Start Lab 02:15:00

# Configuring IAM Permissions with gcloud - AWS

Lab 1 hour 30 minutes No cost Intermediate



## GSP1126


## Overview

Configuring secure access to your cloud resources is paramount. Key considerations include how to manage access, restrict users to necessary resources, and enable authentication for applications and services. Cloud platforms like AWS and Google Cloud offer specialized tools for this task. AWS leverages Identity and Access Management (IAM) roles and policies, while Google Cloud IAM grants permissions to projects or resources.

Beyond user access, you must also manage application and system permissions. AWS employs service roles assigned within an account. Google Cloud uses service accounts with potentially broader reach across projects. This lab explores the command line for IAM configuration. Understanding IAM, especially for those new to Google Cloud, is crucial for setting up appropriate permissions. This lab will cover installing and configuring the gcloud environment, managing multiple configurations, and utilizing service accounts.

## Objectives

In this lab, you learn how to:

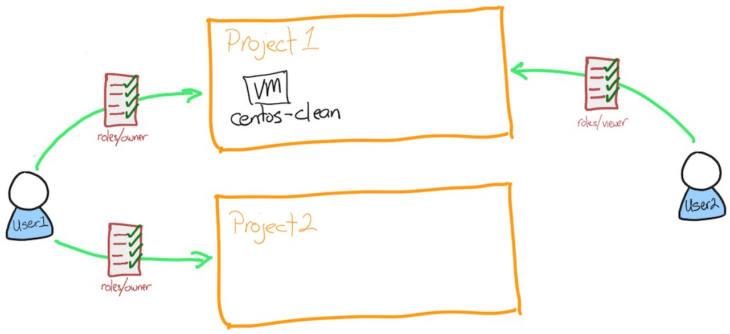
- Install and configure the `gcloud` client
- Create and switch between multiple IAM configurations
- Identify and assign correct IAM permissions
- Create and use a service account

### Starting environment

You start with two user accounts and two projects;

- `user1` is the "owner" of both projects
- `user2` is the "viewer" of only the first project.

There is a Linux virtual machine (vm) running in the first project.




# What is IAM?

Google Cloud offers Cloud Identity and Access Management (IAM), which lets you manage access control by defining who (identity) has what access (role) for which resource.

In IAM, permission to access a resource isn't granted directly to the end user. Instead, permissions are grouped into roles, and roles are granted to authenticated principals. (In the past, IAM often referred to principals as members. Some APIs still use this term.)

## Identities

In Cloud IAM, you grant access to *principals*. Principals can be of the following types:

- Google Account
- Service account
- Google group
- Google Workspace account
- Cloud Identity domain
- All authenticated users
- All users

Learn more about these identity types from the Concepts related to identity Guide.

In this lab, you use Google accounts, service accounts, and Cloud Identity domain groups.

## Roles

A role is a collection of permissions. You cannot assign a permission to the user directly; instead you grant them a role. When you grant a role to a user, you grant them all the permissions that the role contains.

Learn more about roles from the Roles Guide.

# What is gcloud?

The gcloud CLI is a part of the Cloud SDK. You must download and install the SDK on your system and initialize it before you can use the gcloud command-line tool. You can use this tool to perform many common platform tasks either from the command-line or in scripts and other automations.

Learn more about gcloud from the gcloud CLI overview Guide.

### Roles

A role is a collection of permissions. You cannot assign a permission to the user directly; instead you grant them a role. When you grant a role to a user, you grant them all the permissions that the role contains.

Learn more about roles from the Roles Guide.

## What is gcloud?

The gcloud CLI is a part of the Cloud SDK. You must download and install the SDK on your system and initialize it before you can use the gcloud command-line tool. You can use this tool to perform many common platform tasks either from the command-line or in scripts and other automations.

Learn more about gcloud from the gcloud CLI overview Guide.

### Roles

A role is a collection of permissions. You cannot assign a permission to the user directly; instead you grant them a role. When you grant a role to a user, you grant them all the permissions that the role contains.

Learn more about roles from the Roles Guide.

## What is gcloud?

The gcloud CLI is a part of the Cloud SDK. You must download and install the SDK on your system and initialize it before you can use the gcloud command-line tool. You can use this tool to perform many common platform tasks either from the command-line or in scripts and other automations.

Learn more about gcloud from the gcloud CLI overview Guide.

### Roles

A role is a collection of permissions. You cannot assign a permission to the user directly; instead you grant them a role. When you grant a role to a user, you grant them all the permissions that the role contains.

Learn more about roles from the Roles Guide.

## What is gcloud?

The gcloud CLI is a part of the Cloud SDK. You must download and install the SDK on your system and initialize it before you can use the gcloud command-line tool. You can use this tool to perform many common platform tasks either from the command-line or in scripts and other automations.

Learn more about gcloud from the gcloud CLI overview Guide.

### Roles

A role is a collection of permissions. You cannot assign a permission to the user directly; instead you grant them a role. When you grant a role to a user, you grant them all the permissions that the role contains.

Learn more about roles from the Roles Guide.

## What is gcloud?

The gcloud CLI is a part of the Cloud SDK. You must download and install the SDK on your system and initialize it before you can use the gcloud command-line tool. You can use this tool to perform many common platform tasks either from the command-line or in scripts and other automations.

Learn more about gcloud from the gcloud CLI overview Guide.

### Roles

A role is a collection of permissions. You cannot assign a permission to the user directly; instead you grant them a role. When you grant a role to a user, you grant them all the permissions that the role contains.

Learn more about roles from the Roles Guide.

## What is gcloud?

The gcloud CLI is a part of the Cloud SDK. You must download and install the SDK on your system and initialize it before you can use the gcloud command-line tool. You can use this tool to perform many common platform tasks either from the command-line or in scripts and other automations.

Learn more about gcloud from the gcloud CLI overview Guide.

### Roles

A role is a collection of permissions. You cannot assign a permission to the user directly; instead you grant them a role. When you grant a role to a user, you grant them all the permissions that the role contains.

Learn more about roles from the Roles Guide.

## What is gcloud?

The gcloud CLI is a part of the Cloud SDK. You must download and install the SDK on your system and initialize it before you can use the gcloud command-line tool. You can use this tool to perform many common platform tasks either from the command-line or in scripts and other automations.

Learn more about gcloud from the gcloud CLI overview Guide.

### Roles

A role is a collection of permissions. You cannot assign a permission to the user directly; instead you grant them a role. When you grant a role to a user, you grant them all the permissions that the role contains.

Learn more about roles from the Roles Guide.

## What is gcloud?

The gcloud CLI is a part of the Cloud SDK. You must download and install the SDK on your system and initialize it before you can use the gcloud command-line tool. You can use this tool to perform many common platform tasks either from the command-line or in scripts and other automations.

Learn more about gcloud from the gcloud CLI overview Guide.

5. In the SSH session, set the region and zone:

```
gcloud config set compute/region "Region1"
gcloud config set compute/zone "Zone1"
```

6. Inside the SSH session run:

```
gcloud compute instances create lab-1 --zone "Zone1" --machine-type
```

7. You can press ENTER to accept the default zone for this VM.

If you have correctly set everything up, the command creates an instance.

But what size? And where? What image is used?

There are a number of defaults the service uses. Some can be controlled in the `gcloud` configuration. For example, the location of the instance is controlled by the zone setting.

Create an instance with name as lab-1 in Project 1

Check my progress

8. Check your current gcloud configuration. Inside the SSH session run:

```
gcloud config list
```

You now see a `compute` section, a `core` section, and an `active configuration`. You can change each of these, but for this lab you'll only change the zone. Look at the zone your VM was created in.

9. Now list all the zones available to use by running the following inside the SSH session run:

```
gcloud compute zones list
```

10. Identify one of the other zones in the same region as you. For example, if your current zone is `us-west2-a`, you could select `us-west2-b`.

11. Change your current zone for another zone in the same region. Inside the SSH session run the following, replacing `ZONE` with the zone you selected:

```
gcloud config set compute/zone ZONE
```

12. Verify the zone change was made. Inside the SSH session run:

```
gcloud config list
```

You see the stated zone reflects the change you made.

You can change other settings using the `gcloud config set` command. Those changes are permanent; they are written to your home directory.

The default configuration is stored in **~/.config/gcloud/configurations/config_default**.

If you want to use a zone other than the default zone when creating an instance, you can use --zone switch. For example, `gcloud compute instances create lab-1 --zone us-central1-f`

Update the default zone

11. Change your current zone for another zone in the same region. Inside the SSH session run the following, replacing `ZONE` with the zone you selected:

```
gcloud config set compute/zone ZONE
```

12. Verify the zone change was made. Inside the SSH session run:

```
gcloud config list
```

You see the stated zone reflects the change you made.

You can change other settings using the `gcloud config set` command. Those changes are permanent; they are written to your home directory.

The default configuration is stored in **~/.config/gcloud/configurations/config_default**.

If you want to use a zone other than the default zone when creating an instance, you can use --zone switch. For example, `gcloud compute instances create lab-1 --zone us-central1-f`

Update the default zone

11. Change your current zone for another zone in the same region. Inside the SSH session run the following, replacing `ZONE` with the zone you selected:

```
gcloud config set compute/zone ZONE
```

12. Verify the zone change was made. Inside the SSH session run:

```
gcloud config list
```

You see the stated zone reflects the change you made.

You can change other settings using the `gcloud config set` command. Those changes are permanent; they are written to your home directory.

The default configuration is stored in **~/.config/gcloud/configurations/config_default**.

If you want to use a zone other than the default zone when creating an instance, you can use --zone switch. For example, `gcloud compute instances create lab-1 --zone us-central1-f`

11. Change your default zone another zone in the same region. Inside the SSH session run the following, replacing `ZONE` with the zone you selected:

```
gcloud config set compute/zone ZONE
```

12. Verify the zone change was made. Inside the SSH session run:

```
gcloud config list
```

You see the stated zone reflects the change you made.

You can change other settings using the `gcloud config set` command. Those changes are permanent; they are written to your home directory.

The default configuration is stored in **~/.config/gcloud/configurations/config_default**.

If you want to use a zone other than the default zone when creating an instance, you can use --zone switch. For example, `gcloud compute instances create lab-1 --zone us-central1-f`

11. Change your default zone another zone in the same region. Inside the SSH session run the following, replacing `ZONE` with the zone you selected:

```
gcloud config set compute/zone ZONE
```

12. Verify the zone change was made. Inside the SSH session run:

```
gcloud config list
```

You see the stated zone reflects the change you made.

You can change other settings using the `gcloud config set` command. Those changes are permanent; they are written to your home directory.

The default configuration is stored in **~/.config/gcloud/configurations/config_default**.

If you want to use a zone other than the default zone when creating an instance, you can use --zone switch. For example, `gcloud compute instances create lab-1 --zone us-central1-f`

11. Change your default zone another zone in the same region. Inside the SSH session run the following, replacing `ZONE` with the zone you selected:

```
gcloud config set compute/zone ZONE
```

12. Verify the zone change was made. Inside the SSH session run:

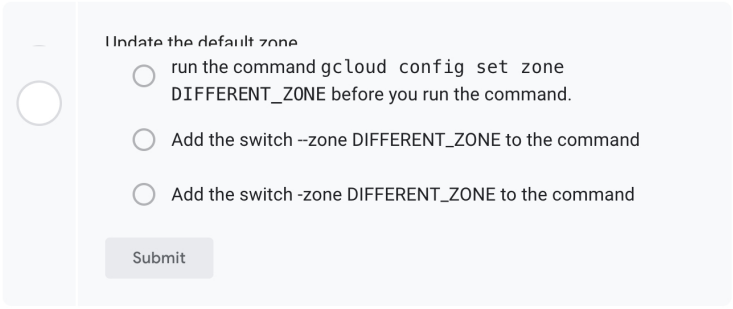
```
gcloud config list
```

You see the stated zone reflects the change you made.

You can change other settings using the `gcloud config set` command. Those changes are permanent; they are written to your home directory.

The default configuration is stored in **~/.config/gcloud/configurations/config_default**.

If you want to use a zone other than the default zone when creating an instance, you can use --zone switch. For example, `gcloud compute instances create lab-1 --zone us-central1-f`




## Task 3. Identify and assign correct IAM permissions

You have been provided two user accounts for this project. The first user has complete control of both projects and can be thought of as the admin account. The second user has viewer only access to the two projects. Call the second user a devops user and that user identity represents a typical devops level user.

Next you use `gcloud` to configure access to one project for the devops user by creating a custom role for the project that permits creation of buckets and instances.

### Examine roles and permissions

1. To view all the roles, run the following inside the SSH session run:

```
gcloud iam roles list | grep "name:"
```

The list of roles is returned. The addition of `grep "name:"` to the command reduces the amount of data returned to just the names of the roles.

Inspect one of these roles to see the permissions assigned to the role. To view the permissions use `gcloud iam roles describe`. Try looking at the simple role **roles/compute.instanceAdmin**.

2. Examine the `compute.instanceAdmin` predefined role. Inside the SSH session run:

```
gcloud iam roles describe roles/compute.instanceAdmin
```

You can see **roles/compute.instanceAdmin** has many permissions, but these are the minimum needed for later:

- compute.instances.create
- compute.instances.delete
- compute.instances.start
- compute.instances.stop
- compute.instances.update
- compute.disks.create

- compute.subnetworks.use
- compute.subnetworks.useExternalIp
- compute.instances.setMetadata
- compute.instances.setServiceAccount

To review the full list of roles and the permissions assigned, refer to the IAM permissions reference Guide.

## Grant access to the second user to the second project

Now that you know that roles contain permissions, how do you assign a role (and therefore all the associated permissions), to a user account?

There are two ways to attach a role:

- To the user and an organization
- To a user and a project

Next you attach the basic role of "viewer" to the second user onto the second project.

**Test that the second user doesn't have access to the second project.**

1. Switch `gcloud` configuration back to the second user (**user2**). Inside the SSH session run:

```
gcloud config configurations activate user2
```

Now you're back to `user2`.

2. Set `PROJECTID2` to the second project. Inside the SSH session, run the following:

```
echo "export PROJECTID2="PROJECT_ID"" >> ~/.bashrc
```

```
. ~/.bashrc
gcloud config set project $PROJECTID2
```

**Note:** This command appends the `bashrc` file, so be careful.

You get a warning: `WARNING: You do not appear to have access to project [your 2nd project id] or it does not exist.`

3. When prompted, *Do you want to continue (Y/n)?*, type N and press ENTER.

This means that user 2 doesn't have access to the PROJECTID2 project, which you fix in the next section.

**Assign the viewer role to the second user in the second project**

1. Switch back to the **default** gcloud configuration, which has the permission to grant access to the second user. Inside the SSH session run:

```
gcloud config configurations activate default
```

2. Install `jq`:

```
sudo yum -y install epel-release
sudo yum -y install jq
```

Next, set the value of `USERID2` to the second user name and bind the role of viewer to the second user onto the second project.

3. Inside the SSH session, run the following:

```
echo "export USERID2="Username2"" >> ~/.bashrc
```

```
. ~/.bashrc
gcloud projects add-iam-policy-binding $PROJECTID2 --member user:$U
```

Once you have run the command, the text looks something like the following (you may need to scroll up):

```
Updated IAM policy for project ["PROJECT_ID"].
bindings:
...

- members:
  - serviceAccount:"PROJECT_ID"@"PROJECT_ID".iam.gserviceaccount.com
  role: roles/storage.admin
- members:
  - user:"Username1"
  - user:"Username2"
  role: roles/viewer
```

Restricting Username 2 to roles/viewer in Project 2

Check my progress

## Task 4. Test that user2 has access

1. Switch your gcloud configuration to **user2**. Inside the SSH session run:

```
gcloud config configurations activate user2
```

2. Change the configuration for user2 to the second project. Inside the SSH session run:

```
gcloud config set project $PROJECTID2
```

You should not see an error message this time.

3. Verify you have viewer access. Inside the SSH session run:

```
gcloud compute instances list
```

You now see 0 instances in this project.

4. Try to create an instance in the second project as the second user. Inside the SSH session run:

```
gcloud compute instances create lab-2 --zone "Zone2" --machine-type
```

This command fails because user2 only has viewer access to the project.

5. Switch your gcloud configuration to **default**. Inside the SSH session run:

```
gcloud config configurations activate default
```

You are now back to using your original user account credentials.

## Create a new role with permissions

Next, create the new role with the set of permissions needed for the devops team.

- Create a custom role called `devops` that has the permissions to create an instance. Inside the SSH session run:

```
gcloud iam roles create devops --project $PROJECTID2 --permissions
```

This command creates a custom role in the project called `devops` with the permissions to create and manage instances.

The full name of the role is listed, note the role is in the project so the path is in the pattern of `projects/PROJECT/roles/ROLENAME`.

Create a new role with permissions for the devops team

Check my progress

## Bind the role to the second account to both projects

You now have the role created and need to bind the user and the role to the project. Use `gcloud projects add-iam-policy-binding` to perform the binding. To make this command easier to execute, set a couple of environment variables first; the project id and the user account.

1. Bind the role of `iam.serviceAccountUser` to the second user onto the second project. Inside the SSH session run:

```
gcloud projects add-iam-policy-binding $PROJECTID2 --member user:$U
```

You need permissions to create an instance with a service account attached. The role `iam.serviceAccountUser` has those permissions, so use this pre-defined role.

Check user2 is bound to project2 and the role roles/iam.serviceAccountUser

Check my progress

2. Bind the custom role `devops` to the second user onto the second project. You can find the second user account on the left of this page. Make sure you set USERID to the second user account.

Inside the SSH session run:

```
gcloud projects add-iam-policy-binding $PROJECTID2 --member user:$U
```

Once you have run the command, the text that looks something like the following (you may need to scroll up):

```
Updated IAM policy for project ["PROJECT_ID"].
bindings:
- members:
```

```
- user:"Username2"@qwiklabs.net
role: projects/"PROJECT_ID"/roles/devops
```

Bound Username 2 to devops role

Check my progress

## Test the newly assigned permissions.

1. Switch your gcloud configuration to **user2**. Inside the SSH session run:

```
gcloud config configurations activate user2
```

Now you're back to user2.

2. Try to create an instance called lab-2. Inside the SSH session run:

```
gcloud compute instances create lab-2 --zone "Zone2" --machine-type
```

Now the instance creation works for user2.

Create an instance with name as lab-2 in Project 2

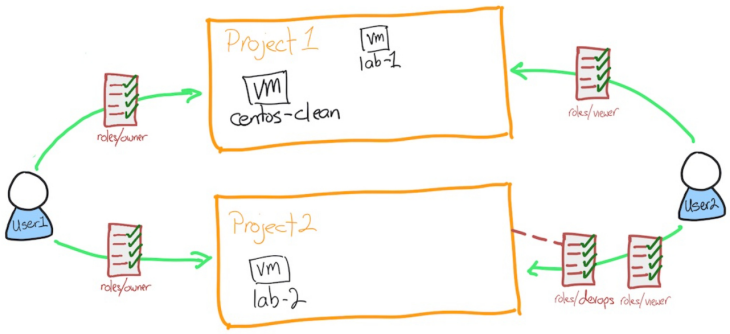
Check my progress

3. Verify the instance exists. Inside the SSH session run:

```
gcloud compute instances list
```

## Your environment

After these last changes your environment looks like this:




What are two of the three items you need to provide when binding an IAM role to a project?

- [ ] network
- [ ] project id
- [ ] account
- [ ] zone

service identifier

Submit

## Task 5. Using a service account

You have seen how to authenticate and use `gcloud` to access Google Cloud services with roles. Now you'll look at a typical approach.

You have an application that uses the Application Programming Interfaces (APIs) to read and write to Cloud Storage buckets. You don't want to have to authenticate every time you launch a new server, that would be both painful and not in the spirit of using the cloud! So, you use *service accounts*.

A service account is a special Google account that belongs to your application or a virtual machine (VM) instead of to an individual end user. Your application uses the service account to call the Google API of a service so that the users aren't directly involved.

Learn more about service accounts from the Service accounts Guide.

Now you create a service account, use that service account with a compute instance, then test that the service account allows the access you need.

### Create a service account

1. Switch your gcloud configuration to **default**, `user2` doesn't have the rights to set up and configure service accounts. Inside the SSH session run:

```
gcloud config configurations activate default
```

2. Set the project to `PROJECTID2` in your configuration. Inside the SSH session run:

```
gcloud config set project $PROJECTID2
```

Make sure you are targeting the right project.

3. Create the service account. Inside the SSH session run:

```
gcloud iam service-accounts create devops --display-name devops
```

Check the created devops service account

Check my progress

4. Get the service account email address. Inside the SSH session run:

```
gcloud iam service-accounts list  --filter "displayName=devops"
```

**Note:** The filter shows only the line you are interested in. Notice that the email address contains the role name and the project id.

5. Put the email address into a local variable called `SA`. Inside the SSH session run:

```
SA=$(gcloud iam service-accounts list --format="value(email)" --fil
```

This command sets the SA local variable to the email address of the service account. Pretty useful right?

6. Give the service account the role of `iam.serviceAccountUser`. Inside the SSH session run:

```
gcloud projects add-iam-policy-binding $PROJECTID2 --member service
```

This role allows the service account to assign a service account to a compute instance.

Check devops service account is bound to project2 and the role roles/iam.serviceAccountUser

Check my progress

## Task 6. Using the service account with a compute instance

1. Give the service account the role of `compute.instanceAdmin`. Inside the SSH session run:

```
gcloud projects add-iam-policy-binding $PROJECTID2 --member service
```

This role allows the service account to manage compute instances.

Check devops service account is bound to project2 and the role roles/compute.instanceAdmin

Check my progress

2. Create an instance with the devops service account attached. You also have to specify an access scope that defines the API calls that the instance can make. Inside the SSH session run:

```
gcloud compute instances create lab-3 --zone "Zone2" --machine-type
```

Access scopes are the legacy method of specifying permissions for your instance. Access scopes are not a security mechanism. Instead, they define the default OAuth scopes used in requests from the `gcloud` tool or the client libraries. They have no effect when making requests not authenticated through OAuth, such as gRPC or the SignBlob APIs.

You must set up access scopes when you configure an instance to run as a service account.

A best practice is to set the full cloud-platform access scope on the instance, then securely limit the service account's API access with IAM roles.

Access scopes apply on a per-instance basis. You set access scopes when creating an instance and the access scopes persist only for the life of the instance.

Access scopes have no effect if you have not enabled the related API 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 Cloud Storage API only if you have enabled the Cloud Storage API on the project.

Check lab-3 has the service account attached

Check my progress

## Task 7. Test the service account

1. Connect to the newly created instance using `gcloud compute ssh`. Inside the SSH session run:

```
gcloud compute ssh lab-3 --zone "Zone2"
```

Press ENTER when asked if you want to continue.

Press ENTER twice to skip making a password.

2. The default image used already contains `gcloud` configuration. Inside the SSH session run:

```
gcloud config list
```

The configuration now has the service account

3. Create an instance. This tests that you have the necessary permissions via the service account:

```
gcloud compute instances create lab-4 --zone "Zone2" --machine-type
```

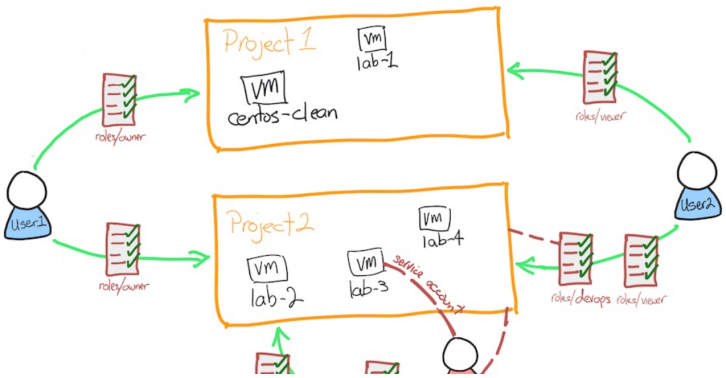
You can press ENTER to accept the default zone for this VM.

4. Check roles attached are working. Inside the SSH session run:

```
gcloud compute instances list
```

Because the service account has permissions, you can see the instances listed.

### Your environment now looks like this




What is NOT true about service accounts?

- It allows automated deployments of resources.
- It prevents a user from directly getting involved in setting up access on the instance.
- Service accounts always provide full admin rights to the project.
- Service accounts can be assigned only the rights necessary for the access required.

Submit

# Congratulations!

Using the Cloud SDK tool (`gcloud`), you've successfully installed and configured the gcloud client, managed multiple IAM configurations, assigned appropriate IAM permissions, and worked with a service account. These tasks demonstrate the similarities between Google Cloud IAM and AWS IAM when using command-line tools for access control. Both interfaces allow you to provision accounts/roles, create service accounts/roles, and switch users. You applied permissions to users within gcloud similarly to assigning roles and policies in AWS, and explored how Google Cloud projects function comparably to AWS accounts when managing access with Google Cloud IAM.

## Next steps / Learn more

- Check out the documentation for Cloud Identity and Access Management

## Google Cloud training and certification

...helps you make the most of Google Cloud technologies. Our classes include technical skills and best practices to help you get up to speed quickly and continue your learning journey. We offer fundamental to advanced level training, with on-demand, live, and virtual options to suit your busy schedule. Certifications help you validate and prove your skill and expertise in Google Cloud technologies.

**Manual Last Updated April 26, 2024**

**Lab Last Tested April 26, 2024**

Copyright 2024 Google LLC All rights reserved. Google and the Google logo are trademarks of Google LLC. All other company and product names may be trademarks of the respective companies with which they are associated.