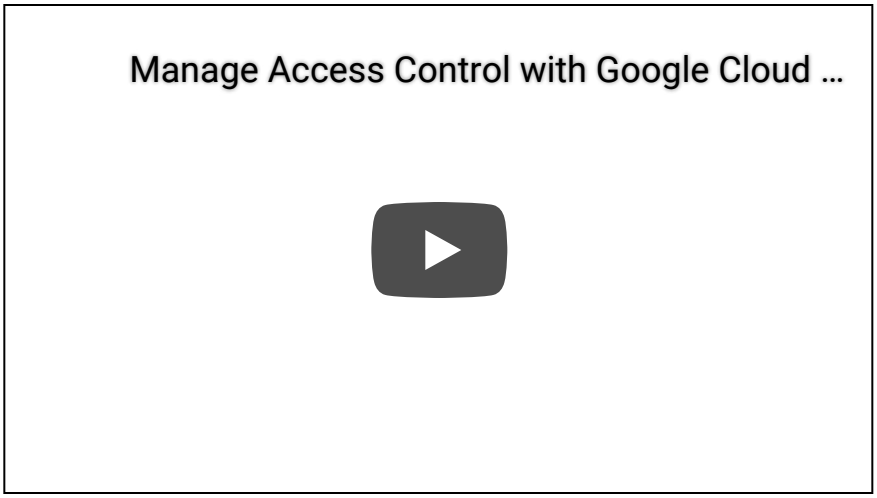
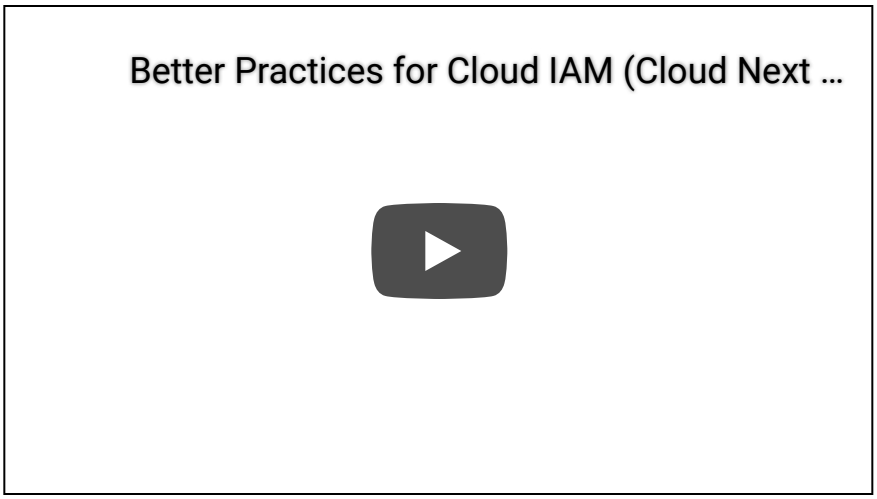
GCP Certification Series: Section 5: Configuring access and security, 5.1 Managing Identity and Access Management (IAM)

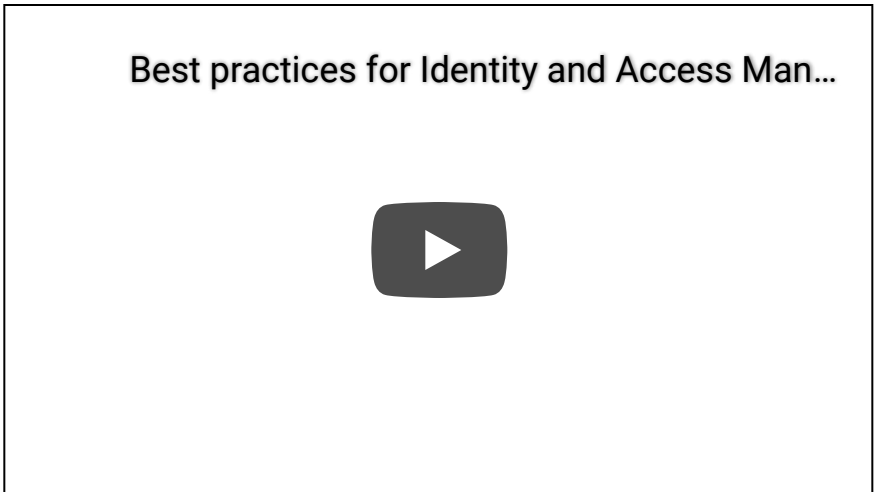
Prashanta Paudel

Nov 13, 2018 · 22 min read

This is the last section in GCP Certification series as well as in the course. This section mostly deals about how to manage access and security of the instances and products in Google Cloud Platform.







Identity and Access management is the framework for business and organization to facilitate digital identity. IAM is basically a security solution in general term.

IAM makes it possible for organizations to give proper access rights to the people according to their job role. This way no-one will have more access and things will work smoothly.

reference: https://searchsecurity.techtarget.com/definition/identity-access-management-IAM-system

Another aspect of IAM is service accounts management. Now with this feature, you can allow certain services to be allowed to operate in an instance by allowing through service account.

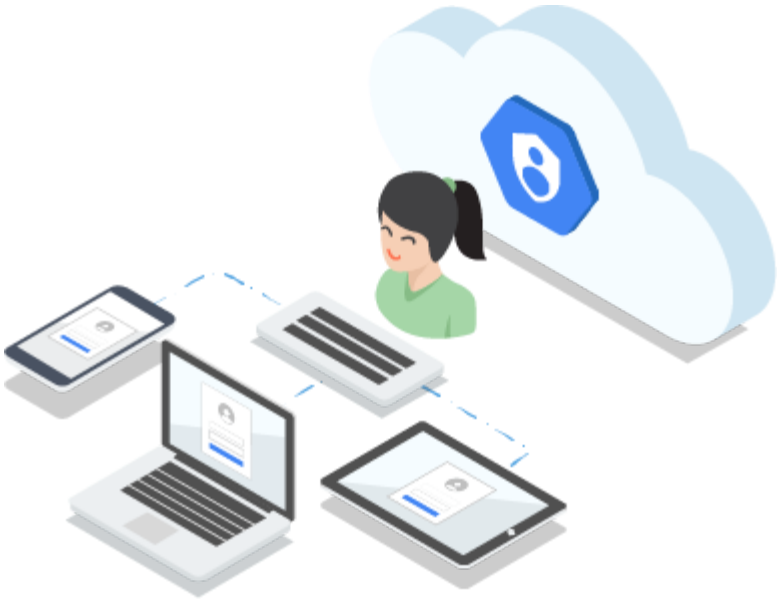
Cloud Identity and Access Management (Cloud IAM) enables you to create and manage permissions for Google Cloud Platform resources. Cloud IAM unifies access control for Cloud Platform services into a single system and presents a consistent set of operations.

Identity and management technologies include (but aren't limited to) password-management tools, provisioning software, security-policy enforcement applications, reporting and monitoring apps and identity repositories.

Enterprise-grade access control

Cloud Identity & Access Management (Cloud IAM) lets administrators authorize who can take action on specific resources, giving you full control and visibility to manage cloud resources centrally. For established enterprises with complex organizational structures, hundreds of workgroups, and potentially many more projects, Cloud

IAM provides a unified view into security policy across your entire organization, with built-in auditing to ease compliance processes.



Enterprise identity made easy

Leverage Cloud Identity, Google Cloud's built-in managed identity to easily create or sync user accounts across applications and projects. Cloud Identity makes it easy to provision and manage users and groups, set up single sign-on, and configure multi-factor authentication directly from the Google Admin Console. With Cloud Identity you get access to the GCP Organization, which enables you to centrally manage projects via the Cloud Resource Manager.

The right roles

Cloud IAM provides the right tools to manage resource permissions with minimum fuss and high automation. Map job functions within your company to groups and roles. Users get access only to what they need to get the job done, and admins can easily grant default permissions to entire groups of users.

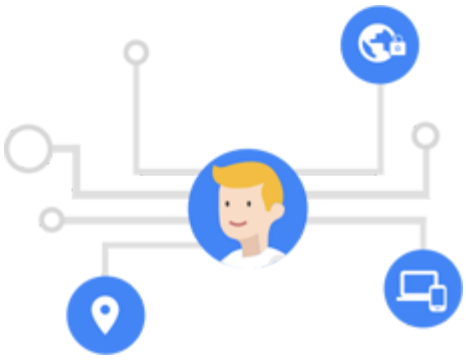
Granular resource control

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

Context-aware access

Create more granular access control policies to resources based on attributes like device security status, IP address, resource type, and date/time. These policies help ensure that the appropriate security

controls are in place when granting access to cloud resources. Sign up for the Cloud IAM conditions private beta here.



Simplicity first

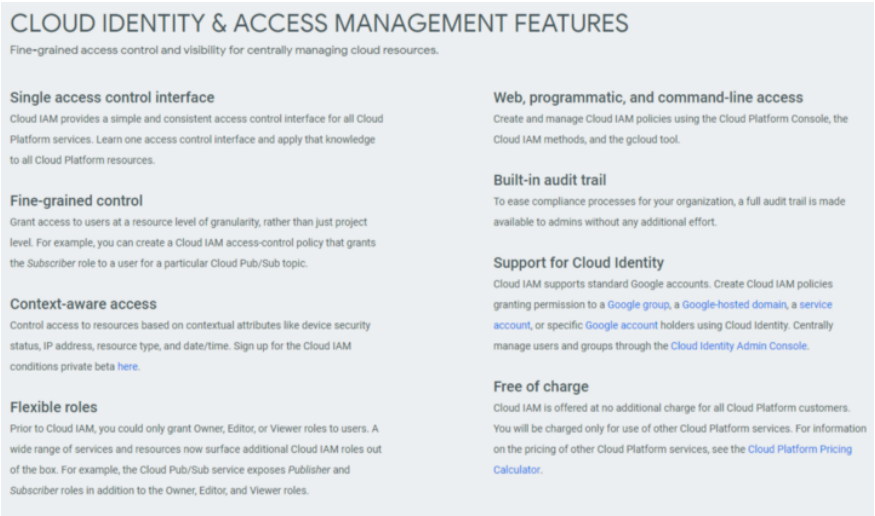
We recognize that an organization's internal structure and policies can get complex fast. Projects, workgroups, and managing who has the authorization to do what all change dynamically. Cloud IAM is designed with simplicity in mind: a clean, universal interface lets you manage access control across all Google Cloud Platform resources consistently. So you learn it once, then apply everywhere.

Built-in audit trail

A full audit trail history of permissions authorization, removal, and delegation gets surfaced automatically for your admins. Cloud IAM lets you focus on business policies around your resources and makes compliance easy.

Access control your way

Control resource permissions using a variety of options: graphically from the Cloud Platform console, programmatically via Cloud IAM methods, or using the gcloud command line interface.



Reference: https://cloud.google.com/iam/

Concepts related to identity

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

- Google account
- Service account

- Google group
- G Suite domain
- Cloud Identity domain

Google account

A Google account represents a developer, an administrator, or any other person who interacts with GCP. Any email address that is associated with a Google account can be an identity, including gmail.com or other domains. New users can sign up for a Google account by going to the Google account signup page.

Service account

A service account is an account that belongs to your application instead of to an individual end user. When you run code that is hosted on GCP, you specify the account that the code should run as. You can create as many service accounts as needed to represent the different logical components of your application. For more information about using a service account in your application, see Getting started with authentication.

Google group

A Google group is a named collection of Google accounts and service accounts. Every group has a unique email address that is associated with the group. You can find the email address that is associated with a Google group by clicking **About** on the homepage of any Google group. For more information about Google groups, see the Google groups homepage.

Google groups are a convenient way to apply an access policy to a collection of users. You can grant and change access controls for a whole group at once instead of granting or changing access controls one-at-a-time for individual users or service accounts. You can also easily add members to and remove members from a Google group instead of updating a Cloud IAM policy to add or remove users.

Note that Google groups don't have login credentials, and you cannot use Google groups to establish identity to make a request to access a

resource.

G Suite domain

G Suite domain represents a virtual group of all the Google accounts that have been created in an organization's G Suite account. G Suite domains represent your organization's Internet domain name (such as *example.com*), and when you add a user to your G Suite domain, a new Google account is created for the user inside this virtual group (such as *username@example.com*).

Like Google groups, G Suite domains cannot be used to establish identity, but they enable convenient permission management.

Cloud Identity domain

A Cloud Identity domain is like a G Suite domain because it represents a virtual group of all Google accounts in an organization. However, Cloud Identity domain users don't have access to G Suite applications and features. For more information, see About Cloud Identity.

allAuthenticatedUsers

This is a special identifier that represents anyone who is authenticated with a Google account or a service account. Users who are not authenticated, such as anonymous visitors, are not included.

allUsers

This is a special identifier that represents anyone who is on the internet, including authenticated and unauthenticated users. Note that some GCP APIs require authentication of any user accessing the service, and in those cases, allUsers will only imply authorization for all authenticated users.

==========

Concepts related to access management

When an authenticated member attempts to make a request, Cloud IAM makes an authorization decision about whether the member is allowed to perform the operation on a resource.

This section describes the entities and concepts involved in the authorization process.

Resource

You can grant access to users for a GCP resource. Some examples of resources are projects,Compute Engine instances, and Cloud Storage buckets.

Some services such as Cloud Pub/Sub and Compute Engine support granting Cloud IAM permissions at a granularity finer than the project-level. For example, you can grant the `pubsub.subscriber` role to a user for a particular Cloud Pub/Sub topic or you can grant the `compute.instanceAdmin` role to a user for a specific Compute Engine instance.

In other cases, you can grant Cloud IAM permissions at the project level. The permissions are then inherited by all resources within that project. For example, to grant access to a Cloud Storage bucket, you must grant the access to the project that contains the bucket. For information on what roles can be granted on which resources, see Understanding Roles.

Permissions

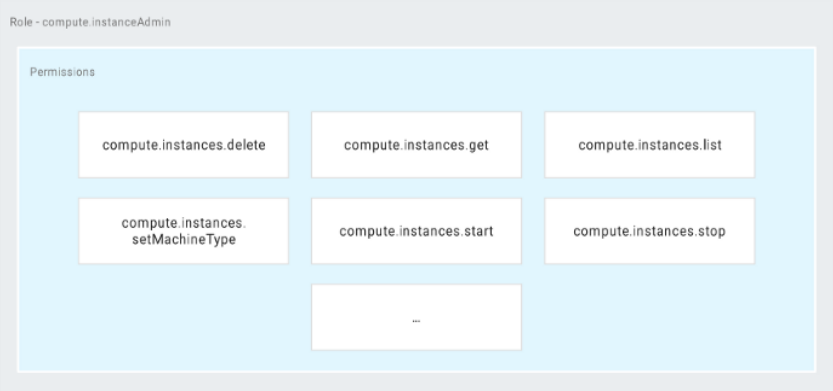
Permissions determine what operations are allowed on a resource. In the Cloud IAM world, permissions are represented in the form of `<service>.<resource>.<verb>` , for example `pubsub.subscriptions.consume` .

Permissions usually, but not always, correspond 1:1 with REST methods. That is, each GCP service has an associated set of permissions for each REST method that it exposes. The caller of that method needs those permissions to call that method. For example, the caller of `Publisher.Publish()` needs the `pubsub.topics.publish` permission.

You don’t assign permissions to users directly. Instead, you assign them a **Role** which contains one or more permissions.

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.



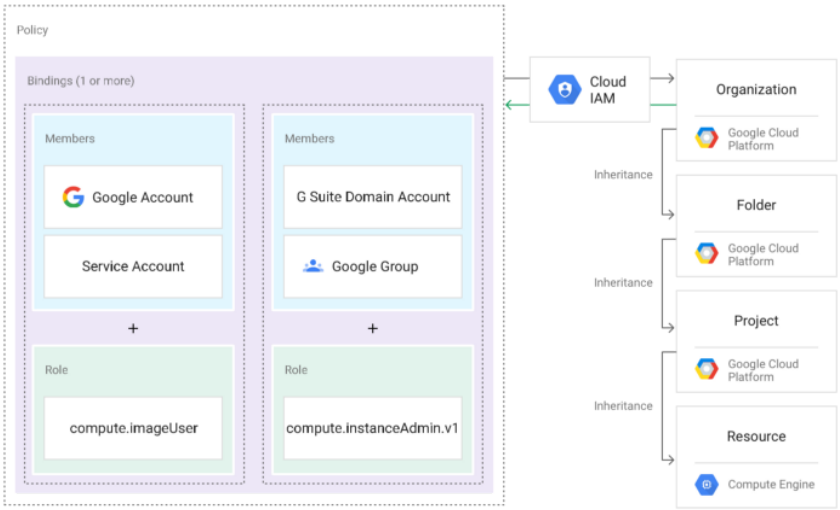
There are three kinds of roles in Cloud IAM:

- **Primitive roles**: The roles historically available in the Google Cloud Platform Console will continue to work. These are the **Owner**, **Editor**, and **Viewer** roles.
- **Predefined roles**: Predefined roles are the Cloud IAM roles that give finer-grained access control than the primitive roles. For example, the predefined role **Pub/Sub Publisher**(roles/pubsub.publisher) provides access to *only* publish messages to a Cloud Pub/Sub topic.
- **Custom roles**: Roles that you create to tailor permissions to the needs of your organization when predefined roles don't meet your needs.

To learn how to assign a role to the user, see Granting, Changing, and Revoking Access. For information about the available fine-grained Cloud IAM predefined roles, see Understanding Roles. For information about custom roles, see Understanding Custom Roles and Creating and Managing Custom Roles.

IAM Policy

You can grant roles to users by creating a *Cloud IAM policy*, which is a collection of statements that define who has what type of access. A policy is attached to a resource and is used to enforce access control whenever that resource is accessed.



A Cloud IAM policy is represented by the IAM `Policy` object. An IAM `Policy` object consists of a list of bindings. A `Binding` binds a list of `members` to a `role`.

`role` is the role you want to assign to the member. The `role` is specified in the form of `roles/<name of the role>`. For example, `roles/storage.objectAdmin`, `roles/storage.objectCreator`, and `roles/storage.objectViewer`.

`members` contains a list of one or more identities as described in the Concepts related to identity section above. Each member type is identified with a prefix, such as a Google account (`user:`), service account (`serviceAccount:`), Google group (`group:`), or a G Suite or Cloud identity domain (`domain:`). In the example snippet below, the `storage.objectAdmin` role is assigned to the following members using the appropriate prefix: `user:alice@example.com`, `serviceAccount:my-other-app@appspot.gserviceaccount.com`, `group:admins@example.com`, and `domain:google.com`. The `objectViewer` role is assigned to `user:bob@example.com`.

The following code snippet shows the structure of a Cloud IAM policy.

```
{
  "bindings": [
    {
      "role": "roles/storage.objectAdmin",
      "members": [
        "user:alice@example.com",
        "serviceAccount:my-other-
app@appspot.gserviceaccount.com",
        "group:admins@example.com",
        "domain:google.com" ]
    },
    {
      "role": "roles/storage.objectViewer",
      "members": ["user:bob@example.com"]
    }
    ]
}
```

Cloud IAM and Policy APIs

Cloud IAM provides a set of methods that you can use to create and manage access control policies on GCP resources. These methods are exposed by the services that support Cloud IAM. For example, the Cloud IAM methods are exposed by the Resource Manager, Cloud Pub/Sub, and Cloud Genomics APIs, just to name a few.

The Cloud IAM methods are:

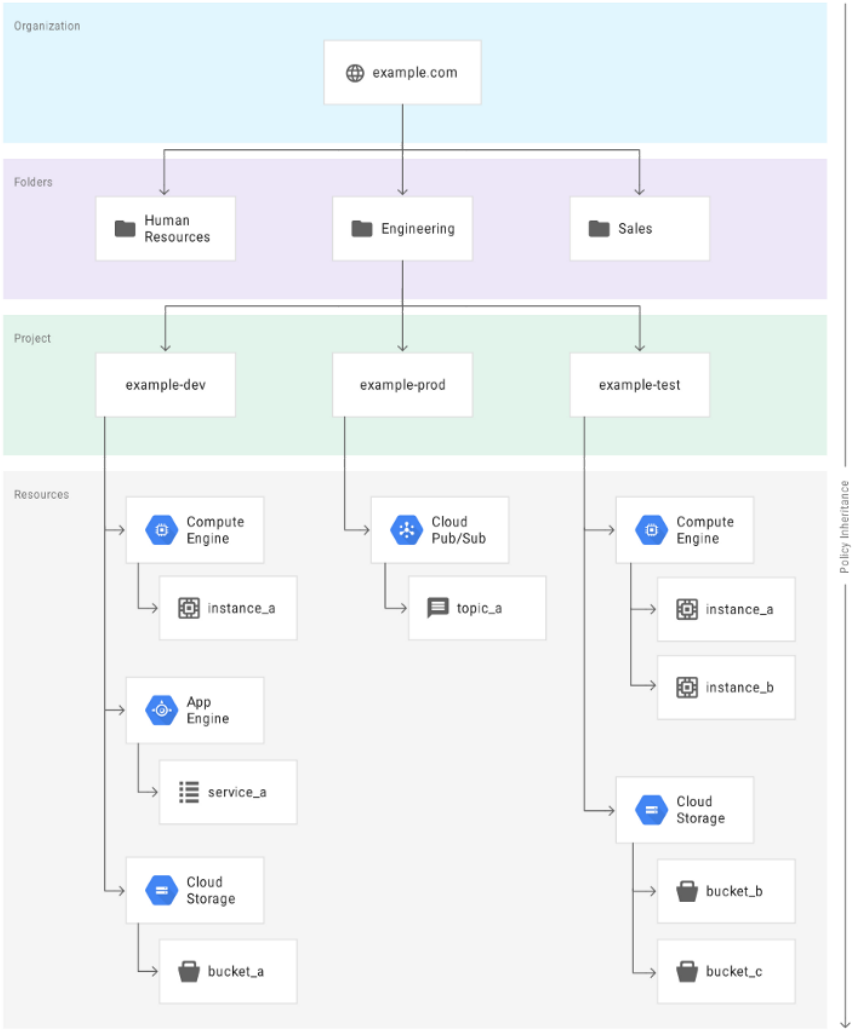
- `setIamPolicy()` : Allows you to set policies on your resources.
- `getIamPolicy()` : Allows you to get a policy that was previously set.
- `testIamPermissions()` : Allows you to test whether the caller has the specified permissions for a resource.

You can find the API reference topics for these methods in the documentation for each service that supports Cloud IAM.

Policy hierarchy

GCP 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 descendants of projects. Each resource has exactly one parent. See the Resource Manager Resource Hierarchy topic for more information.

The following diagram is an example of a GCP resource hierarchy:



You can set a Cloud IAM policy at any level in the resource hierarchy: the organization level, the folder level, the project level, or the resource level. Resources inherit the policies of the parent resource. If you set a policy at the organization level, it is automatically inherited by all its children projects, and if you set a policy at the project level, it's 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 higher up in the hierarchy.

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

For example in the diagram above, topic_a is a Cloud Pub/Sub resource that lives under the project example-prod. If you grant the Editor role to micah@gmail.com for example-prod and grant the Publisher role to song@gmail.com for topic_a, you effectively grant the Editor role for topic_a to micah@gmail.com and the Publisher role to song@gmail.com.

The Cloud IAM policy hierarchy follows the same path as the GCP resource hierarchy. If you change the resource hierarchy, the policy hierarchy changes as well. 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 a higher level. 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, then the user still has the Editor role grant for the child resource.

Cloud IAM support for GCP services

With Cloud IAM, every API method across all GCP services is checked to make sure that the account making the API request has the appropriate permission to use the resource.

Prior to Cloud IAM, you could only grant Owner, Editor, or Viewer roles. These roles give very broad access on a project and did not allow fine-grained separation of duties. GCP services now offer additional predefined roles that give finer-grained access control than the Owner, Editor, and Viewer roles. For example, Compute Engine offers roles such as *Instance Admin* and *Network Admin* and App Engine offers roles such as *App Admin* and *Service Admin*. These predefined roles are available in addition to the legacy Owner, Editor, and Viewer roles.

If you need even more control over permissions, consider creating a Custom Role.

The following products offer Cloud IAM predefined roles:

- Google Cloud Platform project
- GCP Organization

- Compute Engine
- Cloud Source Repositories
- App Engine
- Cloud Storage
- BigQuery
- Cloud Bigtable
- IAM for Cloud SQL
- Stackdriver Debugger
- Cloud Deployment Manager
- Cloud Genomics
- Cloud Key Management Service
- Cloud Pub/Sub
- Cloud Machine Learning Engine
- Cloud Spanner
- Stackdriver Logging
- Cloud IAM for Stackdriver Monitoring
- Cloud Dataflow
- Cloud IAM for Cloud Datastore
- Cloud IAM for Cloud Dataproc
- Cloud IAM for Google Kubernetes Engine
- Cloud IAM for Cloud DNS
- Cloud IAM for Stackdriver Trace
- Cloud IAM for Cloud Billing API
- Cloud IAM for Service Management

For a complete list of predefined roles, see Understanding Roles.

You can grant users certain roles to access resources at a granularity *finer than the project level*. For example, you can create a Cloud IAM access control policy that grants a user the *Subscriber* role for a particular Cloud Pub/Sub topic. For information on what roles can be granted on which resources, see Understanding Roles.

———————————————————————————————————

Service Accounts

This page explains service accounts, types of service accounts, and the IAM roles that are available to service accounts.

Before you begin

- Understand the basic concepts of Cloud IAM.

What are 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.

For example, a Compute Engine VM may run as a service account, and that account can be given permissions to access the resources it needs. This way the service account is the identity of the service, and the service account's permissions control which resources the service can access.

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

Service account keys

Each service account is associated with a key pair, which is managed by Google Cloud Platform (GCP). It is used for service-to-service authentication within GCP. These keys are rotated automatically by Google, and are used for signing for a maximum of two weeks.

You may optionally create one or more external key pairs for use from outside GCP (for example, for use with Application Default Credentials). When you create a new key pair, you download the private key (which is not retained by Google). With external keys, you are responsible for security of the private key and other management operations such as key rotation. External keys can be managed by the IAM API, `gcloud` command-line tool, or the Service Accounts page in the Google Cloud Platform Console. You can create up to 10 service account keys per service account to facilitate key rotation.

Types of service accounts

User-managed service accounts

When you create a new Cloud project using GCP Console and if Compute Engine API is enabled for your project, a Compute Engine Service account is created for you by default. It is identifiable using the email:

```
PROJECT_NUMBER-compute@developer.gserviceaccount.com
```

If your project contains an App Engine application, the default App Engine service account is created in your project by default. It is identifiable using the email:

```
PROJECT_ID@appspot.gserviceaccount.com
```

If you create a service account in your project, you'll name the service account and it will be assigned an email with the following format:

```
SERVICE_ACCOUNT_NAME@PROJECT_ID.iam.gserviceaccount.com
```

You can create up to 100 service accounts per project (including the default Compute Engine service account and the App Engine service account) using the IAM API, the GCP console, or the `gcloud` command-line tool. These default service accounts and the service accounts you explicitly create are the user-managed service accounts.

Caution: The exact behavior of when the default service accounts are created and how they show up in your project might change in the future since they are designed to be used by Compute Engine and App Engine, so it is recommended that you don't rely on the existence of these default accounts for your use. It is recommended that you create additional service accounts explicitly using the IAM API, GCP Console or the `gcloud` command-line tool for your long-term use.

Google-managed service accounts

In addition to the user-managed service accounts, you might see some additional service accounts in your project's IAM policy or in GCP Console. These service accounts are created and owned by Google. These accounts represent different Google services and each account is automatically granted IAM roles to access your GCP project.

Google APIs service account

An example of a Google-managed service account is a Google API service account identifiable using the email:

```
PROJECT_NUMBER@cloudservices.gserviceaccount.com
```

This service account is designed specifically to run internal Google processes on your behalf and is not listed in the **Service Accounts** section of GCP Console. By default, the account is automatically granted the project editor role on the project and is listed in the **IAM** section of GCP Console. This service account is deleted only when the project is deleted. Google services rely on the account having access to your project, so you should not remove or change the service account's role on your project.

Service account permissions

In addition to being an identity, a service account is a resource which has IAM policies attached to it. These policies determine who can use the service account.

For instance, Alice can have the editor role on a service account and Bob can have viewer role on a service account. This is just like granting roles for any other GCP resource.

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 Cloud Storage bucket, you grant the service account (that your automation uses) the permissions to read the Cloud Storage bucket. In this case, the service account is the identity that you are granting permissions for another resource (the Cloud Storage bucket).

The Service Account User role

You can grant the `iam.serviceAccountUser` role at the project level for all service accounts in the project, or at the service account level.

- Granting the `iam.serviceAccountUser` role to a user for a project gives the user access to all service accounts in the project, including service accounts that may be created in the future.
- Granting the `iam.serviceAccountUser` role to a user for a specific service account gives a user access to the service account.

If you grant a user the `compute.instanceAdmin` role with the `iam.serviceAccountUser` role, they can create and manage Compute Engine instances that use a service account.

After you grant IAM roles to service accounts, you can assign the service account to one or more new virtual machine instances. For instructions on how to do this, see Setting up a new instance to run as a service account.

Users who are serviceAccountUsers can use the service account to indirectly access all the resources to which the service account has access. For example, a user who is a serviceAccountUser can start an instance using the service account. They can then use the instance to access anything the service account identity has access to. However, the serviceAccountUser role doesn't allow a user to directly use the service account's roles. Therefore, be cautious when granting the `iam.serviceAccountUser` role to a user.

Service accounts represent your service-level security. The security of the service is determined by the people who have IAM roles to manage and use the service accounts, and people who hold private external keys for those service accounts. Best practices to ensure security include the following:

- Use the IAM API to audit the service accounts, the keys, and the policies on those service accounts.
- If your service accounts don't need external keys, delete them.
- If users don't need permission to manage or use service accounts, then remove them from the IAM Policy.

To learn more about best practices, see Understanding service accounts.

The Service Account Token Creator role

This role enables impersonation of service accounts to create OAuth2 access tokens, sign blobs, or sign JWTs.

The Service Account Actor role

This role has been deprecated. If you need to run operations as the service account, use the Service Account User role. To effectively provide the same permissions as Service Account Actor, you should also grant Service Account Token Creator.

Access scopes

Access scopes are the legacy method of specifying permissions for your VM. Before the existence of IAM roles, access scopes were the only mechanism for granting permissions to service accounts. Although they are not the primary way of granting permissions now, you must still set access scopes when configuring an instance to run as a service account. For information on access scopes seeGoogle Compute Engine documentation

Short-Lived Service Account Credentials

You can create short-lived credentials that allow you to assume the identity of a GCP service account. These credentials can be used to authenticate calls to Google Cloud Platform APIs or other non-Google APIs.

The most common use case for these credentials is to temporarily delegate access to GCP resources across different projects, organizations, or accounts. For example, instead of providing an external caller with the permanent credentials of a highly-privileged service account, temporary emergency access can be granted instead. Alternatively, a designated service account with restricted permissions can be impersonated by an external caller without requiring a more highly-privileged service account's credentials.

For more information, see Creating Short-Lived Service Account Credentials.

Application Default Credentials

Application default credentials are a mechanism to make it easy to use service accounts when operating inside and outside GCP, as well as across multiple GCP projects. The most common use case is testing code on a local machine, and then moving to a development project in GCP, and then moving to a production project in GCP. Using Application Default Credentials ensures that the service account works seamlessly; that is, that it uses a locally-stored service account key when testing on your local machine but uses the project's default Compute Engine service account when running on Compute Engine. For more information, see Application Default Credentials.

==========

Using IAM Securely

Introduction

This page recommends security best practices that you should keep in mind when using Cloud IAM.

This page is designed for users who are proficient with Cloud IAM. If you are just starting out with IAM, these instructions will not teach you how to use it; instead, new users should start with theCloud IAM Quickstart.

Least privilege

❑Predefined roles provide more granular access than the primitive roles. Grant predefined roles to identities when possible, so you only give the least amount of access necessary to access your resources.

❑Grant primitive roles in the following cases:

- when the Cloud Platform service does not provide a predefined role. See the predefined roles table for a list of all available predefined roles.
- when you want to grant broader permissions for a project. This often happens when you're granting permissions in development or test environments.
- when you need to allow a member to modify permissions for a project, you'll want to grant them the owner role because only owners have the permission to grant access to other users for for projects.
- when you work in a small team where the team members don't need granular permissions.

❑Treat each component of your application as a separate trust boundary. If you have multiple services that require different permissions, create a separate service account for each of the services so that they can be permissioned differently.

❑Remember that a policy set on a child resource cannot restrict access granted to its parent. Check the policy granted on every resource and make sure you understand the hierarchical inheritance.

❑Grant roles at the smallest scope needed. For example, if a user only needs access to publish Pub/Sub topic, grant the Publisher role to the user for that topic.

❑Restrict who can act as service accounts. Users who are granted the Service Account Actor role for a service account can access all the resources for which the service account has access. Therefore be cautious when granting the Service Account Actor role to a user.

❑Restrict who has access to create and manage service accounts in your project.

❑Granting owner role to a member will allow them to modify the IAM policy. Therefore grant the owner role only if the member has a legitimate purpose to manage the IAM policy. This is because as your policy contains sensitive access control data and having a minimal set of users manage it will simplify any auditing that you may have to do.

Service accounts and service account keys

❑Rotate your service account keys using the IAM service account API. You can rotate a key by creating a new key, switching applications to use the new key and then deleting old key. Use the `serviceAccount.keys.create()` method and `serviceAccount.keys.delete()` method together to automate the rotation.

❑Implement processes to manage user-managed service account keys.

❑Be careful not to confuse encryption keys with service account keys. Encryption keys are typically used to encrypt data and service account keys are used for secure access to Google Cloud Platform APIs.

❑Do not delete service accounts that are in use by running instances. This could result in all or parts of your application to fail if you have not transitioned to using an alternative service account first.

❑Use the display name of a service account to keep track of what they are used for and what permissions they should have.

❑Don’t check in the service account keys into the source code or leave them in the Downloads directory.

Auditing

❑Use Cloud audit logs to regularly audit changes to your IAM policy.

❑Exporting audit logs to Google Cloud Storage to store your logs for long periods of time.

❑Audit who has the ability to change your IAM policies on your projects.

❑Restrict access to logs using Cloud logging roles.

❑Apply the same access policies to the Cloud Platform resource that you use to export logs as applied to the logs viewer.

❑Use Cloud audit logs to regularly audit access to service account keys.

Policy management

❑Set organization-level IAM policies to grant access to all projects in your organization.

❑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.

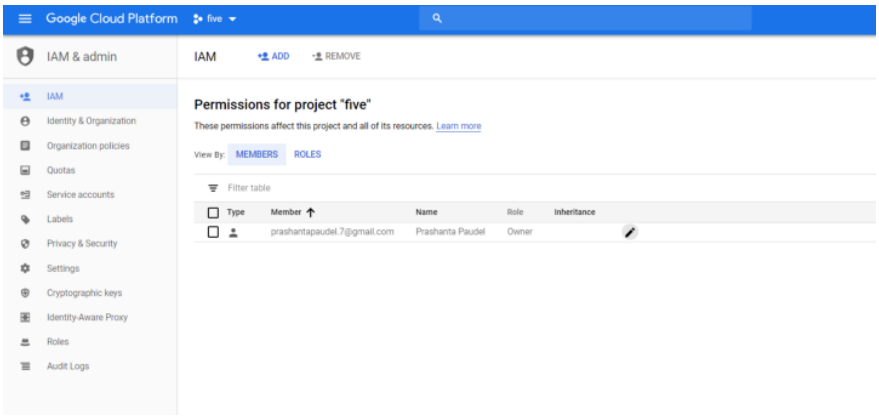
❑If you need to grant multiple roles to allow a particular task, create a Google group, grant the roles to that group, and then add users to that group.

==========

Now lets get back to main tasks in course

Viewing account IAM assignments

To view IAM assignments for a particular user account, go to IAM in the main page and then check the Dashboard



IAM

> *from Shell*

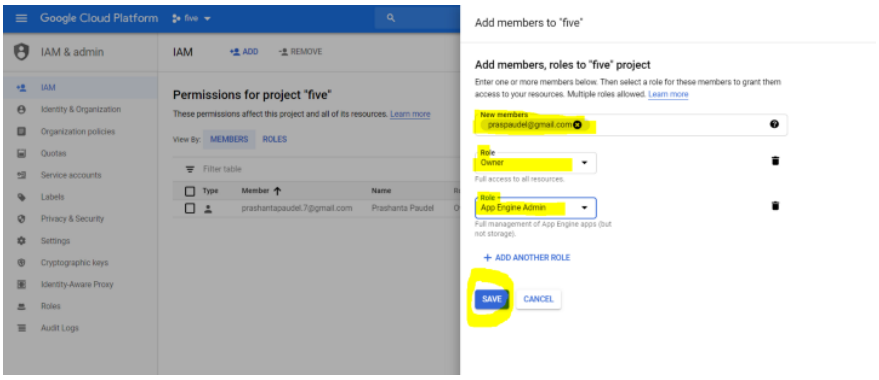
$ gcloud iam roles list

==========

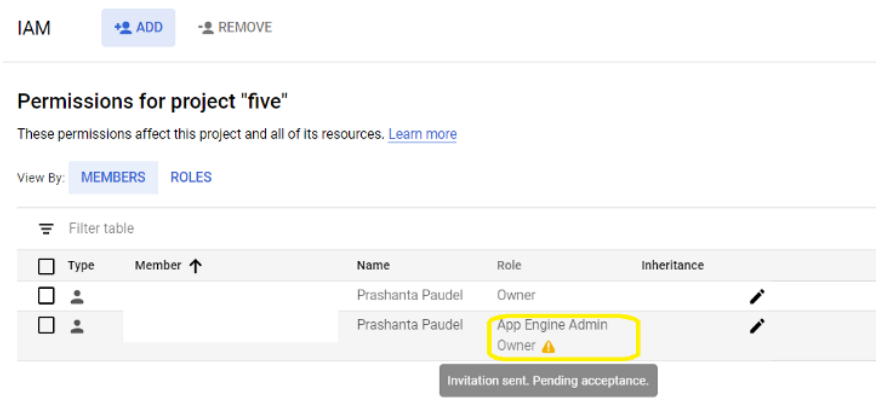
Assigning IAM roles to accounts or Google Groups

First, go to IAM and Services

Then click on ADD and follow the prompt.

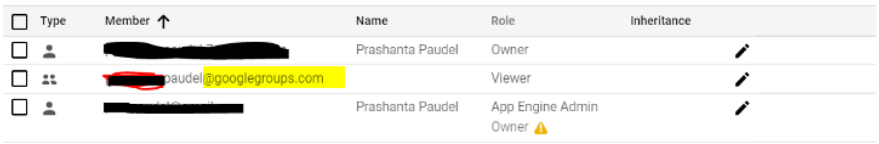


Assign new user roles



Needs acceptance of the invitation.

Adding google group to IAM



add group

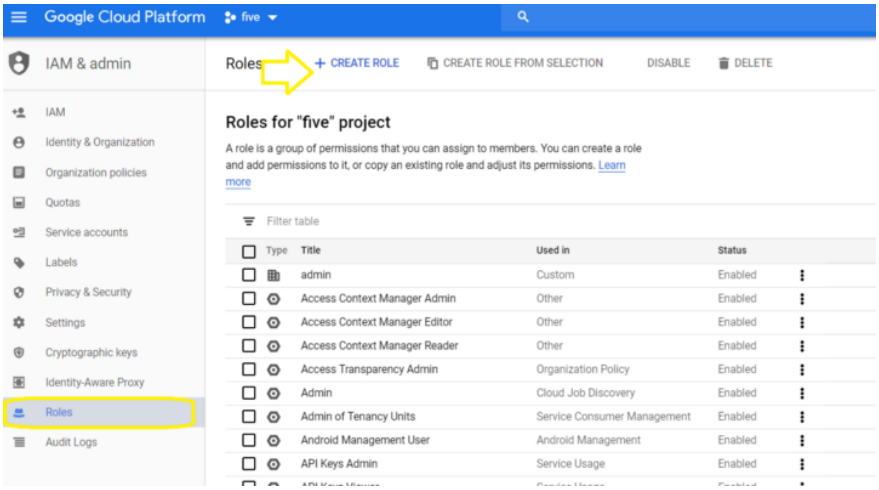
> *In shell*

```
$ gcloud projects add-iam-policy-binding five-222414  --
member user:praspaudel@gmail.com --role roles/viewer
bindings:
- members:
  - user:praspaudel@gmail.com
  role: roles/appengine.appAdmin
- members:
  - group:prashantapaudel@googlegroups.com
  role: roles/bigquery.admin
- members:
  - user:prashantapaudel.7@gmail.com
  - user:praspaudel@gmail.com
  role: roles/owner
- members:
  - group:prashantapaudel@googlegroups.com
  - user:praspaudel@gmail.com
  role: roles/viewer
etag: BwV6jYg5HT4=
version: 1
prashantapaudel_7@cloudshell:~ (five-222414)$
```

———————————————————————————————————————

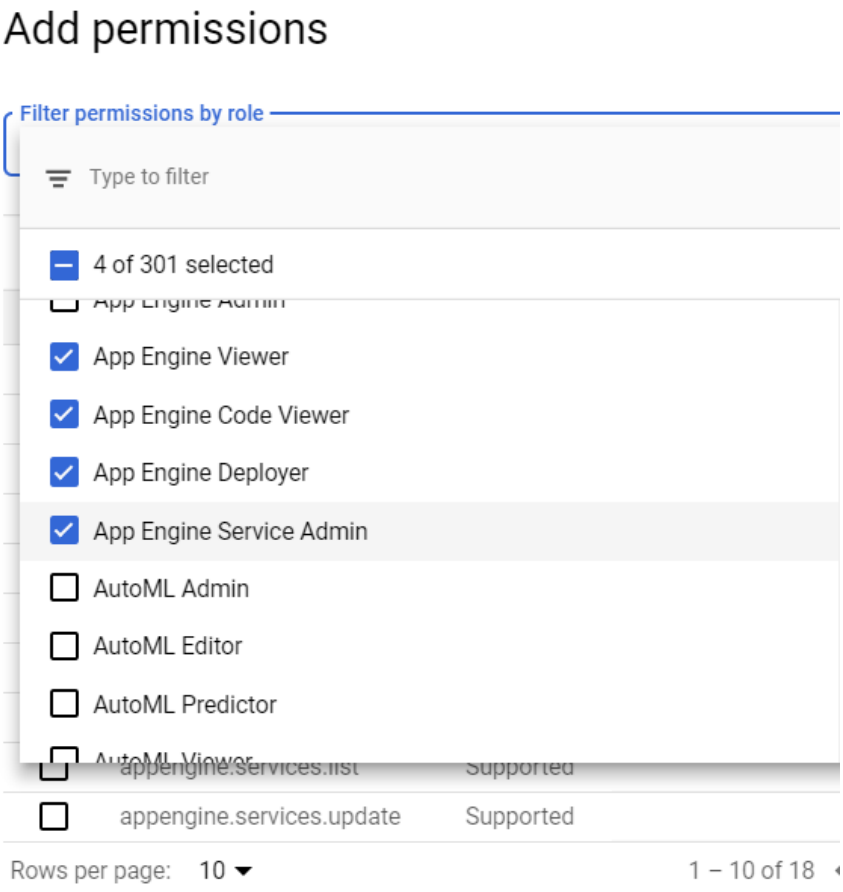
Defining custom IAM roles

First go to IAM and Services, then to Roles as shown below

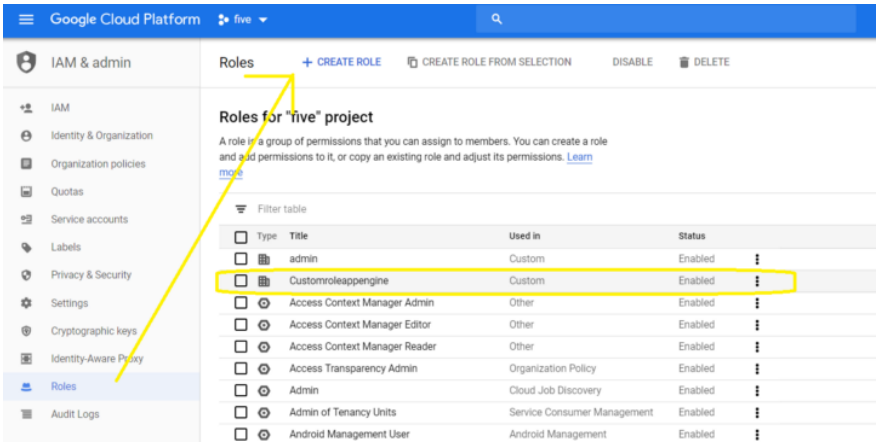


Roles

Now, I created a new role called *customrole* and added permission to operate in App engine only.

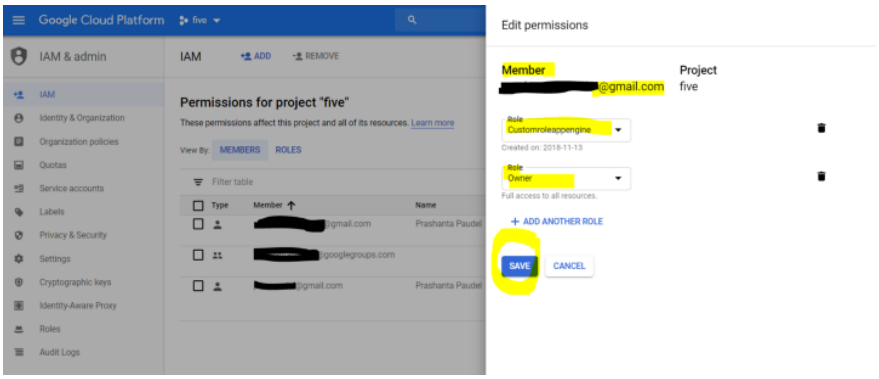


permissions



new created custom role

Now let's try to apply to a new user



new role to a new user

After implementing the role, it is seen in the IAM page which can be changed or revoked anytime.

This can be done in Shell also but will be bit complex.