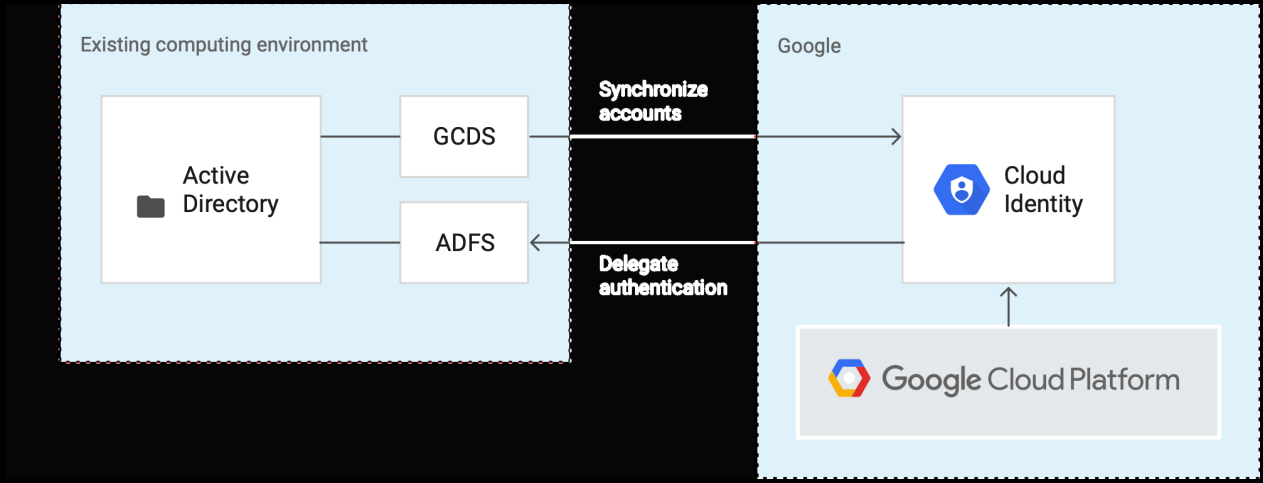
- Many organizations use Active Directory for the user management source but need to maintain Google accounts with Cloud Identity for GCP access
- Result: two separate directory systems to maintain
  - Cloud Identity for GCP/Google Accounts
  - Active Directory for other business functions
- Separately maintaining GCP accounts in Cloud Identity alongside Active Directory is cumbersome and error-prone
  - Need to manage the single source of identity management to manage both GCP and non-GCP identities
- **Solution:** Automatically synchronize Active Directory to Cloud Identity!
  - Create, update, and manage users (including passwords) in Active Directory
  - Changes automatically reflected in Cloud Identity
- Result: AD remains your primary 'single source of truth', and Cloud Identity synchronizes all changes to authenticate with Google applications

How It Works

- Cloud Identity maps (or federates) AD accounts to Cloud Identity accounts
  - Mapping = Federation
- Active Directory is the 'single source of truth'
- One-way sync from AD to Cloud Identity, not the other way around
- All user/password management done in AD:
  - Create/modify/delete all in AD
  - Cloud Identity automatically creates a Google account from AD accounts (or subset of accounts if desired)
- Provides single sign-on capability to GCP

How to synchronize - tools

- Google Cloud Directory Sync (GCDS)
  - Google-provided tool for synchronization process
  - Runs on AD server
- Active Directory Federation Services (ADFS)
  - Microsoft-provided tool



What are we doing?

- Briefly exploring admin console (admin.google.com)
- View, Create, and Manage G Suite accounts
- Enable Cloud Identity, and create Cloud Identity (non-G Suite) accounts
- Reminder: we are not assigning roles to GCP, just creating managed organization identities/members to attach GCP roles to later

Important considerations:

- Be aware that you cannot simulate working with Google Admin Console without registering a live domain name
- Access to manage GCP IAM roles is separate from Google Admin Console access
  - Can have access to one, and not the other

Course Navigation

Course Intro
Section 1

Member Types

IAM Big Picture
Section 2

Creating Identities with Cloud Identity

Establishing Identity
Section 3

Hands-On - Cloud Identity

Granting Access to GCP Resources with Cloud IAM
Section 4

Synchronize Cloud Identity with Active Directory

Service Accounts
Section 5

Google Groups

Configure Multi-Factor Authentication

Next Sections

Back to Main

Why is this important?

Question: Where are you managing your users? What is your primary 'single source of truth'?

- **G Suite users:** Created and managed in Google Admin Console as the **primary** directory
- **Non-G Suite:** Typically **Active Directory** for Microsoft organizations

Problem:

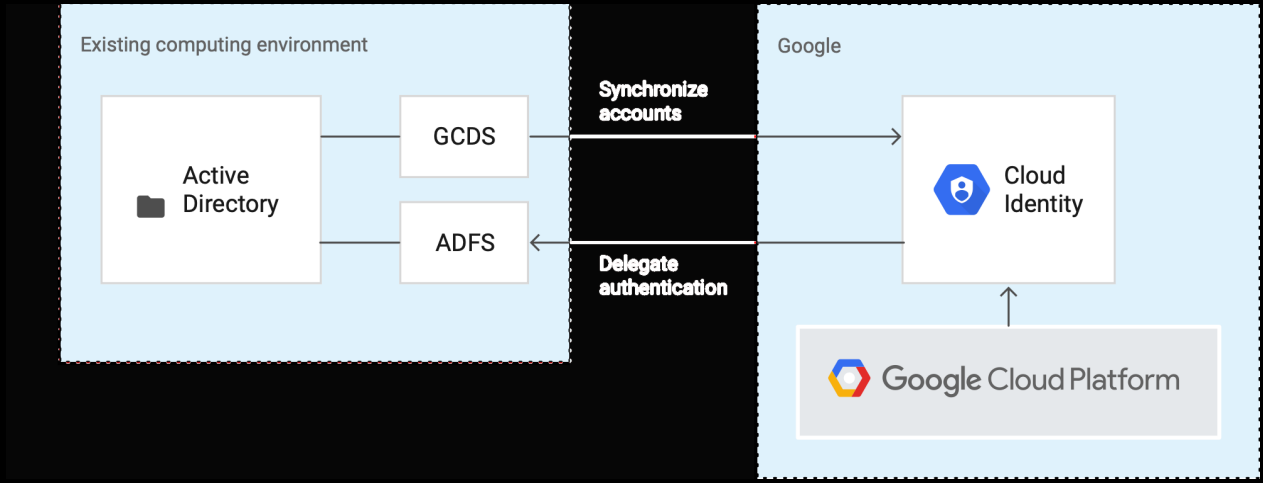
- Many organizations use Active Directory for the user management source but need to maintain Google accounts with Cloud Identity for GCP access
- Result: two separate directory systems to maintain
  - Cloud Identity for GCP/Google Accounts
  - Active Directory for other business functions
- Separately maintaining GCP accounts in Cloud Identity alongside Active Directory is cumbersome and error-prone
  - Need to manage the single source of identity management to manage both GCP and non-GCP identities
- **Solution:** Automatically synchronize Active Directory to Cloud Identity!
  - Create, update, and manage users (including passwords) in Active Directory
  - Changes automatically reflected in Cloud Identity
- Result: AD remains your primary 'single source of truth', and Cloud Identity synchronizes all changes to authenticate with Google applications

How It Works

- Cloud Identity maps (or federates) AD accounts to Cloud Identity accounts
  - Mapping = Federation
- Active Directory is the 'single source of truth'
- One-way sync from AD to Cloud Identity, not the other way around
- All user/password management done in AD:
  - Create/modify/delete all in AD
  - Cloud Identity automatically creates a Google account from AD accounts (or subset of accounts if desired)
- Provides single sign-on capability to GCP

How to synchronize - tools

- Google Cloud Directory Sync (GCDS)
  - Google-provided tool for synchronization process
  - Runs on AD server
- Active Directory Federation Services (ADFS)
  - Microsoft-provided tool



What are Google Groups?

- Group individual user accounts into groups for easy management
- Managed in either admin console (for G Suite/Cloud Identity admins), or legacy groups.google.com site for non-admins
- In the form of an email address (my-group@company-name.com)
- Non-GCP features:
  - Group email list
  - Email alias (info@company-name.com)
  - Message board capabilities

Why do we care about groups?

- Problem: Management for larger numbers of users with the same permissions
  - Adding/removing individual users who need the same permissions is cumbersome and prone to error
- Solution: Add identical-access users to groups
  - Add a single group to an IAM role = permissions applied to all users in the group
  - Can add/remove users from a group to have permissions apply in GCP automatically

What is multi-factor authentication?

- Protects accounts from unauthorized access
- Creates an additional security barrier between users and 'the bad guys'
- Requires both 'something you know' (password), and 'something you have' (phone/security key)

Importance when working with GCP

- GCP administrators have control over critical company resources
- Compromised accounts would be disastrous!
- Multi-factor authentication substantially hardens accounts
  - Bad guys need your phone/security key in addition to your password

How to enable - individually and corporately

- Individual enforcement must be enabled on Admin Console (on by default)
- Individual enablement - account.google.com
- Can also enforce across organization via Admin Console

Granting Access to GCP Resources with Cloud IAM

GCP Resource Hierarchy

Why is this Important?

Understanding IAM requires knowledge of how GCP resources are organized:

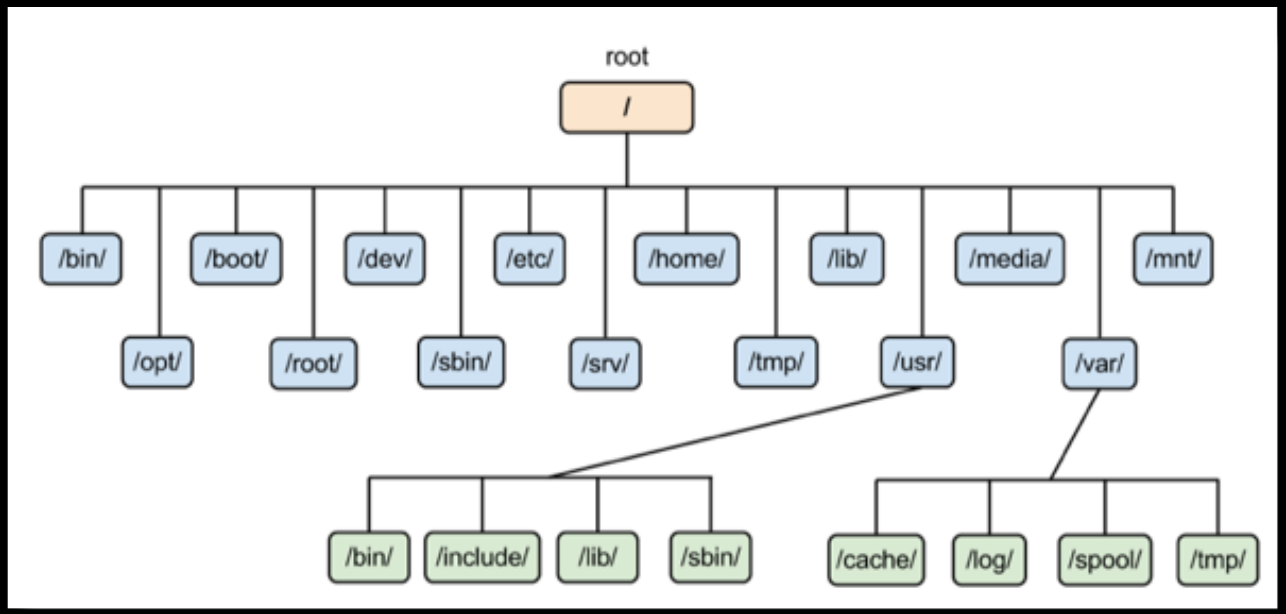
- Dictates inheritance of roles
- Establish organization context for the rest of this course

What is the GCP Resource Hierarchy?

- A "file system"-style arrangement of resource organization
- Each resource has exactly one parent

File System Arrangements and Relationships

- Single `root` layer
- Children resources have exactly one parent
- Permissions are inherited from the top down

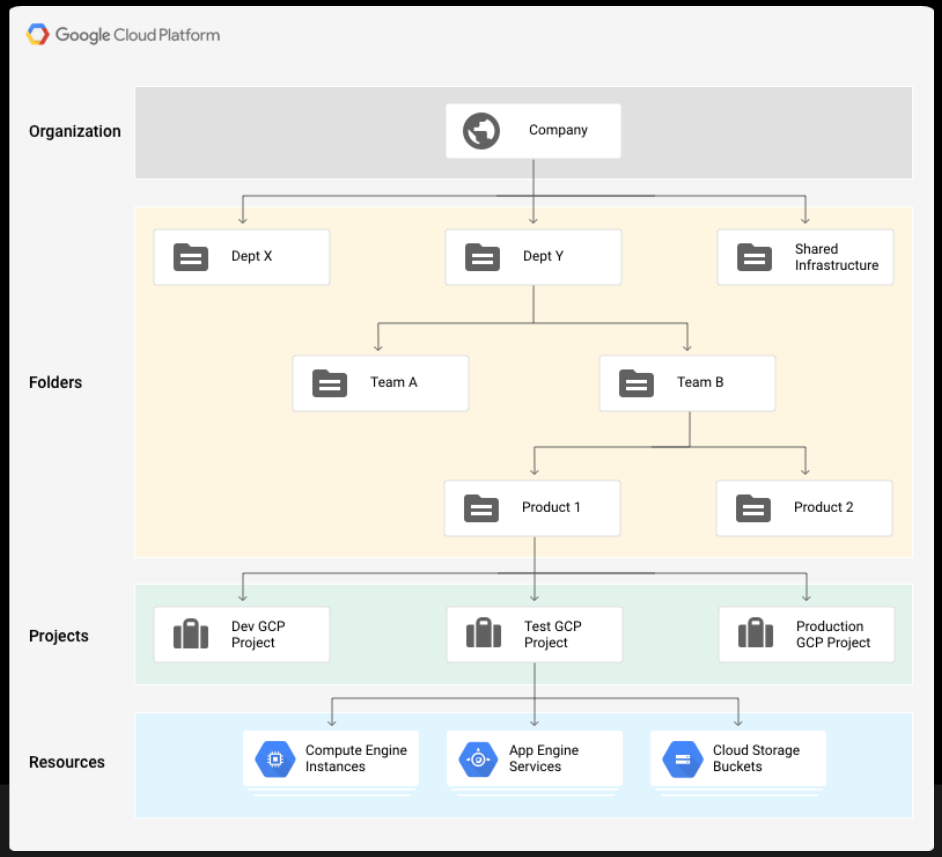


GCP Cloud Resource Hierarchy

GCP Layers and file system equivalents:

- Organization = `root`
- Folders = folders and sub-folders
- Projects = applications
  - GCP Resources inside of projects

Note: Personal Gmail/Google accounts (non-G Suite/Cloud Identity) will only have projects, no organization/folders.



Cloud Resource Hierarchy levels explained

Organization – Represents an entire company (@companyname.com):

- Root node
- Organization level roles applied to all layers – “global”
- Provides centralized administration

Folders - Optional grouping mechanism:

- IAM roles granted to folders are applied to all projects (or other folders) inside

Projects – The core organizational component of GCP:

- All billable resources created in projects
- Services in the same project have a default level of trust
- Billing, API’s, Roles

Resources – GCP services (Compute Engine, BigQuery, etc.):

- Created in projects
- Some resources have granular IAM roles (more on that later)

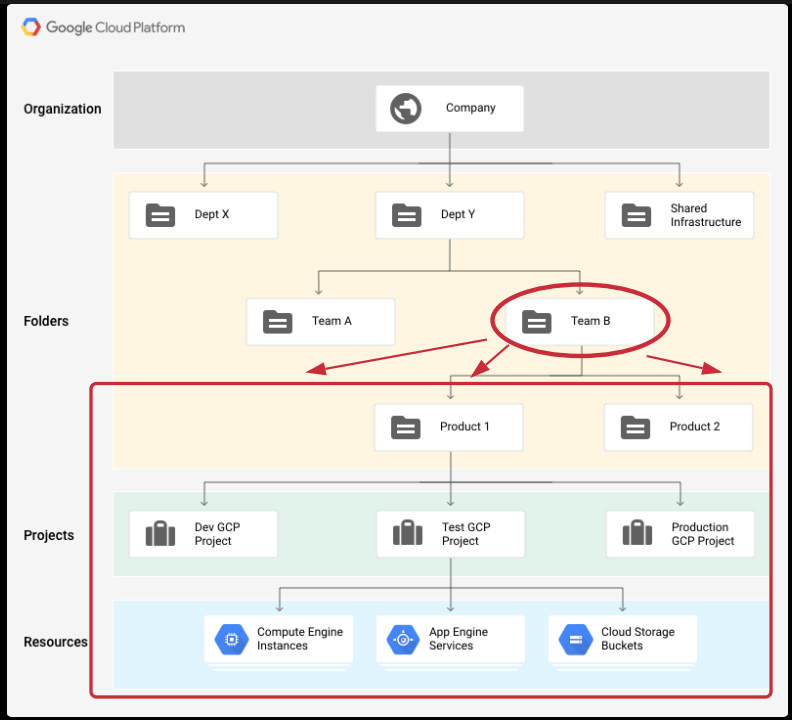
Why Does Resource Hierarchy Matter in Regards to IAM?

Inheritance – Permissions are inherited from the top, going downward:

- Lower hierarchy resources (projects, folders, granular resources) inherit roles from higher levels.

Example: Assign **bob@professionalwireless.net** the **Project Editor** role for folder **Team B**:

- Bob's Project Editor role applies to all new and existing folders, projects, and project resources inside of the **Team B folder**.



Granting Access to GCP Resources with Cloud IAM

Roles and Resources

Let's Get down to Business...

What we cover:

- What are roles and permissions?
- What is a resource?
- Bringing the 'who', 'can do what', and 'on which resource' together with an **IAM Policy**

What is a Role?

Role = 'can do what'

- **Roles** are assigned to **members**
- Collection of **permissions** that determine what a member can do, and with which resource
- A **permission** is a specific operation allowed on a resource
- You do not assign permissions to members, but permissions are bundled into roles, which are assigned
- Members can have multiple, overlapping roles assigned

Role - compute.instanceAdmin

Permissions

compute.instances.delete	compute.instances.get	compute.instances.list
compute.instances.setMachineType	compute.instances.start	compute.instances.stop
...		

Role Types - Primitive, Predefined, Custom

Primitive Role:

- Very broad, 'old-school' role - existed before the current implementation of IAM
  - Applies to all resources in a project
- **Owner:** Full control over all project resources, including assigning IAM roles and billing
- **Editor:** Same as Owner, but without IAM or Billing access
- **Viewer:** View rights only. Cannot edit anything in the project

Predefined Role:

- More granular and specific to a certain service
- **Predefined** (static) group of permissions
- Example: Compute Instance Admin = edit rights to GCE resources, but not to the rest of the project resources

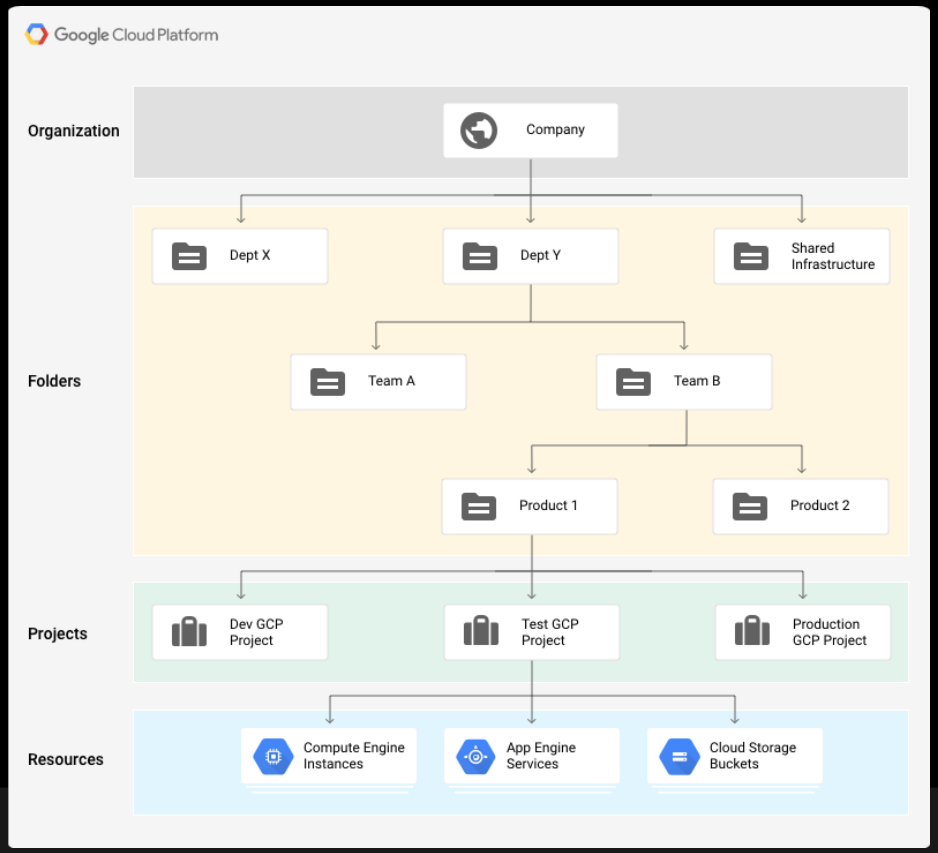
Custom Role:

- Mix and match individual permissions
- Even more granular — choose only the specific permissions you need.

What is a resource?

Simply put: Everything in GCP

- Projects
- Organizations
- Folders
- Compute Engine Instances
- Cloud Storage buckets
- BigQuery datasets
- And much more....



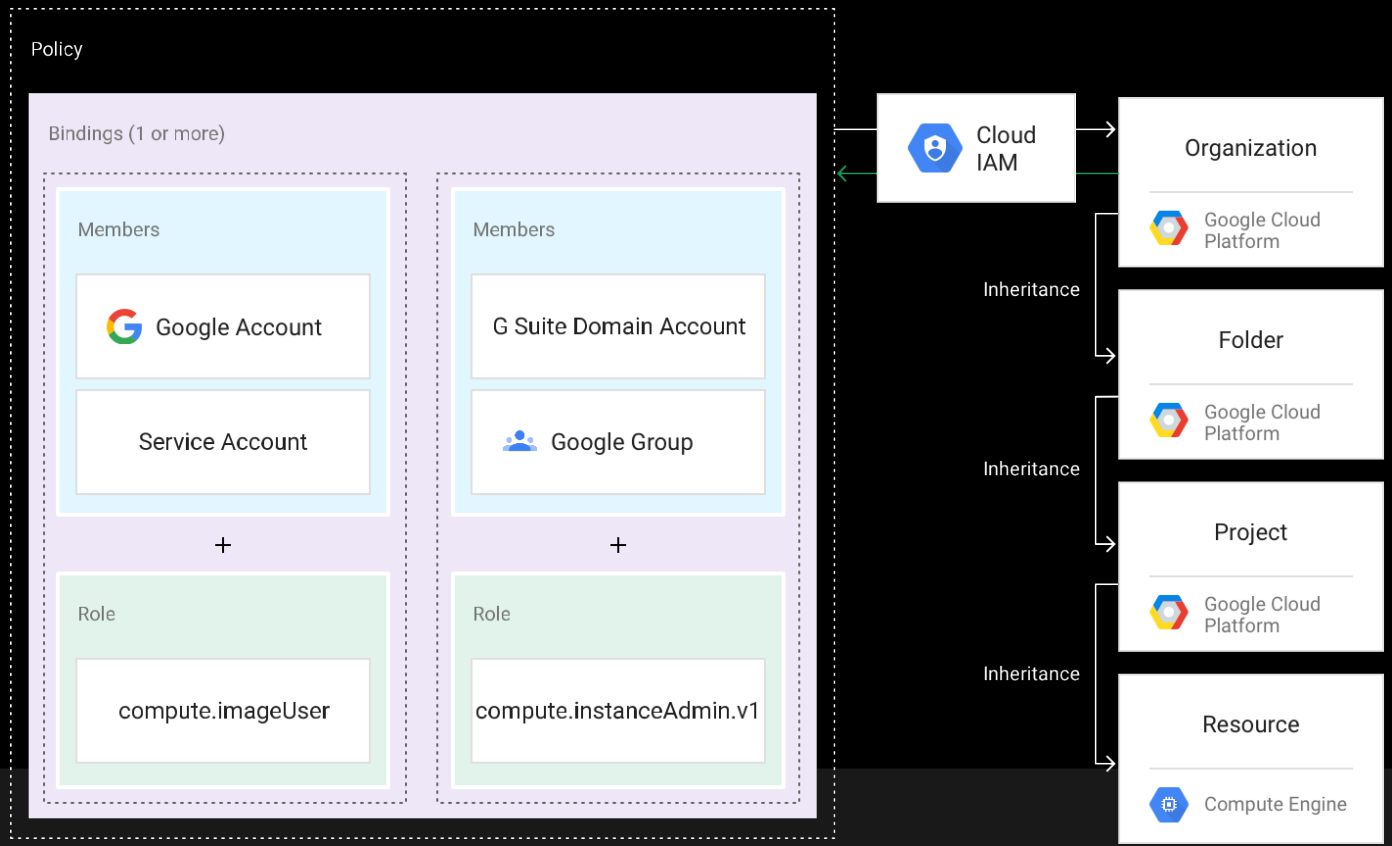
Cloud IAM Policy - Bringing it All Together

IAM Policy = Collection of statements (bindings) that define who has what type of access

- Attached to a resource
  - Organization, Folder, Project, Cloud Storage bucket
  - Higher-level resource policies inherited by lower levels
- Binding - pairs (or 'binds') a member to a role
  - Example: Grant bob@professionalwireless.net the Project Viewer and Compute Instance Admin roles

IAM Policy

IAM Policy


Policy
Bindings (1 or more)
Members
Google Account
Service Account
+
Role
compute.imageUser
Members
G Suite Domain Account
Google Group
+
Role
compute.instanceAdmin.v1
Cloud IAM
Organization
Google Cloud Platform
Inheritance
Folder
Google Cloud Platform
Inheritance
Project
Google Cloud Platform
Inheritance
Resource
Compute Engine

What are We Covering?

We are looking at how to:

- View Organization node roles
- Establish initial GCP IAM context
- Create a project and view IAM interactions

Organizations Node - Key Considerations:

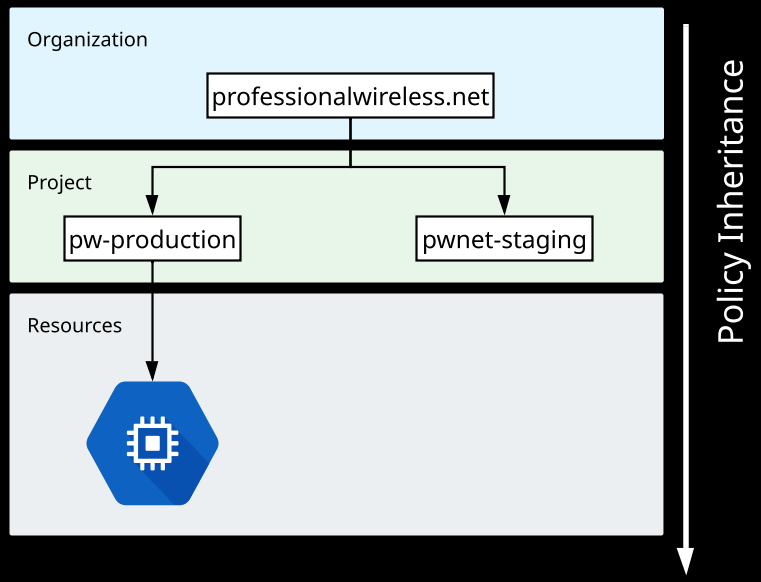
- Organization Administrator role: FULL access to assign any role to any resource in a GCP organization:
  - You should **use multi-factor authentication** for Org Admin roles
- G Suite/Cloud Identity Super Administrators (not a GCP role) have the Org Admin role by default

What are We Covering?

- Managing IAM roles in both Organizations and Projects:
  - Viewing inheritance of policies between both
- Only web console, for now, we will use command line later

Lesson tasks:

- Add member to organization node role
- Creating projects
  - Assign billing account...or can we?
- Add another member to project, but not to organization
- Enable viewing other projects



What are We Covering?

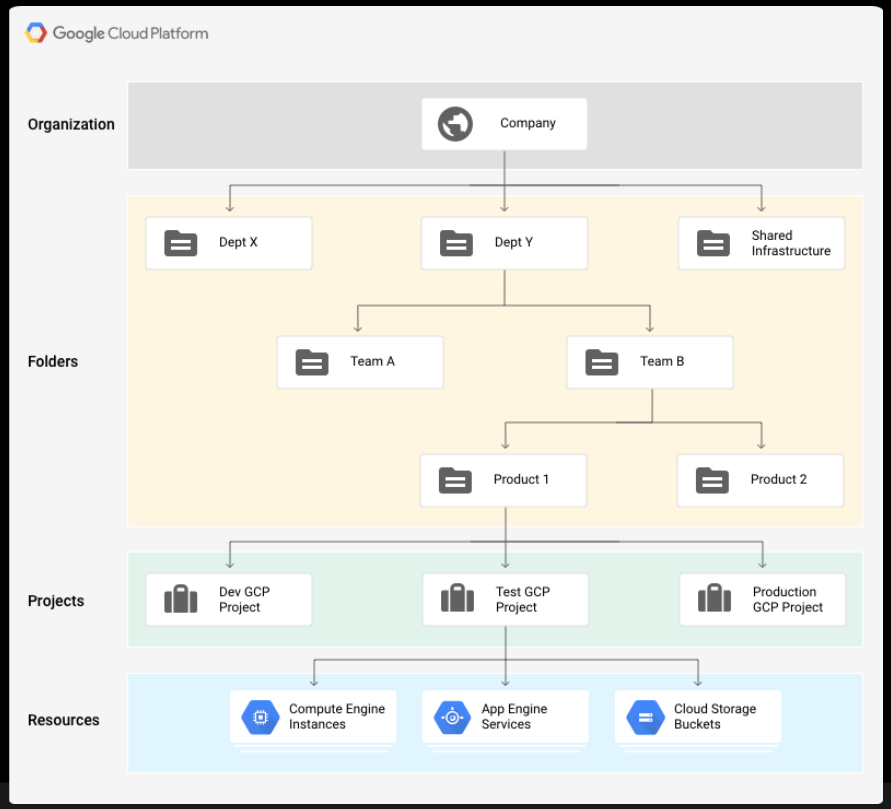
Creating and working with folders.

Roles:

- **Folder Admin** - Create, edit, move, delete folders + set IAM roles on folders
- **Folder Creator** - Create folders
- **Folder Editor** - edit and delete folders
- **Folder IAM Admin** - set IAM roles on folders
- **Folder Mover** - move folders, move projects in/out of folders
- **Folder Viewer** - View folder info, and nothing more

Key considerations:

- Must have Folder specific roles to work with folders - even if you're an Org Admin
  - Moving projects into folder also requires Editor/Owner role
- Removing a project from a folder will remove folder-applied roles from that project:
  - Applies to sub folders as well



What are We Covering?

- Viewing relationship of permissions and roles in Roles menu
- Creating Custom Roles
  - Custom roles can be created at different layers of the resource hierarchy

Some GCP Services Have an Additional IAM 'Layer'

- Most service IAM roles apply at the project level or higher
- Some services can assign roles to only a certain 'container' instead of all instances in a project
- Examples:
  - Cloud Storage Buckets
    - We will cover Cloud Storage in more detail later
  - BigQuery datasets
  - Cloud Pub/Sub topics

Granular access to Compute Engine instances

- Problem: Members need SSH access to either
  - Some GCE instances in a project
  - All instances in project, but not have ability to create/delete instances
- Solution: Grant OS Login role
- How it works:
  - Grant **OS Login** and **Service Account User** roles to members
  - Add metadata per-instance or project-wide to enable OS Login for affected instances
    - `enable-oslogin:TRUE`
  - Members can connect via SSH using `gcloud compute ssh` command (or use web console) to administer instance, but cannot create/delete instances.
- OS Login Roles
  - **OS Admin Login** - SSH with admin access
  - **OS Login** - SSH without admin access

Granting Access to GCP Resources with Cloud IAM

Managing IAM Policies with Command Line

What are we covering?

We'll be covering managing IAM policies using a command line (in this case, Cloud Shell).

Two methods of command line interaction:

- Edit and replace the entire IAM policy for the entire project/organization/etc.
  - Not adding/removing single binding, but completely overwriting project/org/folder IAM policy with a new one
- Add/remove individual bindings to an existing policy
  - Example: Add a Project Editor role for a single member

Command Line Methods

- **Fully replace existing policy**
  - Getting/downloading the current policy is optional, can always set a fresh policy without downloading the old policy first
  - **Get** (download) a copy of the policy:
    - `gcloud projects get-iam-policy [PROJECT_ID] --format [FORMAT] > [FILE-PATH]`
    - JSON or YAML format
  - Add/edit/remove members in downloaded policy file
  - **Set** new policy by applying an edited one:
    - `gcloud projects set-iam-policy [PROJECT_ID] [FILEPATH]`
    - Provide the file path to the file you just edited
- **Directly add/remove a single IAM role:**
  - Does not affect other assigned roles in a project
  - `gcloud projects add-iam-policy-binding [PROJECT_ID] --member user:[EMAIL] --role [ROLE_ID]`
  - `gcloud projects remove-iam-policy-binding [PROJECT_ID] --member user:[EMAIL] --role [ROLE_ID]`

Editing policies on organizations/folders:

- Similar commands, but for organizations/folders instead of projects
  - `gcloud organizations get-iam-policy [ORGANIZATION_ID] ...`
  - `gcloud resource-manager folders get-iam-policy [FOLDER_ID] ...`

Why This is Important

An example of troubleshooting:

- Problem: Sometimes a member is unable to perform the role that you think their IAM role should let them do.
  - OR: a member has access to a resource they shouldn't have, and you need to track down why
  - Overlapping, inherited roles from different resource hierarchy layers add to complexity
- Solution: Use Policy Troubleshooter to troubleshoot.
- Basic concept: Enter the member account, the API they're using, and the **permission** (not role) they need for the resource, and get a report of what IAM roles allow/don't allow what they need to do.

Considerations:

- Need to know the full API name of a resource: https://cloud.google.com/iam/docs/full-resource-names
- Need to know the exact permissions a member requires: https://cloud.google.com/iam/docs/permissions-reference

Granting Access to GCP Resources with Cloud IAM

Principle of Least Privilege

This information is important to remember:

- Give people just enough access to do what they need, and not any more:
  - Use predefined roles over primitive roles
  - Even better: Use custom roles if you need to really fine-tune a scope of access
- Grant roles at smallest scope necessary:
  - E.g., Compute Admin instead of Editor
  - Even better: Compute Instance Admin instead of Compute Admin
- If someone needs full project edit rights, use Editor over Owner:
  - Owner can change IAM policy, Billing

Other Best Practices

- When possible, use groups
  - Easier management
- Have more than one Organization Administrator
  - If one org admin is suddenly unavailable, someone else can fill in as back up
- Remember, projects separate member access. Separate teams needing different access = separate projects
  - Example: Separate production and development environments into separate projects

What is a Service Account?

- Short definition: application/service/VM authentication with GCP services
- Special type of Member (the 'who')
  - Not attached to a user ('people' accounts)
  - Resources don't need end-user authentication
    - Still needs authentication, but different method
- Identity represented by email address
  - Example: 126692436861-compute@developer.gserviceaccount.com
- By default, GCE instances use service accounts for GCP access

Why do we care about Service Accounts?

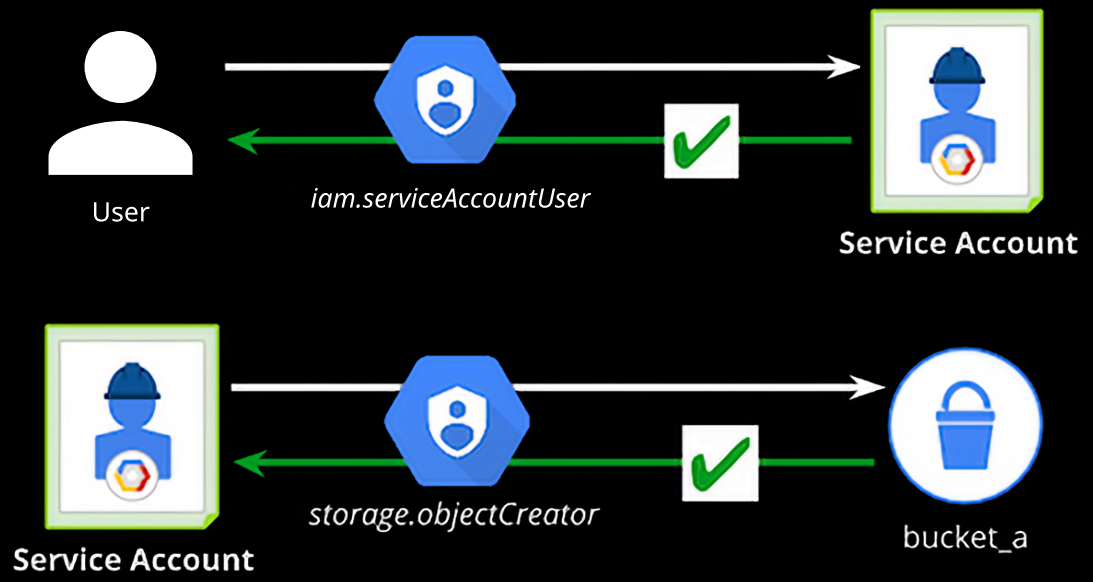
- Not dependent on a user:
  - If applications authenticate with user account, and user leaves the organization/account disabled, application stops working
- Better isolation of services:
  - Each service can use its own service account, limiting scope of access

Types of Service Accounts

- **User-managed** (not necessarily user-created)
  - Created by both users and Google (automatically)
  - User manages account's access via IAM roles/scopes
  - Google-created - API-related accounts
    - Default Compute Engine service account
      - **PROJECT_NUMBER-compute@developer.gserviceaccount.com**
    - Default App Engine service account
      - **PROJECT_ID@appspot.gserviceaccount.com**
  - User-created
    - **SERVICE_ACCOUNT_NAME@PROJECT_ID.iam.gserviceaccount.com**
- **Google-managed**
  - Created and managed by Google - IAM roles automatically controlled by Google
  - Generally invisible to the end-user
    - In other words, leave it alone.
  - Example: Cloud Services account - PROJECT_NUMBER@cloudservices.gserviceaccount.com
    - Created when project is created
    - Runs internal Google processes on your behalf

Both a Member **AND** a Resource

- Service accounts are both a member (who) and a resource
- Service accounts are granted permissions to a resource
- Users (person) are granted the role serviceAccountUser to a service account
  - In other words, permissions to impersonate, or act as the service account
  - Example: OS Login role requires Service Account User role



Putting it all together

- Service accounts grant application/VMs access
- Users are granted access to act as a service account
  - Service accounts are granted access to resources.
- Service accounts use keys for access
- Service accounts granted access based on both scopes and IAM

	End users (people)	Service Accounts
Authentication	username/password	RSA public/private key
Creation/management	Google Admin Console	GCP (projects, organizations)
Access management	IAM roles	Also IAM roles

What are we covering?

- Create a new service account - Google-managed key
- Edit, disable, and enable service account
- Create another service account with the command line
- Assign IAM roles to the created service account
  - Also using web console and command line
- Delete service accounts

Command line

- Create service account
  - `gcloud iam service-accounts create (account_name) --description "(description)" --display-name "(display name)"`
- Show service accounts in a project
  - `gcloud iam service-accounts list`
- Rename service account display name/description (cannot change address)
  - `gcloud iam service-accounts update (SA-NAME)@(PROJECT-ID).iam.gserviceaccount.com --description "(UPDATED-SA-DESCRIPTION)" --display-name "(UPDATED-DISPLAY-NAME)"`
- Disable/enable service account
  - `gcloud iam service-accounts (disable/enable) (SA-NAME)@(PROJECT-ID).iam.gserviceaccount.com`
- Delete service account
  - `gcloud iam service-accounts delete (SA-NAME)@(PROJECT-ID).iam.gserviceaccount.com`
- Add IAM binding
  - `gcloud projects add-iam-policy-binding (PROJECT-ID) --member serviceAccount:(SA-NAME)@(PROJECT-ID).iam.gserviceaccount.com --role (ROLE)`

Service Account and Compute Engine

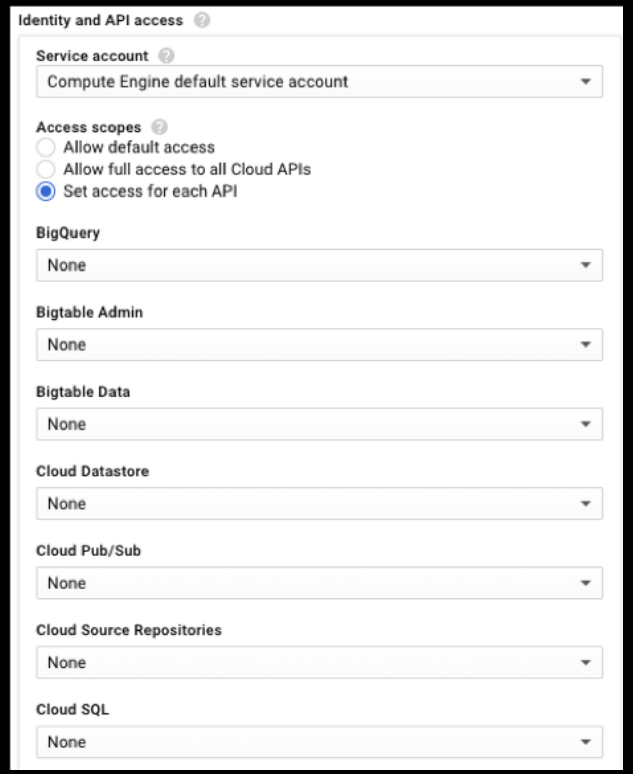
- When user is logged into GCE instance, they are acting as a Compute Engine Service Account
- Interactions with GCP on the instance are under the Service Account's permissions
- Can only assign one service account per instance, can change account with instance stopped

Default Compute Engine Service Account

- Automatically created when Compute Engine API enabled
- Automatically assigned to GCE instances (unless changed to another service account)
- Automatically assigned Editor role...
  - ...but uses scopes for instances by default

Scopes

- Applied to default GCE service account
- Legacy permissions method before current implementation of IAM
- Grant access to GCP API's
- If using another service account on instance, only IAM roles are used



Service Account Keys

- Service accounts need to authenticate to GCP resources
- Service accounts authenticate with **service account keys**
  - Think of it as the service account's 'password'
- Format: public/private RSA key pair

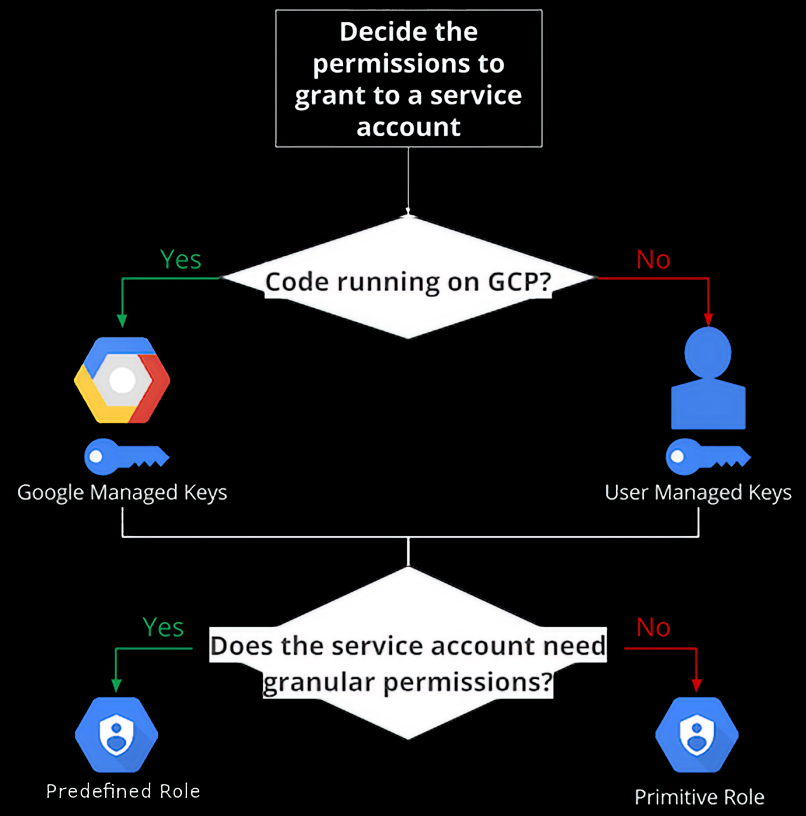
Key Types

Google Managed

- Google stores and manages the key for you
- No interaction needed by end user
- Recommended for code/applications running on GCP

User Managed

- User-created service accounts can use user-managed (custom) keys, which you store/manage
- Google maintains public copy for verification, but the public/private key pair is yours to manage
- If you lose your private copy of the key, Google cannot retrieve it!
- Recommended for code/applications outside of GCP



What are we covering?

- Create, view, and delete keys using both web console and command line
- Authenticate a local backup application to GCP with service account key
- Be aware: Google still has their own managed key alongside user-managed keys

Rotating keys

- You don't exactly "rotate" keys.
- Rather, you create a new key, point application to the new key location, and then delete the old one.

Command line reference

- **Create key**
  - `gcloud iam service-accounts keys create /(PATH)/(FILENAME).json --iam-account (SERVICE_ACCOUNT)`
- **List keys per service account**
  - `gcloud iam service-accounts keys list --iam-account (SERVICE_ACCOUNT)`
- **Delete service account key (requires key ID)**
  - `gcloud iam service-accounts keys delete (KEY-ID) --iam-account (SERVICE_ACCOUNT)`

Key management

- Rotate service account keys (user managed)
- IMPORTANT: Don't delete keys in use by running resources
  - Assign account to new key before deleting old one
- Don't check in service account keys to source code or leave in downloads directory!
- Name service keys (file name) to reflect use and permissions

Limit access — principle of least privilege

- For multi-service applications, treat each app component as a separate trust boundary
  - Create separate service account for each service
- Restrict service account end-user access
  - Restrict who can create and manage service accounts (Service Account Admin)

Why are we talking about Cloud Storage access security?

Security in storage buckets is REALLY important!

- Ignoring Cloud Storage security means becoming famous for the **wrong** reasons
- These incidents are 100% preventable
  - Result of misconfigured permissions

Security is in YOUR control!

- Access is as "locked down" or "wide open" as you choose

Corporate tech giant leaves secret data exposed to public internet

There's a Hole in 1,951 Amazon S3 Buckets

Poorly configured cloud buckets strike again

Customer Data Left Publicly Exposed on Cloud Storage Server

Cloud Storage access management methods

Two methods: IAM roles and Access Control Lists (ACLs)

- Both methods are independent and not reliant on each other
  - Either/or method of access — either IAM or ACL will grant access to a resource

IAM roles on Cloud Storage

- Same IAM principles as the rest of GCP
- Hierarchical in nature
- Granted at **project** or **bucket** level, not per **object**
- Bucket-level **legacy roles** are equivalent to ACLs

Access Control Lists (ACLs) — legacy method

- Completely independent from IAM
- Granted at **bucket** or **object** level, but not **project**
- Object-level ACLs inherit bucket-level ACLs

Which access method should we choose?

Google's recommendation: Use IAM over ACL when possible

- IAM provides:
  - Enterprise-grade control across all of GCP
  - Audit trail via logs
- ACL provides:
  - Fine-grained, object-level access control

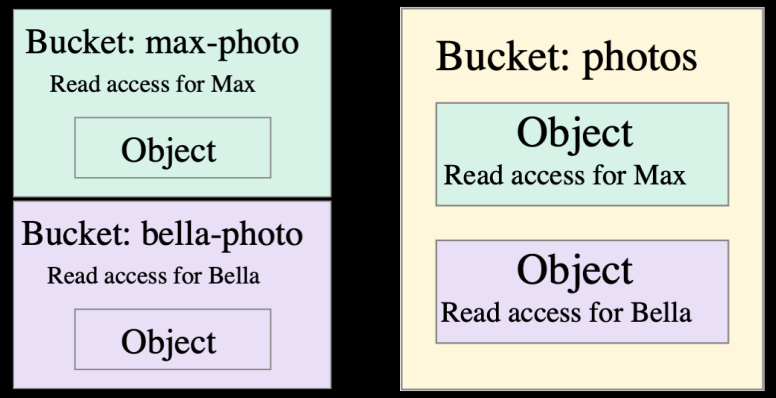
Another perspective: uniform vs. fine-grained access

Uniform access

- Bucket-level access with **only** IAM roles for sensitive data
- Lower chance of accidental data exposure

Fine-grained access

- Mix of project/bucket IAM roles and bucket/object ACLs
- Higher chance of accidental data exposure
- Not recommended for sensitive data — especially for different people



Uniform

Fine-grained

Signed URLs

URL-based timed access

- Useful for temporarily giving access without needing to sign in with Google account
- Gives user read, write, or delete access for limited time
- Anyone with the URL can access within the time period
  - URL is the "key" to access the data

What are we covering?

Apply IAM roles to buckets

- Assign IAM roles to buckets
  - `gsutil iam ch user:(user_email):(role1,role2) gs://(BUCKET)`
- Remove IAM role from bucket
  - `gsutil iam ch -d user:(user_email):(role1,role2) gs://(BUCKET)`
- Remove all roles from bucket for given user
  - `gsutil iam ch -d user:(user_email) gs://(BUCKET)`
- Give public read access to bucket
  - `gsutil iam ch allUsers:objectViewer gs://(BUCKET)`

Apply ACLs to buckets and objects

- Assign ACL roles to buckets and objects
  - `gsutil acl ch -u (user_email):(O/R/W) gs://(BUCKET)/(OBJECT)`
- Delete ACLs for user
  - `gsutil acl ch -d (user_email) gs://(BUCKET)/(OBJECT)`
- Set default ACL for all new objects (in this case, public read access)
  - `gsutil defacl ch -u AllUsers:R gs://(BUCKET)`

Create signed URL

- Create service account
  - `gcloud iam service-accounts create signed-url-agent --display-name "Signed URL Service Account Agent"`
- Create user-managed key for service account
  - `gcloud iam service-accounts keys create keyfile.json \ --iam-account (SERVICE_ACCOUNT)`
- Grant Storage Object Viewer IAM role to service account

```
gcloud projects add-iam-policy-binding (PROJECT_ID) \
  --member serviceAccount:(SERVICE_ACCOUNT) \
  --role roles/storage.objectViewer
```

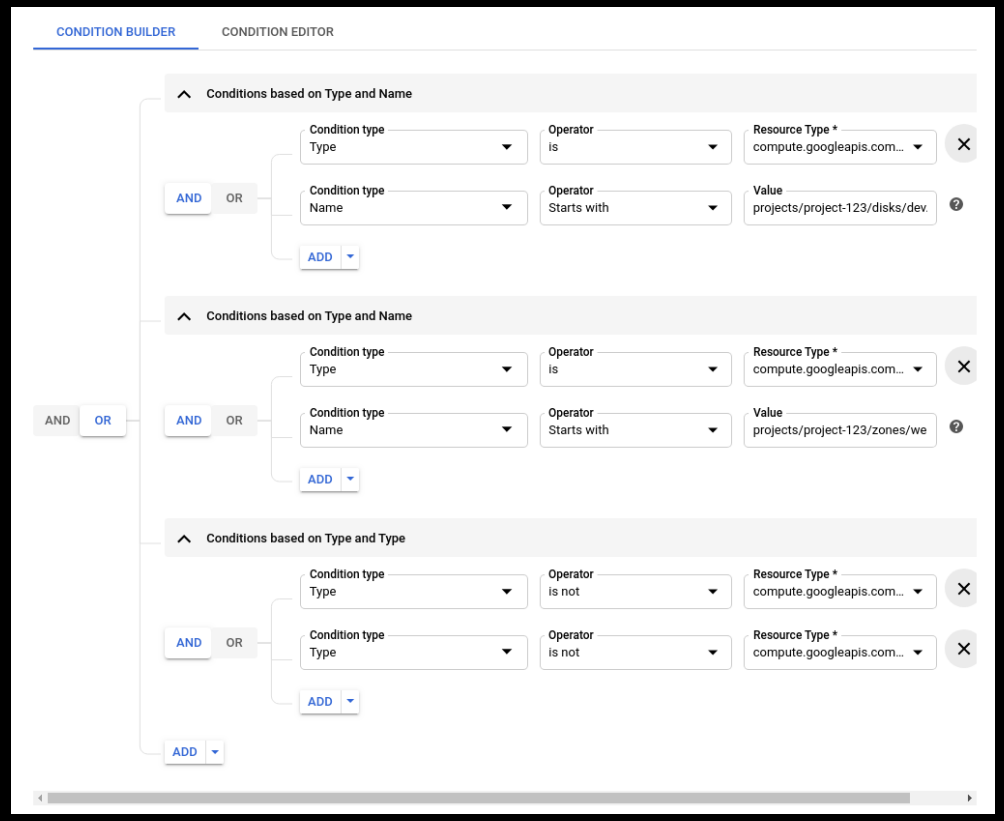
- Generate signed URL on object to grant timed access
  - `gsutil signurl -d 1m keyfile.json gs://(BUCKET)/(OBJECT)`
  - Might have to install pyOpenSSL library
    - `sudo pip install pyopenssl`

What are IAM Conditions?

Condition-based IAM access:
- Access granted only if additional conditions are met
- Time and Resource based conditions

Time-Based Conditions:
- Temporary and time-dependent access
- Access removed after (x) days, or until (x) date
- Limit access to certain days/hours of the week (e.g., during office hours)

Resource-Based Conditions:
- Limit access to certain resources based on name/type/service
- Provides even more granular access
- Example: Only allow access to instances that start with name 'devteam-', but not others in the same project



Who did what, where, and when?

All IAM-related actions generate an Admin Activity log:

- Admin Activity logs are always on and cannot be deleted
- Digital 'paper trail'

Viewing actions:

- Activity field in the web console dashboard
- Logs in Stackdriver Logging, using basic and advanced filters by service

Security 'Mission Control'

Organization-level security overview

- Much more than just IAM monitoring
- IAM-related alerts:
  - Public Cloud Storage buckets
  - Non-organization IAM members

Congratulations on completing the course. What next?

Apply what you have learned

- Practice with labs and playground projects
- If applicable, review access management for your live GCP environment

Continue your learning journey

- Explore other GCP and other tech domain deep dives
- As always, hands on practice is essential

Get certified!

- IAM is a core component in every GCP certification