# IAM overview

This page describes how Google Cloud's Identity and Access Management (IAM) system works and how you can use it to manage access in Google Cloud.

IAM lets you grant granular access to specific Google Cloud resources and helps prevent access to other resources. IAM lets you adopt the security principle of least privilege, which states that nobody should have more permissions than they actually need.

## How IAM works

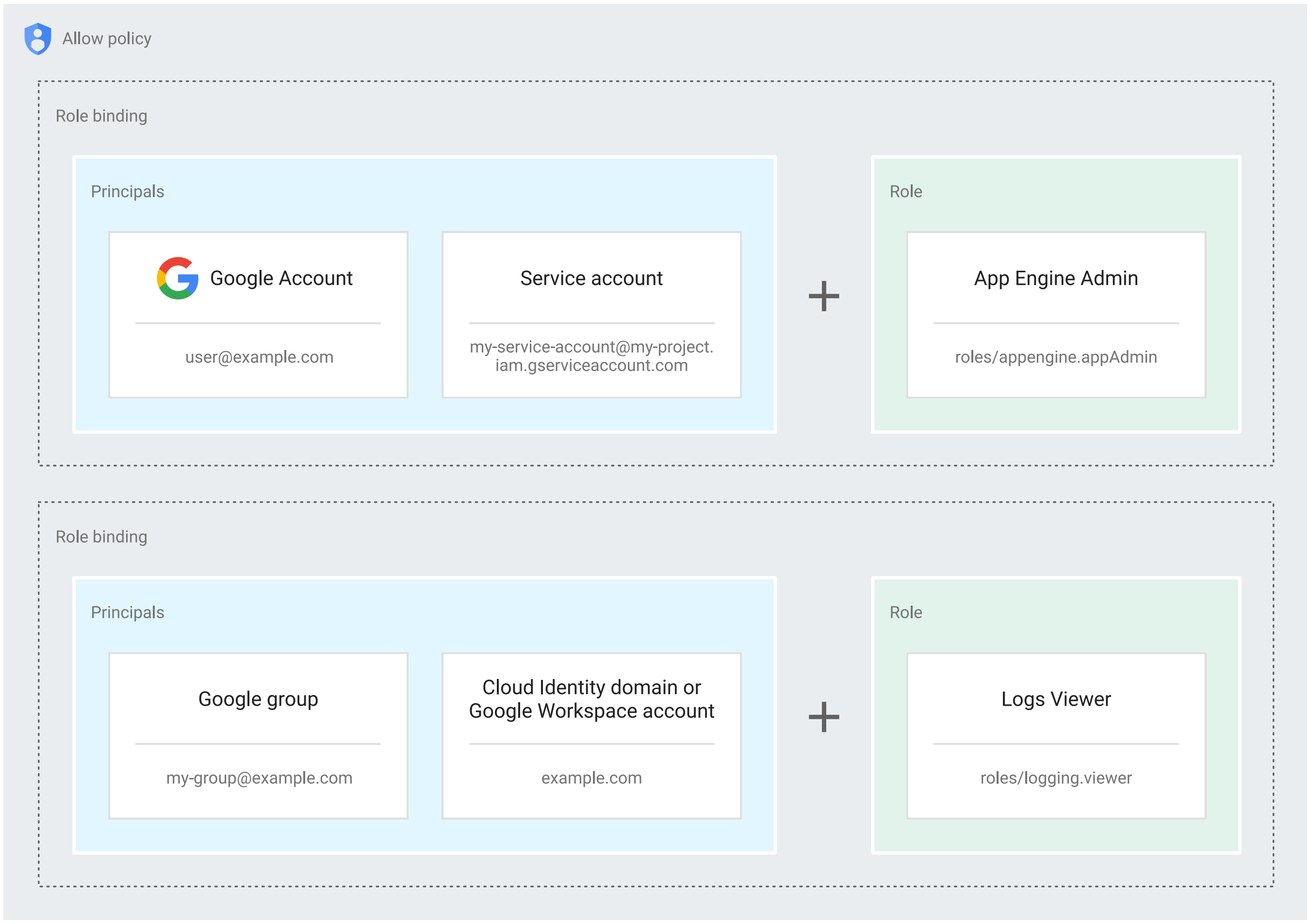
With IAM, you manage access control by defining *who* (identity) has *what access* (role) for *which resource*. For example, Compute Engine virtual machine instances, Google Kubernetes Engine (GKE) clusters, and Cloud Storage buckets are all Google Cloud resources. The organizations, folders, and projects that you use to organize your resources are also resources.

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.)

An *allow policy*, also known as an *IAM policy*, defines and enforces what roles are granted to which principals. Each allow policy is attached to a resource. When an authenticated principal attempts to access a resource, IAM checks the resource's allow policy to determine whether the action is permitted.

**Note:** You can also use deny policies to prevent principals from using specific IAM permissions. For more information, see Deny policies (/iam/docs/deny-overview).

The following diagram illustrates permission management in IAM.



This model for access management has three main parts:

- **Principal**. A *principal* can be a Google Account (for end users), a service account (for applications and compute workloads), a Google group, or a Google Workspace account or Cloud Identity domain that can access a resource. Each principal has its own identifier, which is typically an email address.
- **Role**. A *role* is a collection of permissions. Permissions determine what operations are allowed on a resource. When you grant a role to a principal, you grant all the permissions that the role contains.
- **Policy**. The *allow policy* is a collection of role bindings that bind one or more principals to individual roles. When you want to define who (principal) has what type of access (role) on a resource, you create an allow policy and attach it to the resource.

In the preceding diagram, for example, the allow policy binds principals, such as `user@example.com`, to roles, such as the App Engine Admin role (`roles/appengine.appAdmin`). If the allow policy is attached to a project, the principals gain the specified roles within the project.

The rest of this page describes these concepts in greater detail.

## Concepts related to identity

In 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

### Google Account

A Google Account represents a developer, an administrator, or any other person who interacts with Google Cloud. Any email address that's 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 (https://accounts.google.com/signup).

### Service account

A service account is an account for an application or compute workload instead of an individual end user. When you run code that's hosted on Google Cloud, the code runs as the account you specify. 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 (/docs/authentication/getting-started).

**Note:** If you use Google Kubernetes Engine (GKE), you can also grant roles to Kubernetes service accounts (/kubernetes-engine/docs/how-to/kubernetes-service-accounts), which differ from IAM service accounts.

## Google group

A Google group is a named collection of Google Accounts and service accounts. Every Google group has a unique email address that's associated with the group. You can find the email address that's 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 (https://groups.google.com/) homepage.

Google Groups are a convenient way to apply access controls 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 principals to and remove principals from a Google group instead of updating an allow policy to add or remove users.

Google Groups don't have login credentials, and you cannot use Google Groups to establish identity to make a request to access a resource.

## Google Workspace account

A Google Workspace account represents a virtual group of all of the Google Accounts that it contains. Google Workspace accounts are associated with your organization's internet domain name, such as `example.com`. When you create a Google Account for a new user, such as `username@example.com`, that Google Account is added to the virtual group for your Google Workspace account.

Like Google Groups, Google Workspace accounts cannot be used to establish identity, but they enable convenient permission management.

## Cloud Identity domain

A Cloud Identity domain is like a Google Workspace account, because it represents a virtual group of all Google Accounts in an organization. However, Cloud Identity domain users don't have access to Google Workspace applications and features. For more information, see About Cloud Identity (https://support.google.com/a/answer/7319251).

## All authenticated users

The value `allAuthenticatedUsers` is a special identifier that represents all service accounts and all users on the internet who have authenticated with a Google Account. This identifier includes accounts that aren't connected to a Google Workspace account or Cloud Identity domain, such as personal Gmail accounts. Users who aren't authenticated, such as anonymous visitors, aren't included.

**Note:** Consider using `allUsers` (#all-users), as described on this page, rather than `allAuthenticatedUsers`. In many cases, granting access to all users is no more of a security risk than granting access only to authenticated users.

This principal type doesn't include identities that come from external identity providers (IdPs). If you use workforce identity federation (/iam/docs/workforce-identity-federation) or workload identity federation (/iam/docs/workload-identity-federation), don't use `allAuthenticatedUsers`. Instead, use one of the following:

- To include users from all IdPs, use `allUsers`.
- To include users from specific external IdPs, use the identifier for all identities in a workforce identity pool (/iam/docs/workforce-identity-federation#representing-workforce-users) or all identities in a workload identity pool (/iam/docs/workload-identity-federation#impersonation).

Some resource types do not support this principal type.

### All users

The value `allUsers` is a special identifier that represents anyone who is on the internet, including authenticated and unauthenticated users.

Some resource types do not support this principal type.

**Note:** Some Google Cloud services require authentication before a user can access the service. For these services, `allUsers` includes only authenticated users.

## Concepts related to access management

When an authenticated principal attempts to access a resource, IAM checks the resource's allow policy to determine whether the action is allowed.

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

## Resource

If a user needs access to a specific Google Cloud resource, you can grant the user a role for that resource. Some examples of resources are projects (/resource-manager/docs/cloud-platform-resource-hierarchy#projects), Compute Engine instances (/compute/docs/instances), and Cloud Storage buckets (/storage/docs/buckets).

Some services support granting IAM permissions at a granularity finer than the project level. For example, you can grant the Storage Admin role (`roles/storage.admin`) to a user for a particular Cloud Storage bucket, or you can grant the Compute Instance Admin role (`roles/compute.instanceAdmin`) to a user for a specific Compute Engine instance.

In other cases, you can grant IAM permissions at the project level. The permissions are then inherited by all resources within that project. For example, to grant access to all Cloud Storage buckets in a project, grant access to the project instead of each individual bucket. Or to grant access to all Compute Engine instances in a project, grant access to the project rather than each individual instance.

For information on what roles can be granted on which resources, see Understanding roles (/iam/docs/understanding-roles#predefined_roles) and refer to the **Lowest Resource** column for a given role.

## Permissions

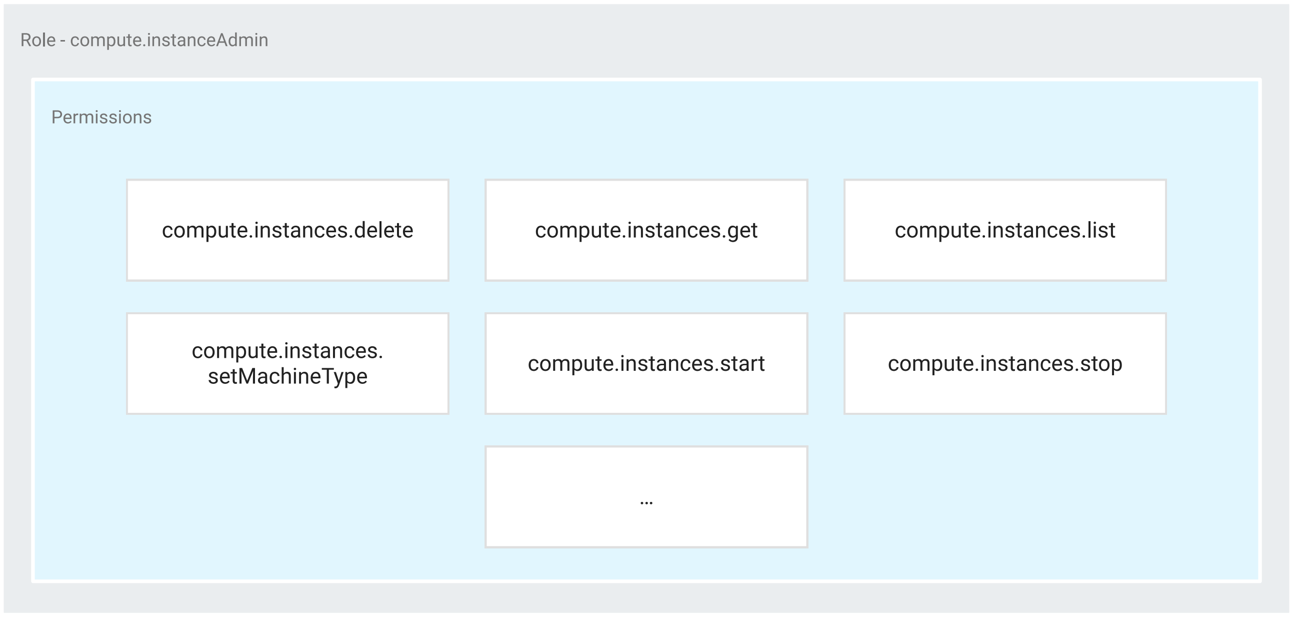
Permissions determine what operations are allowed on a resource. In the IAM world, permissions are represented in the form of ***`service.resource.verb`***, for example, `pubsub.subscriptions.consume`.

Permissions often correspond one-to-one with REST API methods. That is, each Google Cloud service has an associated set of permissions for each REST API method that it exposes. The caller of that method needs those permissions to call that method. For example, if you use Pub/Sub, and you need to call the `topics.publish()` method, you must have the `pubsub.topics.publish` permission for that topic.

You don't grant permissions to users directly. Instead, you identify *roles* that contain the appropriate permissions, and then grant those roles to the user. For a list of all available permissions and the roles that contain them, see the permissions reference (/iam/docs/permissions-reference).

## Roles

A role is a collection of permissions. You cannot grant 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 several kinds of roles in IAM:

- **Basic roles**: Roles historically available in the Google Cloud console. These roles are Owner, Editor, and Viewer.

**Caution:** Basic roles include thousands of permissions across all Google Cloud services. In production environments, do not grant basic roles unless there is no alternative. Instead, grant the most limited predefined roles (/iam/docs/understanding-roles#predefined_roles) or custom roles (/iam/docs/understanding-custom-roles) that meet your needs.

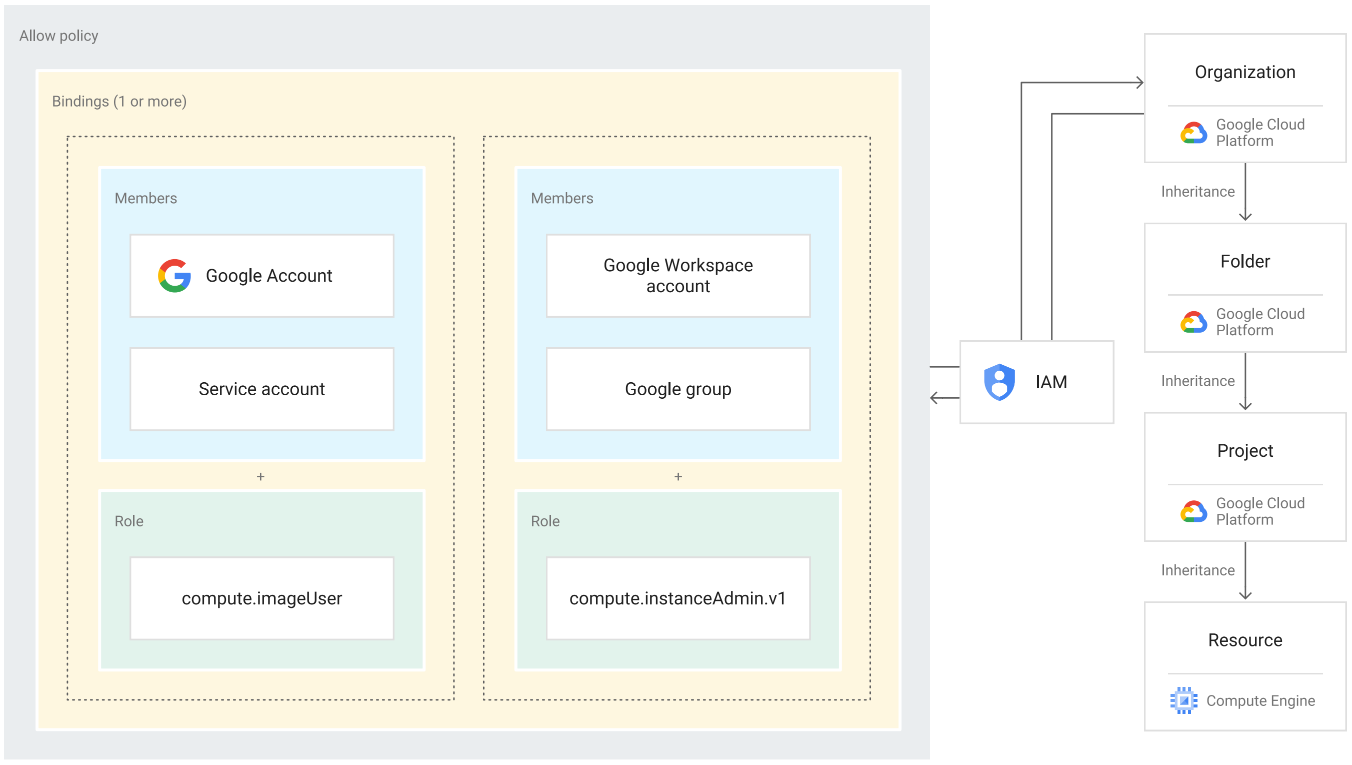
- **Predefined roles**: Roles that give finer-grained access control than the basic roles. For example, the predefined role Pub/Sub Publisher (`roles/pubsub.publisher`) provides access to *only* publish messages to a Pub/Sub topic.
- **Custom roles**: Roles that you create to tailor permissions to the needs of your organization when predefined roles don't meet your needs.

For more information about roles, see the following resources:

- To learn how to grant a role to a user, see Granting, changing, and revoking access (/iam/docs/granting-changing-revoking-access).
- For information about available IAM predefined roles, see Understanding roles (/iam/docs/understanding-roles).
- For information about custom roles, see Understanding custom roles (/iam/docs/understanding-custom-roles) and Creating and managing custom roles (/iam/docs/creating-custom-roles).

## Allow policy

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



An allow policy consists of a list of role bindings. A role binding binds a list of principals to a role.

- `role`: The role you want to grant to the principal. `role` is specified in the form of `roles/`*`service`*`.`*`roleName`*. For example, Cloud Storage provides the roles `roles/storage.objectAdmin`, `roles/storage.objectCreator`, and `roles/storage.objectViewer`, among others.
- `members`: A list of one or more principals as described in the Concepts related to identity (#concepts_related_identity) section in this document. Each principal type is identified with a prefix, such as a Google Account (`user:`), service account (`serviceAccount:`), Google group (`group:`), or a Google Workspace account or Cloud Identity domain (`domain:`). In the following example code snippet, the `storage.objectAdmin` role is granted to the following principals by using the appropriate prefix: `user:ali@example.com`, `serviceAccount:my-other-app@appspot.gserviceaccount.com`, `group:admins@example.com`, and `domain:google.com`. The `objectViewer` role is granted to `user:maria@example.com`.

The following code snippet shows the structure of an allow policy.

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

#### IAM and policy APIs

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

The IAM methods are:

- **`setIamPolicy()`**: Sets allow policies on your resources.
- **`getIamPolicy()`**: Gets an allow policy that was previously set.
- **`testIamPermissions()`**: Tests 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 IAM.

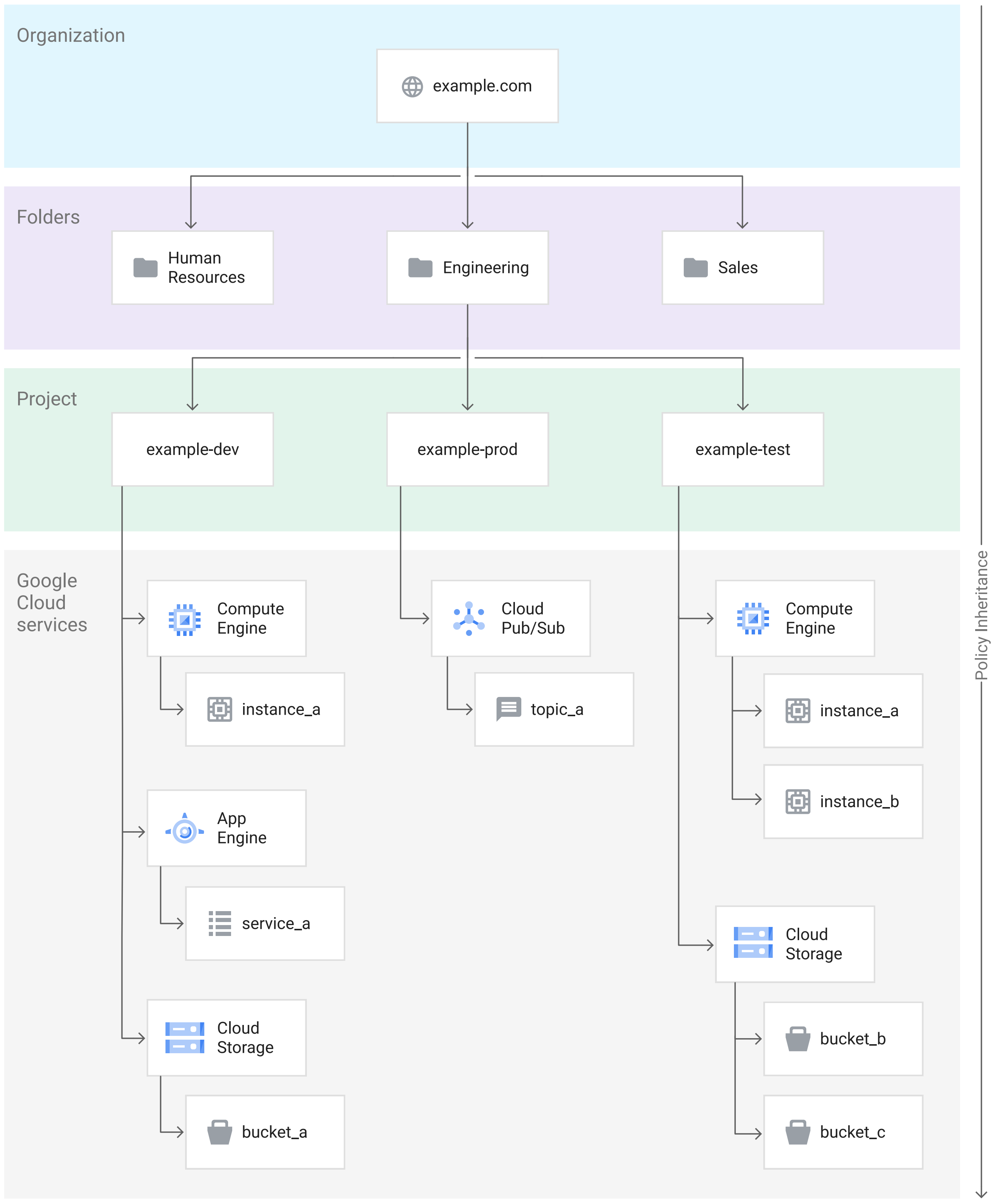
## Resource hierarchy

Google Cloud resources are organized hierarchically:

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

Each resource has exactly one parent. For more information, see the Resource Manager resource hierarchy (/resource-manager/docs/cloud-platform-resource-hierarchy).

The following diagram is an example of a Google Cloud resource hierarchy.



You can set an allow policy at any level in the resource hierarchy: the organization level, the folder level, the project level, or the resource level. Resources inherit the allow policies of all of their parent resources. The effective allow policy for a resource is the union of the allow policy set on that resource and the allow policies inherited from higher up in the hierarchy.

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

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

The allow policies for child resources inherit from the allow policies for their parent resources. 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. If you change the resource hierarchy, the policy inheritance changes as well. For example, moving a project into an organization causes the project to inherit from the organization's allow policy.

## IAM support for Google Cloud services

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

Google Cloud services offer predefined roles that provide fine-grained access control. For example, Compute Engine offers roles such as Compute Instance Admin and Compute Network Admin, and App Engine offers roles such as App Engine Admin and App Engine Service Admin.

Predefined roles are available for most Google Cloud services. For details, see the list of all predefined roles (/iam/docs/understanding-roles#predefined_roles). If you need even more control over permissions, consider creating a custom role (/iam/docs/creating-custom-roles).

You can grant users certain roles to access resources at a granularity *finer than the project level*. For example, you can create an allow policy that grants a user the Subscriber role for a particular Pub/Sub topic. The list of all predefined roles (/iam/docs/understanding-roles#predefined_roles) shows the *lowest-level*, or finest-grained, type of resource that accepts each role.

## Consistency model for the IAM API

The IAM API (/iam/docs/reference/rest) is eventually consistent (https://wikipedia.org/wiki/Eventual_consistency). In other words, if you write data with the IAM API, then immediately read that data, the read operation might return an older version of the data. Also, changes you make might take time to affect access checks.

This consistency model affects how the IAM API works. For example, if you create a service account, then immediately refer to that service account in another request, the IAM API might say that the service account could not be found. This behavior occurs because operations are eventually consistent; it can take time for the new service account to become visible to read requests.

## What's next

- For a list of available IAM roles, see Understanding roles (/iam/docs/understanding-roles).
- To get help with choosing the most appropriate predefined roles, read Choose predefined roles (/iam/docs/choose-predefined-roles).
- To learn about creating roles for your specific needs, read Understanding custom roles (/iam/docs/understanding-custom-roles).
- For instructions on how to grant, change, and revoke IAM roles to principals, see Granting, changing, and revoking access to resources (/iam/docs/granting-changing-revoking-access).
- Explore the Policy Intelligence tools (/iam/docs/policy-intelligence-tools), which help you understand and manage your allow policies to proactively improve your security configuration.
- To learn how to help secure your applications, see Identity-Aware Proxy overview (/iap/docs/concepts-overview).

Last updated 2023-06-08 UTC.