

# Cloud Identity and Access Management (IAM)

Architecting with GCP Fundamentals:  
Infrastructure

CLOUD IAM, CLOUD RESOURCE MANAGER



Last modified 2017-11-27

© 2017 Google Inc. All rights reserved. Google and the Google logo are trademarks of Google Inc. All other company and product names may be trademarks of the respective companies with which they are associated.

## Agenda

- **Cloud Identity and Access Management (IAM)**
- Organization
- Roles
- Members
- Service accounts
- Cloud IAM best practices
- Quiz
- Lab

## Cloud Identity and Access Management



Who



can do what



on which resource

**Project Owners** invite members to projects and grant roles. Roles consist of a collection of permissions for one or more resources.

## Cloud IAM objects

- Organization
- Folders
- Projects
- Members
- Roles
- Resources
- Products
- G Suite Super Admins

## Cloud IAM resource hierarchy

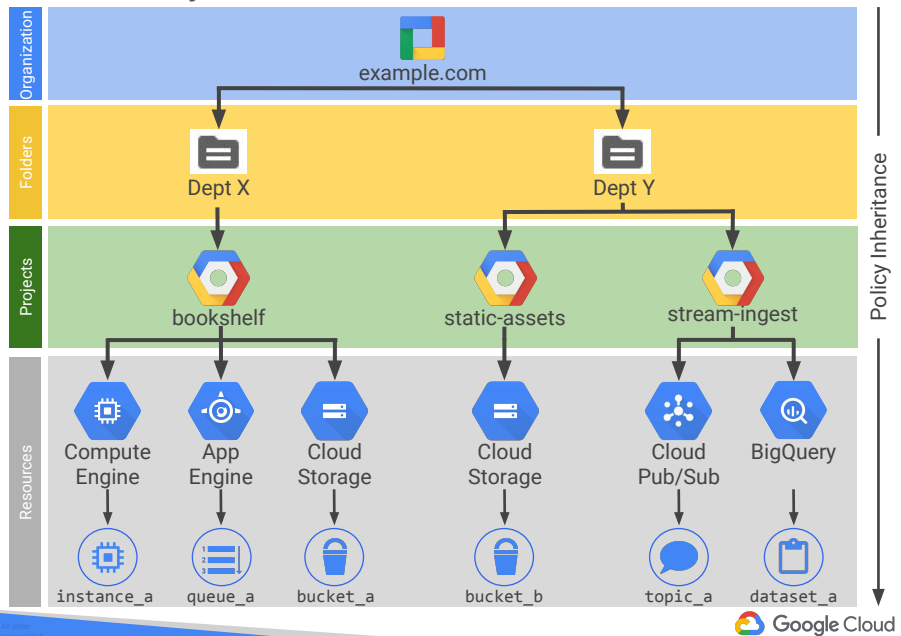
A policy is set on a resource, and each policy contains a set of:

- Roles
- Role members

Resources inherit policies from parent:

- Resource policies are a union of parent and resource.

If parent policy is less restrictive, it overrides a more restrictive resource policy.



Google Cloud Platform resources are organized hierarchically, where the Organization node is the root node in the hierarchy, the projects are the children of the organization, and the other resources are the children of projects. Each resource has exactly one parent.

Cloud IAM allows you to set policies at the following levels of the resource hierarchy:

- **Organization level:** The organization resource represents your company. Cloud IAM roles granted at this level are inherited by all resources under the organization.
- **Project level:** Projects represent a trust boundary within your company. Services within the same project have a default level of trust. For example, App Engine instances can access Cloud storage buckets within the same project. Cloud IAM roles granted at the project level are inherited by resources within that project. When setting policies at the project level, be sure to use audit logs to track project-level permission changes.
- **Resource level:** In addition to the existing Cloud Storage and BigQuery ACL systems, additional resources such as Genomics Datasets and Pub/Sub topics support resource-level roles so that you can grant certain users permission to a single resource.

Resources inherit the policies of the parent resource. If you set a policy at the organization level, it is automatically inherited by all its child projects, and if you set a policy at the project level, it is inherited by all its child resources. The effective policy for a resource is the union of the policy set at that resource and the policy inherited

from its parent. This policy inheritance is transitive; in other words, resources inherit policies from the project, which inherits policies from the organization. Therefore, the organization-level policies also apply at the resource level.

The Cloud IAM policy hierarchy follows the same path as the GCP resource hierarchy. If you change the resource hierarchy, the policy hierarchy changes also. For example, moving a project into an organization will update the project's Cloud IAM policy to inherit from the organization's Cloud IAM policy.

Child policies cannot restrict access granted at the parent. For example, if you grant the Editor role to a user for a project, and grant the Viewer role to the same user for a child resource, the user still has the Editor role for the child resource. When using Cloud IAM, a best practice is to follow the principle of least privilege. The principle applies to identities, roles, and resources. Always select the smallest scope that's necessary to reduce your exposure to risk. You wouldn't want to grant everybody the Owner role on your entire organization: intentional hacks or accidental mistakes could bring down your apps. You want to be specific and deliberate. Assign to a specific security admin group the security admin role to manage SSL and firewall rules on specific projects.

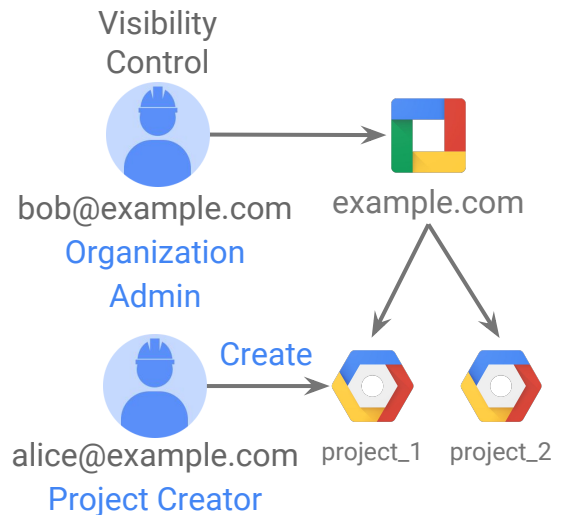
For more information, see: [Identity and Access Management Overview](#)

## Agenda

- Cloud Identity and Access Management (IAM)
- **Organization**
- Roles
- Members
- Service accounts
- Cloud IAM best practices
- Quiz
- Lab

## Organization node

- An organization node is a root node for Google Cloud resources.
- Organization roles:
  - **Organization Admin:** Control over all cloud resources; useful for auditing
  - **Project Creator:** Controls project creation; control over who can create projects



For more information, see:

<https://cloud.google.com/resource-manager/docs/quickstart-organizations>

A large number of projects can become unwieldy to manage at scale. This is why Cloud IAM includes the concept of an *organization node*. The organization node sits above projects and is your company's root node for Google Cloud resources. If you have a Google for Work account, when you enable the organization node, any project created by users in your domain will automatically belong to your organization node; no more shadow projects and no more rogue admins.

The Organization Admin role gives your admin visibility and control over all of your company's resources on Google Cloud Platform. Using the Project Creator role, you can restrict who can create projects, regardless of whether they have policies on individual projects. The project roles can also be applied at the organization level and can be inherited by all the projects in your company. For example, you can assign your networking team the Network Admin role at the organization level so they have permissions to manage all the networks in all the projects in your company.



## Organization

- Organization is created by Google Sales
- Organization Owners are established at creation
  - **G Suite** Super Admins are *the only Organization Owners*
- **Organization Owner**
  - Assigns the Organization Administrator role from the **G Suite Admin Console**—(**Admin** is a separate product)
    - Organization Administrators manage GCP from the Cloud Console
- Always have more than one organization owner, for security purposes.

Personal/trial accounts have no "organization," and organization features are hidden in the UI.

<https://cloud.google.com/resource-manager/docs/creating-managing-organization>  
<https://cloud.google.com/resource-manager/docs/quickstart>

The account with Organization Owner role is empowered to modify all projects within the organization.

Changes to the organization itself still occur only through Google Sales.

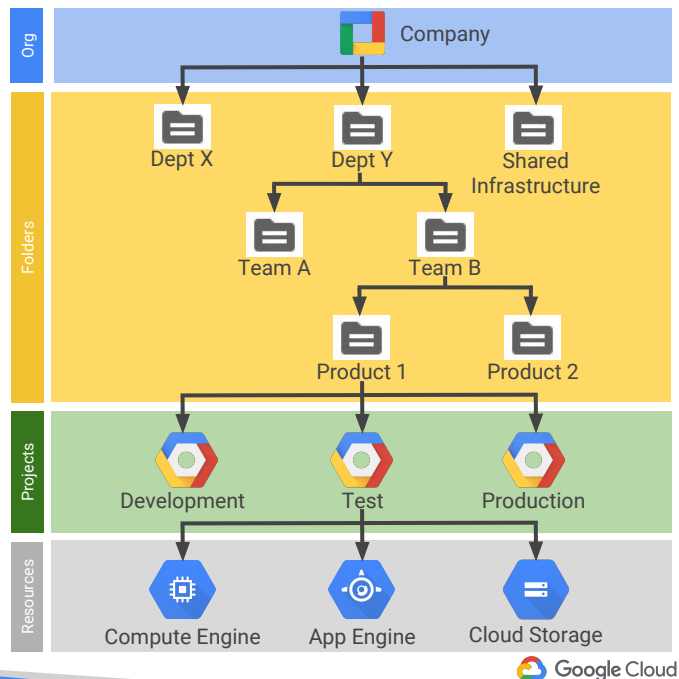
<https://cloud.google.com/resource-manager/docs/overview#organization>

## Folders

Additional grouping mechanism and isolation boundaries between projects:

- Different legal entities
- Departments
- Teams

Folders allow delegation of administration rights.



Folder resources provide an additional grouping mechanism and isolation boundaries between projects. They can be seen as sub-organizations within the organization. Folders can be used to model different legal entities, departments, and teams within a company. For example, a first level of folders could be used to represent the main departments in your organization. Because folders can contain projects and other folders, each folder could then include other sub-folders, to represent different teams. Each team folder could contain additional sub-folders to represent different applications.

Folders allow delegation of administration rights, so for example, each head of a department can be granted full ownership of all GCP resources that belong to their departments. Similarly, access to resources can be limited by folder, so users in one department can only access and create Cloud resources within that folder.

## Resource manager roles

### Organization

- **Admin:** Full control over all resources
- **Viewer:** View access to all resources

### Folder

- **Admin:** Full control over folders
- **Creator:** Browse hierarchy and create folders
- **Viewer:** View folders and projects below a resource

### Project

- **Creator:** Create new projects (automatic owner) and migrate new projects into organization
- **Deleter:** Delete projects



**Organization-level Administrators** can grant a role to a member that spans all projects.

Example use: Granting a security auditor read access to all logs (Log Viewer role) for all projects.

This would be much more efficient than granting the security auditor (or security auditor group) access for each project individually.

First Organization Administrator role - assigned by Google - usually not the G Suite Super Admin

Change Cloud IAM policy defaults for the organization

Project-level settings can override inherited defaults

Manage billing accounts and payment methods

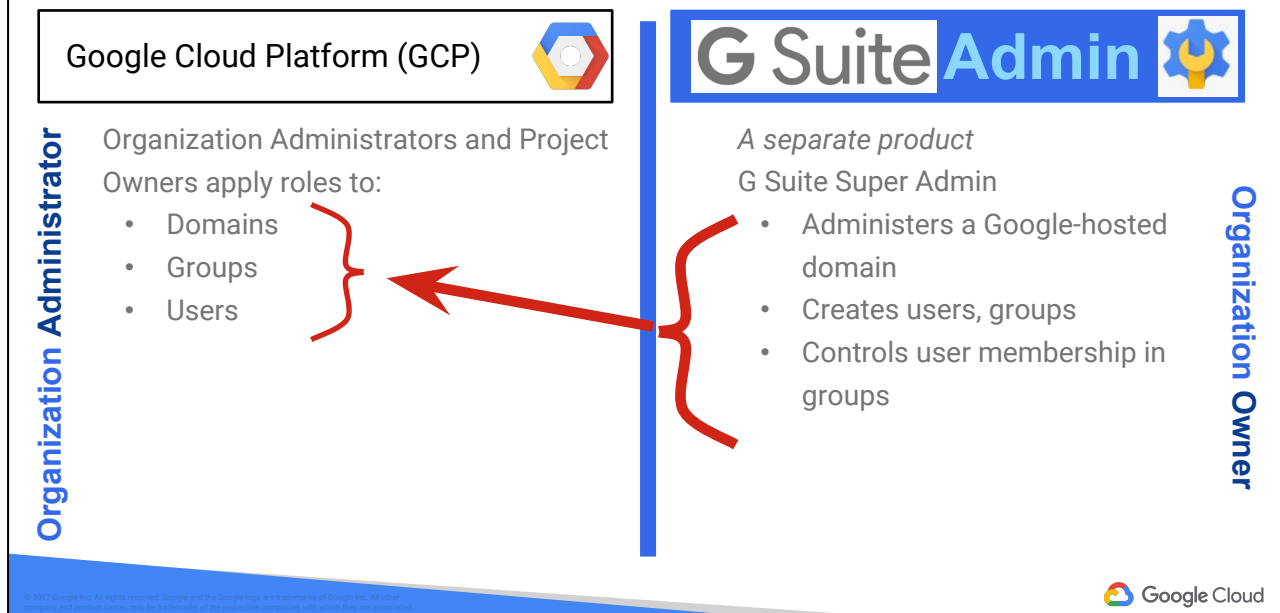
Full list of resource manager roles here:

[https://cloud.google.com/iam/docs/understanding-roles?hl=en\\_US&\\_ga=2.73506386.-1171753218.1503062023&\\_gac=1.254720634.1510004312.Cj0KCQiArYDQBRDoARIsAMR8s\\_TiY1w4BPcHZItQ3Mkv2jSj7fw17PcWR1DwvZZMn9nvkfE8PO6YsEaAobUEALw\\_wcB#crm\\_name\\_short\\_roles](https://cloud.google.com/iam/docs/understanding-roles?hl=en_US&_ga=2.73506386.-1171753218.1503062023&_gac=1.254720634.1510004312.Cj0KCQiArYDQBRDoARIsAMR8s_TiY1w4BPcHZItQ3Mkv2jSj7fw17PcWR1DwvZZMn9nvkfE8PO6YsEaAobUEALw_wcB#crm_name_short_roles)

**Project Lien Modifier:** A lien represents an encumbrance on the actions that can be performed on a resource:

<https://cloud.google.com/resource-manager/reference/rest/v1/liens>

## GCP vs. G Suite Admin



Both G Suite and GCP are part of a single product line called "Google Cloud," but they are separate products.

When "Organization" is added to GCP, an MSA (Master Services Agreement) is signed that usually includes a Google-hosted domain with the G Suite Admin product. Best practice is that the account/person who is the G Suite Super Admin is different from the account/person who is the *first* GCP Organization Owner.

G Suite Admin has TWO functions with respect to GCP:

1. The G Suite Super Administrator can assign GCP Organization Owner to an account from within the G Suite Admin console. A GCP Organization Owner can also create more Organization Owners by assigning the role to an account within the GCP Console. The G Suite Super Admin cannot assign any other GCP roles to accounts from the Admin console.
2. The G Suite Super Admin creates users and groups, controls membership of users in groups, and controls Google-hosted domains (domain names: @mycompany1.com, @mycompany2.com).

## GPC authorization

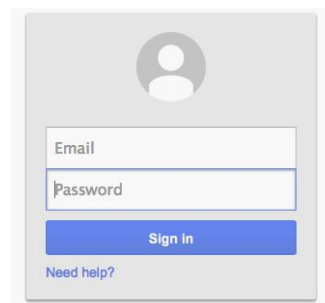
Use Google's credential system

- Manage accounts using Google G Suite \*\*
- Sync existing credentials using Google Cloud Directory Sync
- Optionally implement single sign-on (SSO)

Built-in features

- Session activity tracking
- Session management tools
- Security alerts
- Suspicious activity detection

\*\* or Google G Suite for Education and Google G Suite for Government



 Google Cloud

### Notes:

The simplest mechanism for accessing Google Cloud Platform is to use a Google account. While simple, this mechanism does not provide centralized identity management. Organizations should instead use Google G Suite user management to create and manage accounts/credentials. Note: This is for user management only. It does not require you to use Google G Suite products like Gmail, Google docs, and Google Slides.

Google G Suite user management is a single place to manage and control accounts, including suspending bad accounts. Because Google manages logins, you get the benefits and security of Google authentication management: Password resets, session and device management (when/where people are logging in), and suspicious activity detection and alerts.

For more information on managing Google G Suite user accounts, see:

[https://support.google.com/a/topic/14588?hl=en&ref\\_topic=2425090](https://support.google.com/a/topic/14588?hl=en&ref_topic=2425090).

## Google Cloud Directory Sync (GCDS)

G Suite



- Synchronizes G Suite accounts to match the user data in existing LDAP or MS Active Directory
  - Synchs groups and memberships, not content or settings
  - Supports sophisticated rules for custom mapping of users, groups, non-employee contacts, user profiles, aliases, and exceptions
- One-way synchronization from LDAP to directory
  - Administer in LDAP, then periodically update to G Suite
- Runs as a utility in your server environment

With Google Cloud Directory Sync (GCDS), the G Suite Admin can automatically add, modify, and delete users, groups, and non-employee contacts to synchronize the data in a G Suite domain with an LDAP directory server or MS Active Directory. The data in the LDAP directory server is never modified or compromised. GCDS is a secure tool that help keep track of users and groups.

The G Suite Admin uses the GCDS Configuration Manager to customize synchronizations and can perform test synchronizations to find what works best for the organization and then schedule synchronizations to occur when needed.

<https://support.google.com/a/answer/106368?hl=en>

## Single sign-on (SSO)

- Use your own authentication mechanism and manage your own credentials.
- Federate your identities to Google Cloud Platform (GCP).
- Users do not have to sign in a second time to access GCP resources.
- Revoke access to GCP using your existing credential management.
- Google Apps Directory Sync integrates with LDAP.

### Notes:

If you have your own identity system, you can enable SSO. You can continue using your own system/processes with SSO configured, and when user authentication is required, Google will redirect to your system. If the user is authenticated in your system, access to Google Cloud Platform is given. Otherwise, the user is prompted to sign in.

Your users must have a corresponding account in Google's system (a matching username), typically provisioned using Google Apps Directory Sync.

## Setting up SSO

- If your existing auth supports SAML2, SSO configuration is 3 links and a certificate.
  - If SAML2 not supported, use a third-party solution.

Sign-in page URL	https://sso.weston-widgets.com/auth
	URL for signing in to your system and Google Apps
Sign-out page URL	https://sso.weston-widgets.com/logout
	URL for redirecting users to when they sign out
Change password URL	https://sso.weston-widgets.com/info
	URL to let users change their password in your system; when defined here, this is shown even when Single Sign-on is not enabled
Verification certificate	A certificate file has been uploaded. <a href="#">Replace certificate</a>
	The certificate file must contain the public key for Google to verify sign-in requests. ?

### Notes:

SSO configuration is a relatively simple process. SSO is built on SAML2, a secure/industry-standard protocol for exchanging user assertions. With Google SSO, the only assertion that is used is the username.

To prevent tampering, SAML allows information to be digitally signed. To configure this, you need a digital certificate used to validate the signature. This ensures that incoming information originated from you and has not been tampered with. If your system doesn't support SAML2, you can use a third-party ID management plug-in such as Ping or Okta.



## Cloud IAM best practices

### Principle of least privilege

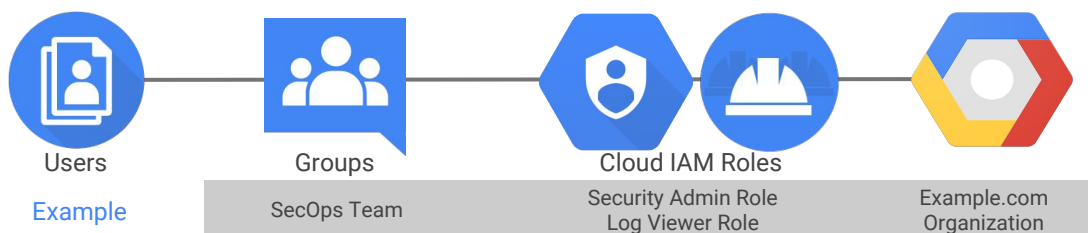
- Always apply the minimal access level required.

### Use groups.

Control who can change policies and group memberships.

### Audit policy changes.

- Audit logs record project-level permission changes.
- Additional levels are being added.



### Notes:

The principle of least privilege applies to identities, roles, and resources. Always select the smallest scope necessary to reduce your exposure to risk. You wouldn't grant everybody the owner role on your entire organization: intentional hacks or accidental mistakes could bring down your applications. You want to be specific and deliberate. Assign a specific group the security admin role to manage SSL certificates and firewalls on specific projects.

Managing permissions for individual users can be cumbersome and error-prone. Use groups instead. In the example, there is a SecOps group (for your security operations team). When new members join the team, add them to the group. The SecOps team probably needs multiple roles; for example, Security Admin to manage firewalls and Log Viewer for auditing. Assign the relevant roles to the appropriate group.

Policies allow you to secure your resources. You also want to make sure you control how additional users gain access to resources through policies and group memberships. Without strict control over policy changes and group memberships, you may inadvertently allow new users more permissions than they need (which also violates the principle of least privilege).

## Best practices: Use groups

If group membership is secure, assign roles to groups and let the **G Suite** Admins handle membership.

- Always maintain an alternate.
- For high-risk areas, assign roles to individuals directly and forego the convenience of group assignment.

Vary from the standard only when you need to and when you know why you need to make the change and how your policies will be implemented.

## Agenda

- Cloud Identity and Access Management (IAM)
- Organization
- **Roles**
- Members
- Service accounts
- Cloud IAM best practices
- Quiz
- Lab

## Primitive roles

*A project can have multiple owners, editors, viewers, and billing administrators.*



### Owner

- Invite members
- Remove members
- Can delete project
- Includes Editor rights



### Editor

- Deploy applications
- Modify code
- Configure services
- Includes Viewer rights



### Viewer

- Read-only access



### Billing administrator

- Manage billing
- Add administrators
- Remove administrators

There are two kinds of roles in Cloud IAM:

- **Primitive roles:** The original roles available in the Google Cloud Platform Console. These are the Owner, Editor, and Viewer roles. Still assigned by default to projects. Primitive roles are quite broad.
- **Curated roles:** Curated roles are new Cloud IAM roles that give finer-grained access control than the primitive roles (discussed in the next section).

The permissions granted by the primitive roles are:

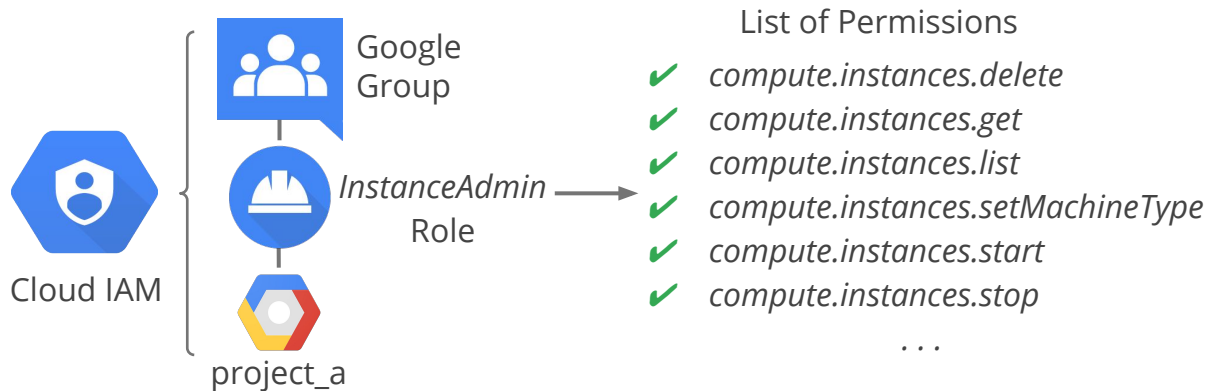
- “is Owner” allows full administrative access. This includes the ability to add members and set the authorization level of team members.
- “can Edit” allows modify and delete access. This allows a developer to deploy the application and modify or configure its resources.
- “can View” allows read-only access.

It is a best practice to ensure that a project has more than one owner for continuity reasons. Many organizations will create a Google Group for project ownership, and put at least two Google accounts into that group. Some organizations use a dedicated project owner account. If there is only one owner, and the sole owner’s account is deleted, the entire project would also be deleted.

The recommended practice is to use Google accounts for your project’s team members. When someone leaves the company, your G Suite administrator should use the G Suite [Admin Console](#) to mark the account as [suspended](#) (instead of deleting the account.) The domain administrator can then perform various tasks, such

as reassigning ownership of apps and docs, before deleting the team member's account.

## Curated roles



Cloud IAM curated roles provide significantly greater granularity for permission-granting.

A Cloud IAM role is a collection of permissions. Most of the time, to do any meaningful operations, you need more than 1 permission. For example, to manage instances in a project, you need to create, delete, start, stop, and change an instance. So the permissions are grouped together into a role to make it easier to manage.

To give a user the desired permissions, you grant a role to the user on a resource. In this slide, a group of users is granted the InstanceAdmin role on a project so the users can manage instances in the project. Whenever possible, it is a best practice to use groups. You should also strictly control the ability to change policies and group memberships that will allow additional users to gain access to resources.

For a complete list of roles by product, see:

[https://cloud.google.com/iam/docs/#supported\\_cloud\\_platform\\_services](https://cloud.google.com/iam/docs/#supported_cloud_platform_services)

## Roles

- Groups of permissions
  - Represent *abstract functions*
  - Customize roles to align with *real jobs*
- Permissions are classes and methods in the APIs
  - `<service>.<resource>.<verb>`
  - Usually (but not always) 1:1 with REST API
- Roles can be customized <sup>BETA</sup>

Google provides a set of curated roles for products.

Read the product-specific Cloud IAM document to understand why the roles were defined as they are and why some permissions were included and not others.

You really need to understand the API for the product to modify the permissions so you know exactly what behavior you are enabling or disabling.

Permissions are very granular controls.

<https://cloud.google.com/iam/docs/understanding-roles>

<https://cloud.google.com/iam/docs/faq>

Customizing Roles is in beta:

<https://cloud.google.com/iam/docs/understanding-custom-roles>

## Essential roles

- Organization roles
- Folder roles
- Project roles
- Product-specific roles
  - Crafted for each resource/product (20+)
  - Product-specific Cloud IAM documentation
  - Some in Beta

There are several kinds of roles. This course will look at each of them in context.

Roles are collections of "permissions." Permissions directly map to classes and methods in the Google Cloud Platform APIs.

When a role is granted, the permissions it contains authorize the user to call specific methods in the APIs.



## Project roles

- Viewer
  - Read-only actions that do not modify state
  - Browser role (Beta) - only "information about" (metadata)
- Editor
  - + actions that modify state
- Owner
  - + manage access control for a project and all its resources
  - + set up project billing

<https://cloud.google.com/iam/docs/understanding-roles>

One user with the Project Owner role for a particular project can invite another user to become a Project Owner also.

## Project owner invitation and acceptance

The diagram illustrates the process of sending a project owner invitation. At the top, a user interface shows an invitation sent to 'someoneyouknow@gmail.com' with the role 'Owner'. A green arrow labeled 'email' points from this UI to an email preview. The email, titled 'Welcome, Somone Youknow!', is from 'console' and states: 'The owner of the project: deadpool-cpb100 has invited you to join the project'. It includes 'DECLINE' and 'ACCEPT INVITATION' links. A yellow arrow points from the 'ACCEPT INVITATION' link back to the user interface at the bottom, which now shows 'Somone Youknow' as an 'Owner' of the project.

- Only sent for **project owner** role, not other roles
- Not sent to organization owners when assigned project ownership by other organization owners

Welcome, Somone Youknow! **console**

The owner of the project: deadpool-cpb100 has invited you to join the project

[DECLINE](#) [ACCEPT INVITATION](#)

Somone Youknow  
someoneyouknow@gmail.com

Owner

Member invitation and acceptance provides identity verification and an accountability trail.

Anyone with a valid Google identity is a potential invitee. There is currently no method to set limits or block specific domains, groups, or emails.

No email is sent when you're granting a role other than the Owner.

No email is sent when an organization Owner adds another organization Owner as an owner of a project within that organization.

Organization Owners don't get invitation emails; they are just added.

## Product-specific roles

- Roles crafted for each product - *read the Cloud IAM doc!*
- Example: Compute Engine roles
  - Compute Engine Instance Admin: [VMs and disks](#)
  - Service Account Actor: [service accounts](#)
  - Compute Engine Image User: [images](#)
  - Compute Engine Network Viewer: [read-only for all networking](#)
  - Compute Engine Network Admin: [all networking except firewall rules and certificates](#)
  - Compute Engine Security Admin: [firewall rules, ssl certificates](#)
  - Compute Engine Storage Admin: [disks, images, snapshots](#)

<https://cloud.google.com/iam/docs/understanding-roles>

There are 20+ products with product-specific roles. Each has a separate Cloud IAM document explaining the permission details and the strategy behind the role definition. No invitation and acceptance with product-specific roles.

Compute Engine example:

<https://cloud.google.com/compute/docs/access/iam>

## Product-specific roles: Examples



### Compute Engine

Compute Network Admin  
 Compute Network Viewer (Beta)  
 Compute Security Admin  
 Compute Instance Admin  
 Compute Storage Admin (Beta)



### Cloud Storage

Storage Admin  
 Storage Object Admin  
 Storage Object Creator  
 Storage Object Viewer



### Logging

Logs Viewer  
 Logs Writer  
 Private Logs Viewer  
 Logs Config Writer



### BigQuery

BigQuery Viewer  
 BigQuery Editor  
 BigQuery Admin  
 BigQuery User



### App Engine

App Engine Admin  
 App Engine Deployer  
 App Engine Service Admin  
 App Engine Viewer



### Cloud Dataflow

Dataflow Viewer  
 Dataflow Developer  
 Dataflow Worker



### Cloud Pub/Sub

Pub/Sub Viewer  
 Pub/Sub Editor  
 Pub/Sub Admin  
 Pub/Sub Publisher  
 Pub/Sub Subscriber

Other roles:

Deployment Manager  
 Stackdriver debugger  
 Genomics

...

For full list, see [docs](#)

## Agenda

- Cloud Identity and Access Management (IAM)
- Organization
- Roles
- **Members**
- Service accounts
- Cloud IAM best practices
- Quiz
- Lab

# Members

## Users

- Google accounts: @gmail, @google
- **G Suite** domains: Google-hosted domains (*mydomain.com*)
- Google Groups (*groupname@mydomain.com*)
- **GCP does not create or manage users or groups**
  - **G Suite Admin** Super Administrator manages users and groups for an organization. "**G Suite Admin**" is a separate product from GCP.

## Service accounts








- Created and managed in GCP

example.com  
groups.google.com/a/your-domain.com  
Groups for Business feature  
groups.your-domain.com


## GCP access without Gmail

Create your Google Account

One account is all you need  
One free account gets you into everything Google.

Take it all with you  
Switch between devices, and pick up wherever you left off.



**Name**  
First  Last


**Your email address**  
  
[I would like a new Gmail address](#)

**Create a password**

**Confirm your password**

**Birthday**  
Month  Day  Year

**Gender**  
I am...

**Mobile phone**  


**Location**  
United States

[Next step](#)

- You can just use Gmail.
- You can get a Google password without Gmail.
- There are benefits to having a domain, including group permissions.

<https://accounts.google.com/SignUpWithoutGmail?hl=en>

## Agenda

- Cloud Identity and Access Management (IAM)
- Organization
- Roles
- Members
- **Service accounts**
- Cloud IAM best practices
- Quiz
- Lab



## Service accounts (1 of 2)

- Provide an identity for carrying out **server-to-server** interactions in a project without supplying user credentials.
- Used to **authenticate** from one service to another:
  - Programs running within Compute Engine instances can automatically acquire access tokens with credentials
  - Token used to access any service API in your project and any other services that granted access to that service account
  - Convenient when not accessing user data

A service account is an identity for your programs to use to authenticate and gain access to Google Cloud Platform APIs. Service accounts authenticate applications running on your virtual machine instances to other Google Cloud Platform services. For example, if you write an application that reads and writes files on Cloud Storage, it must first authenticate to the Google Cloud Storage XML API or JSON API. You can enable service accounts and grant read-write access to the account on the instance where you plan to run your application. Then, program the application to obtain credentials from the service account. Your application authenticates seamlessly to the API without embedding any secret keys or credentials in your instance, image, or application code.

## Service accounts (2 of 2)

- Identified by an **email** address:
  - `123845678986-compute@project.gserviceaccount.com`
- Three types of service accounts:
  - User-created (custom)
  - Built-in
    - Compute Engine and App Engine default service accounts
  - Google APIs service account
    - Runs internal Google processes on your behalf

By default, all projects come with the Compute Engine default service account. When you start a new instance using `gcloud`, the default service account is enabled on that instance.

Apart from the default service account, all projects come with a Google Cloud Platform APIs service account, identifiable using the email:

`{project-number}@cloudservices.gserviceaccount.com`

This is a service account designed specifically to run internal Google processes on your behalf. This account is not listed in the Service Accounts section of the Google Cloud Platform Console, but by default, it is automatically granted the Editor role on the project and is listed in the Cloud IAM section of the Google Cloud Platform Console. This service account is only deleted when the project is deleted. However, you can change the roles granted to this account, including revoking all access to your project. Certain resources rely on this service account and the default editor permissions granted to the service account. For example, managed instance groups and autoscaling use the credentials of this account to create, delete, and manage instances. If you revoke permissions to the service account, or modify the permissions in such a way that it does not grant permissions to create instances, this will cause managed instance groups and autoscaling to stop working. For these reasons, it is recommended that you do not modify this service account's roles.

Alternatively, you can also start an instance with a custom service account. Custom

service accounts provide more flexibility than the default service account, but they require more management from you. You can create as many custom service accounts as you need, assign any arbitrary access scopes or Cloud IAM roles to them, and assign the service accounts to any virtual machine instance.

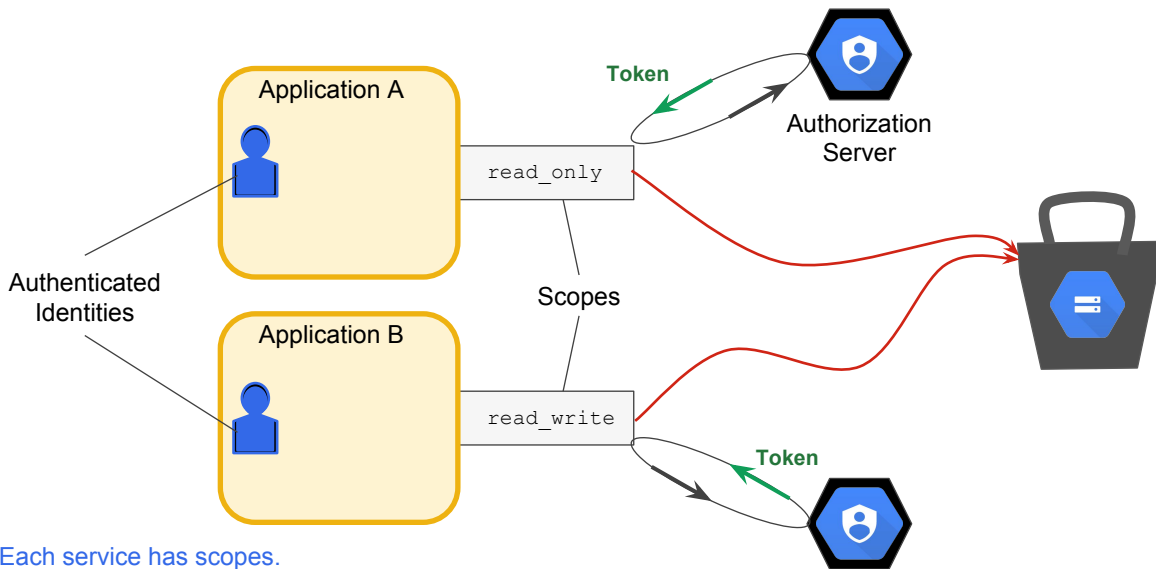
App Engine default service accounts are beyond the scope of this course. For more information, see:

<https://developers.google.com/identity/protocols/application-default-credentials>.

## Default Compute Engine service account

- Automatically created per project with auto-generated name and email address:
  - Name has -compute suffix  
`39xxxx0965-compute@developer.gserviceaccount.com`
- Automatically added as a project Editor
- By default, enabled on all instances created using `gcloud` or GCP Console
  - Override by specifying another service account or by disabling service accounts for the instance

## Scopes



<https://cloud.google.com/storage/docs/authentication>

OAuth uses scopes to determine whether an authenticated identity is authorized for access to a resource.

Applications use scopes in the access process.

In this example, Applications A and B contain Authenticated Identities (service accounts).

Both of them want to use a Cloud Storage bucket.

They request access from Google Authorization Servers.

An access token is returned with the scope.

Application A can only read from the bucket.

Application G can read and write.

But neither application can read or modify ACLs (Access Control Lists), because this would require the `full_access` scope that neither has.

Access scopes grant access only if the respective API has been enabled on the project that the service account belongs to. For example, granting an access scope for Cloud Storage on a virtual machine instance allows the instance to call the Firebase Cloud Storage API only if you have enabled the Cloud Storage API on the project. If the API is not enabled on the project, the access scope has no effect. Additionally, if you plan to use the service account to access another project, you must grant the service account the appropriate Cloud IAM roles on the target project.

Default Compute Engine service account automatically enabled with the following access scopes:

<https://www.googleapis.com/auth/cloud.useraccounts.readonly>

[https://www.googleapis.com/auth/devstorage.read\\_only](https://www.googleapis.com/auth/devstorage.read_only)

<https://www.googleapis.com/auth/logging.write>

<https://www.googleapis.com/auth/monitoring.write>

<https://www.googleapis.com/auth/service.management.readonly>

<https://www.googleapis.com/auth/servicecontrol>

## Customizing scopes for a VM

You can create an instance with customized scopes for your use case, using the default service account.

- Scopes can be changed after an instance is created.

For user-created service accounts, use Cloud IAM roles instead.

Identity and API access ?

Service account ?  
Compute Engine default service account ▼

Access scopes ?

☐ Allow default access

☐ Allow full access to all Cloud APIs

☒ Set access for each API

BigQuery  
None

Bigtable Admin  
None

Bigtable Data  
None

Cloud Datastore  
None

Scopes can be changed after an instance is created by stopping it.

Service accounts can use scopes through the Cloud SDK. `gcloud` and `gsutil` commands automatically pick up tokens, and you can easily run these commands in scripts to automate workflows. You can also write custom tools or application code using Google client libraries or alternatively, write your own code to consume tokens.

These scopes provide the following access

Read-only access to the Cloud User Accounts API\*

Read-only access to the Cloud Storage JSON API v1

Read/write access to the Stackdriver Logging API v2

Read/write access to the Stackdriver Monitoring API v3

Read-only access to the Google Service Management API v1\*

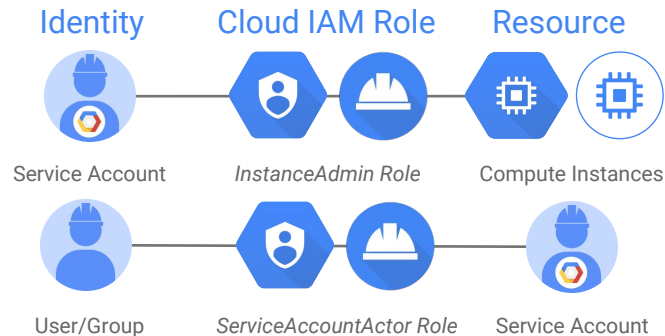
Read-write access to the Google Service Control API v1\*

## Service account permissions

Default service accounts support primitive (project) and curated (Cloud IAM) roles

- User-created service accounts use Cloud IAM roles only

[Roles](#) for service accounts can be assigned to groups or users



One of the features of a Cloud IAM service account is that you can treat it as a resource or as an identity.

### *Service accounts as a resource*

By treating a service account as a resource, you can grant permission to a user to access that service account. You can grant an Owner, Editor, Viewer, or serviceAccountActor role to a user for the service account. For example, if you want to allow a user to create a VM with the service account or authenticate as a service account, you first need to grant the user the serviceAccountActor role. In this case, the service account is the resource, and you grant a user permission to use this service account resource.

### *Service accounts as an identity*

You can grant a role to a service account to access a resource. In this case, the service account is the identity. Some examples include:

- The default Compute Engine and App Engine service accounts are granted Editor roles on the project when they are created so that the code executing in your App or VM instance has the necessary permissions. In this case, the service accounts are identities that are granted the Editor role for a resource (project).
- If you want to allow your automation to access a storage bucket, you grant the service account (that your automation uses) the permissions to read the



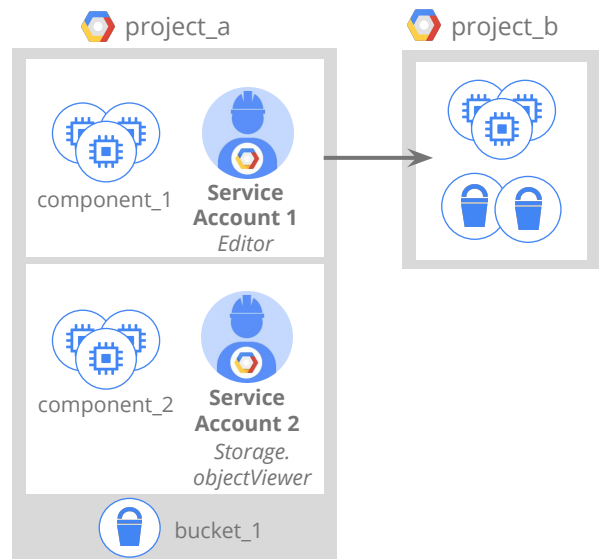
- storage bucket. In this case, the service account is the identity that you are granting permissions for another resource (Cloud Storage bucket).

#### *Granting another user the ability to act as a service account*

If you want other users to be able to use a service account, you must grant the `serviceAccountActor` role to the user. Users with the `serviceAccountActor` role can act as that service account to perform operations such as creating and managing Compute Engine instances that use a service account. For information on how to do this, see the Compute Engine documentation. Users who are `serviceAccountActors` for a service account can access all the resources for which the service account has access. Therefore be cautious when granting the `serviceAccountActor` role to a user.

## Example: Service accounts and Cloud IAM

- VMs running component\_1 are granted Editor access to project\_b using *Service Account 1*
- VMs running component\_2 are granted objectViewer access to bucket\_1 using *Service Account 2*
- Service account permissions can be changed without re-creating VMs



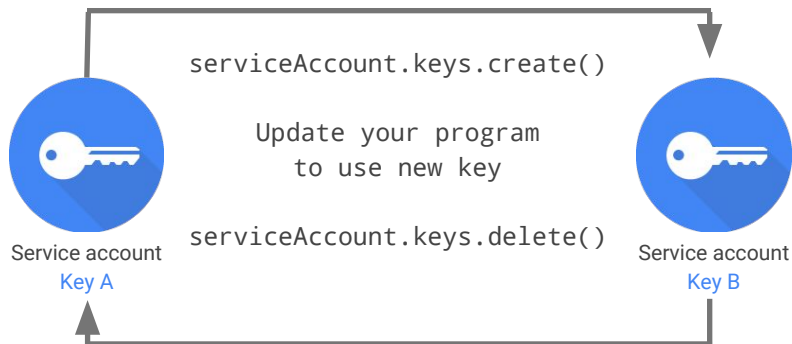
You can grant different groups of VMs in your project different identities. This makes it easier to manage different permissions for each group. For example, if one component in your app needs to have the Editor role on another project, you can have a service account with this permission that is used only by the VMs running the component. Other VMs can be assigned the permissions required by their functionalities. This way you can scope permissions for VMs. You also can change the permissions of the service accounts without having to recreate the VMs.

Cloud IAM also lets you slice a project into different microservices, each with access to different resources, by creating service accounts to represent each one. You assign the service accounts to the VMs when they are created, and you don't have to ensure that credentials are being managed correctly. Google Cloud Platform manages security for you.

## Service accounts and keys

### Service accounts authenticate with keys

- Google manages keys and key rotation for Compute Engine and App Engine
- Alternatively, create, manage, and rotate keys yourself



Users require a username and password to authenticate. Apps use a key. One or more keys can be generated for each Cloud IAM service account. Keys are sensitive and need to be carefully managed because they give you access to resources. When you run applications on Compute Engine or App Engine, Google manages the keys for you and automatically rotates them. You never have the risk of losing/exposing your key. When you run apps elsewhere, you can generate and download the keys to use in your code. It is a best practice to keep them safe and rotate them.

A service account is both an identity and a resource. A service account is used as an identity for your application to authenticate; for example, a Compute Engine VM running as a service account. To give the VM access to the necessary resources, you need to grant the relevant Cloud IAM roles to the service account. At the same time, you need to control who can create VMs with the service account so random VMs cannot assume the identity. Here, the service account is the resource to be permissioned. You assign the ServiceAccountActor role to the users you trust to use the service account.

## Agenda

- Cloud Identity and Access Management (IAM)
- Organization
- Roles
- Members
- Service accounts
- **Cloud IAM best practices**
- Quiz
- Lab

## Resource hierarchy

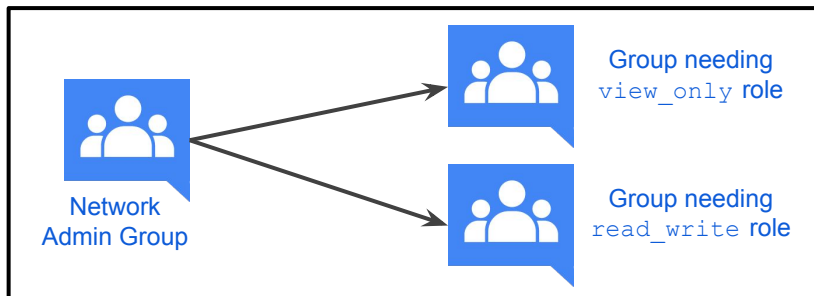
- Use projects to group resources that share the same trust boundary.
- Check the policy granted on each resource and make sure you understand the inheritance.
- Use "principle of least privilege" when granting roles.
- Audit policies in Cloud audit logs: `setiampolicy`.
- Audit membership of groups used in policies.

1. Mirror your Cloud IAM policy hierarchy structure to your organization structure.
2. Use Google Cloud Platform projects to group resources that share the same trust boundary. For example, resources for the same product or microservice can belong to the same Google Cloud Platform project.
3. Set policies at the organization level and at the project level instead of at the resource level.
4. Check the policy granted on every resource and understand the hierarchical inheritance.
5. If you need to grant a role to a user or group that spans across multiple projects, set that role at the organization level instead of setting it at the project level.
6. Use labels to annotate, group, and filter resources.
7. If you want to limit project creation in your organization, modify the organization-level policy to grant the Project Creator role to a group that you manage.
8. Use the security principle of least privilege to grant roles.
9. Grant roles at the smallest scope needed.
10. Audit your policies to ensure compliance. Cloud audit logs contain all the calls to `setiampolicy` so you are able to trace when a policy has been enacted.
11. Audit the ownership and the membership of the Google groups used in policies.
12. Grant roles at the smallest scope needed. For example, if a user only needs access to publish messages to a Pub/Sub topic, grant the Publisher role to the user for that topic.

## Groups

Grant roles to Google groups instead of individuals:

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



- Use multiple groups to get better control.
- Groups are not only associated with job roles.
- Groups can exist for the purpose of role assignment.

In the diagram, a single group was created that was associated with a job role, "network admin." However, the Cloud IAM administration soon realized that there were different sub-groups requiring different permissions. In this example, standard settings are kept in files in a bucket. Some network admins need access to view these files. A few select individuals have the authority to edit and delete these files.

The original group is still used for group mailings and for those roles that every network administrator needs, but the other groups and their membership were established only to assign additional Cloud IAM roles. Adding and removing individuals from all three groups controls their total access.

1. Grant roles to a Google group instead of to individual users when possible. It is easier to add members to and remove members from a Google group instead of updating a Cloud IAM policy to add or remove users.
2. If you need to grant multiple roles to allow a particular task, create a Google group, grant the roles to that group, and then add users to that group.
3. Control the ownership of the Google group used in Cloud IAM policies.

## Service accounts

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

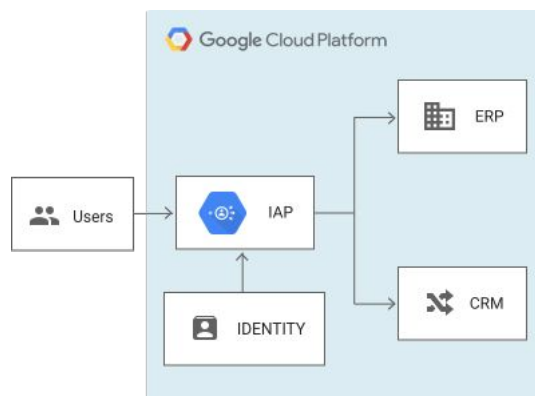
1. Restrict who can act as service accounts. Users who are Service Account Actors for a service account can access all the resources for which the service account has access. Therefore be cautious when granting the `serviceAccountActor` role to a user.
2. Grant the service account only the minimum set of permissions required to achieve its goal.
3. Create service accounts for each service with only the permissions required for that service.
4. Use the display name of a service account to keep track of the service accounts. When you create a service account, populate its display name with the purpose of the service account.
5. Define a naming convention for your service accounts.
6. Implement processes to automate the rotation of user-managed service account keys.
7. Take advantage of the Cloud IAM service account API to implement key rotation.
8. Audit service accounts and keys using either the `serviceAccount.keys.list()` method or the Logs Viewer page in the console.
9. Do not delete service accounts that are in use by running instances on App Engine or Compute Engine.

## Cloud Identity-Aware Proxy (Cloud IAP)

Enforce access control policies for applications and resources:

- Identity-based access control
- Central authorization layer for applications accessed by HTTPS

Cloud IAM policy is applied after authentication.



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

Applications and resources protected by Cloud IAP can only be accessed through the proxy by users and groups with the correct Cloud IAM role. When you grant a user access to an application or resource by Cloud IAP, they're subject to the fine-grained access controls implemented by the product in use without requiring a VPN. Cloud IAP performs authentication and authorization checks when a user tries to access a Cloud IAP-secured resource.

Cloud IAP secures authentication and authorization of all requests to App Engine or Cloud Load Balancing HTTPS. Cloud IAP doesn't protect against the following:

- Activity inside your VM, such as if someone accesses the VM via SSH. This includes the App Engine flexible environment when direct SSH access to your VM is enabled.
- Activity within a project, such as another VM inside the project.

For more information, see: <https://cloud.google.com/iap/docs/concepts-overview>



## Agenda

- Cloud Identity and Access Management (IAM)
- Organization
- Roles
- Members
- Service accounts
- Cloud IAM best practices
- **Quiz**
- Lab

## Quiz

**What abstraction is primarily used to administer user access in Cloud IAM?**

1. Leases, an abstraction of periodic entitlements
2. Roles, an abstraction of job roles
3. Credentials, an abstraction of an authorization token
4. Privileges, an abstraction of access rights

## Quiz

What abstraction is primarily used to administer user access in Cloud IAM?

1. Leases, an abstraction of periodic entitlements
2. Roles, an abstraction of job roles \*
3. Credentials, an abstraction of an authorization token
4. Privileges, an abstraction of access rights

### Explanation:

Cloud IAM administration uses pre-defined roles for administration of user access.

The roles are defined by more granular permissions. But permissions are not applied to users directly; only through the roles that are assigned to them.

## Quiz

### How is a user identity created in Cloud IAM?

1. User identities are created from the Cloud Identity console that is only visible to GCP Super-administrators.
2. User identities are created from the Cloud IAM area of the GCP Console, or by using the `gcloud` command.
3. User identities are created through a federated Active Directory domain.
4. User identities are created from outside of GCP in a Google-administered domain.

## Quiz

### How is a user identity created in Cloud IAM?

1. User identities are created from the Cloud Identity console that is only visible to GCP Super-administrators.
2. User identities are created from the Cloud IAM area of the GCP Console, or using the gcloud command.
3. User identities are created through a federated Active Directory domain.
4. User identities are created from outside of GCP in a Google-administered domain. \*

### Explanation:

Cloud IAM access is built on top of an identity authorization and access management system that is used by all Google Cloud products, not just GCP.

## Quiz

**What technology can be used along with Cloud IAM to provide another layer of security and access control in GCP?**

1. Machine learning; specifically, intrusion detection
2. Antivirus and anti-exploit software built into GCP VMs
3. Networking; specifically, firewall rules
4. Dynamic per-user resource throttling

## Quiz

**What technology can be used along with Cloud IAM to provide another layer of security and access control in GCP?**

1. Machine learning; specifically, intrusion detection
2. Antivirus and anti-exploit software built into GCP VMs
3. Networking; specifically, firewall rules \*
4. Dynamic per-user resource throttling

### **Explanation:**

Because GCP is a collection of networked services, you can administer fine-grained access to resources by limiting network access. Example: Using Cloud IAM roles, you could grant a group of users access to a particular VM running an application. Using firewall rules, you could permit access to the VM only from specific IP ranges, so that those users could only gain access to the VM from the corporate network, and not from another location.

## Agenda

- Cloud Identity and Access Management (IAM)
- Organization
- Roles
- Members
- Service accounts
- Cloud IAM best practices
- Quiz
- **Lab**



# Lab: Cloud IAM

## Objectives

In this lab, you learn how to perform the following tasks:

- Use Cloud IAM to implement access control
- Restrict access to specific features or resources
- Use the Service Account Actor role

**Completion:** 30 minutes

**Access:** 60 minutes



## Lab review

In this lab, you:

- Granted and revoked Cloud IAM roles to a:
  - User, Username 2
  - Service Account Actor

*You could allocate Service Account Actor credentials and "bake" them into a VM to create specific-purpose authorized bastion hosts.*

## More resources

Using service accounts

<https://cloud.google.com/compute/docs/access/create-enable-service-accounts-for-instances>

Authorizing requests to Google Compute Engine

<https://cloud.google.com/compute/docs/api/how-tos/authorization>

Using OAuth 2.0 to access Google APIs

<https://developers.google.com/identity/protocols/OAuth2>



© 2017 Google Inc. All rights reserved. Google and the Google logo are trademarks of Google Inc. All other company and product names may be trademarks of the respective companies with which they are associated.