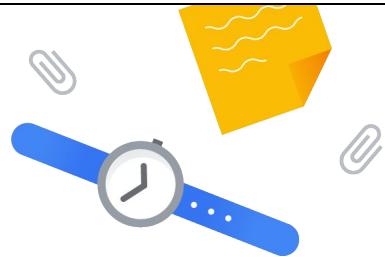




Service identity and authentication

Google Cloud

In this module, we discuss the fundamentals of service identity, and how you can invoke other Google Cloud services from your Cloud Run service.



01 Service account and identity

Agenda



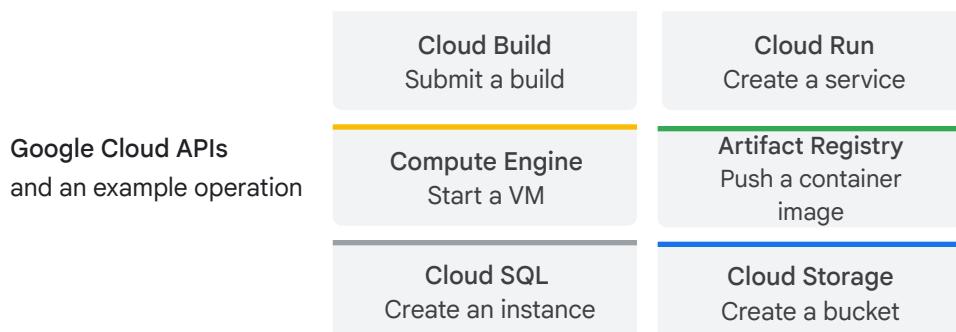
Google Cloud

In a previous module, you learned that Google Cloud is *a collection of APIs* that let you create virtual resources.

You invoke these APIs from your Cloud Run application code with client libraries.

In this topic, we examine how Cloud Run services use service accounts to invoke Google Cloud APIs.

Example Google Cloud APIs

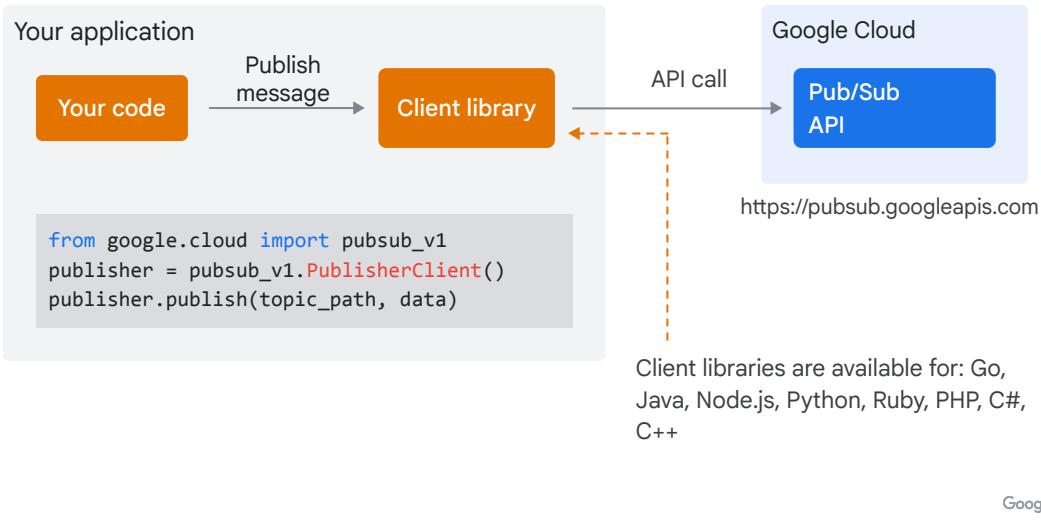


Google Cloud

Here are some examples of Google Cloud products with an example API call.

To interact with these service APIs, you can use the gcloud CLI, the web console, a client library, or use any other process that calls the API, like Terraform.

An example API call from an application



Here's another example API call from an application using a client library.

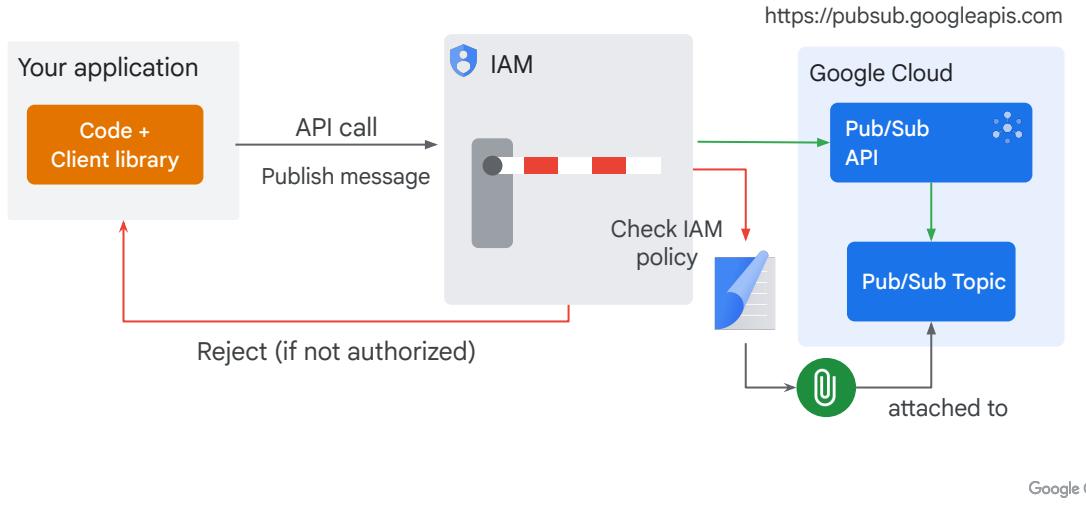
Pub/Sub is a message broker service on Google Cloud that you can use to send messages between services asynchronously.

To publish a message to Pub/Sub from a Python application, you would use the pubsub client library as shown in the sample code. The client library handles the API call to pubsub.googleapis.com.

For most Google Cloud services, there are client libraries available for Go, Java, Node.js, Python, Ruby, PHP, C# and C++ programming languages.

<https://cloud.google.com/apis/docs/cloud-client-libraries>

Publishing a message requires authorization

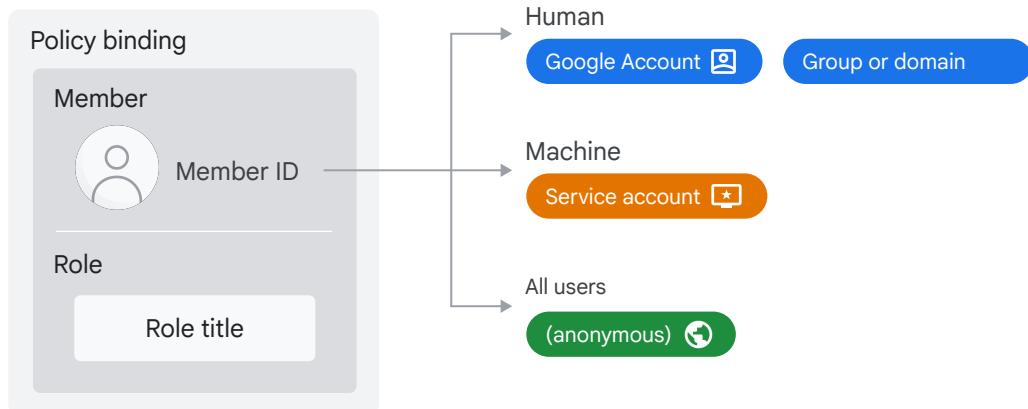


In this example, the application code uses the pubsub client library to send an API call to `pubsub.googleapis.com`. IAM will inspect the request and identify your application by the credentials in the API request.

Now that IAM has the identity, it will need to figure out what operations are allowed for the identity to perform on the Pub/Sub topic.

IAM does this by checking policy bindings in an IAM policy that you attach to the Pub/Sub topic.

Policy binding



Google Cloud

A policy binding binds one or more members (identities) to a single role. A role contains a set of permissions that allows the member identity to perform specific actions on Google Cloud resources. For example, the Pub/Sub Publisher role includes the pubsub.topics.publish permission that provides access to publish messages to a topic.

IAM supports the following types of identities:

- *Human identities*: Your Google account is a human identity which you use to sign in to Google Cloud. Your Google Account can also be part of a group or a domain.
- *Service account*: Used by machines or applications. Examples of machines with a service identity are a virtual machine, a Cloud Run service, a Cloud Run function, or other services.
- *All users*: A special identifier to allow everyone or allow public access to a service on Google Cloud.

A member can be attached to multiple policy bindings in an IAM policy enabling that member to have more than one role.

Service account

- A service account is a special type of identity that is meant to be used by machines.
- A service account does not have a password and can't sign in using a browser.



Service account

Google Cloud

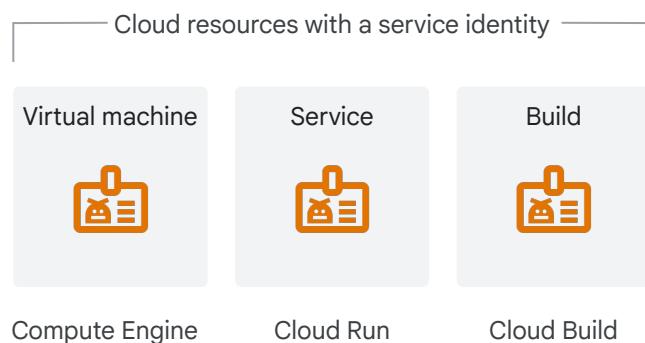
Just like your Google Account is your identity from the point of view of IAM, a Cloud Run service also has its own identity, called a service account.

A service account is a special type of account used by machines, applications, or services. It's identified by its email address, which is unique to the account.

Service accounts differ from user accounts in a few key ways:

- Service accounts do not have passwords, and cannot sign in by using browsers or cookies.
- You can let other users or service accounts act on behalf of a service account.
- Service accounts are not members of your Google Workspace domain, unlike user accounts, although you can add them to groups.

Service account usage



Google Cloud

Service accounts are meant to be used by machines.

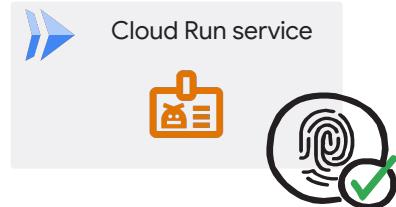
If you run code somewhere, for example on a virtual machine, in a Cloud Run service, or as part of a build in Cloud Build, you have access to a built-in service account.

If you use one of the client libraries to connect to Google Cloud APIs, the libraries will automatically use this built-in service account for authentication.

You can always replace the service account with your own user-managed service account, which is recommended.

Service accounts in Cloud Run

- Every Cloud Run service or job is linked to a service account, known as the service identity.
- Use a per-service account for each service.
- Grant minimal, selective permissions to each service account.



Google Cloud

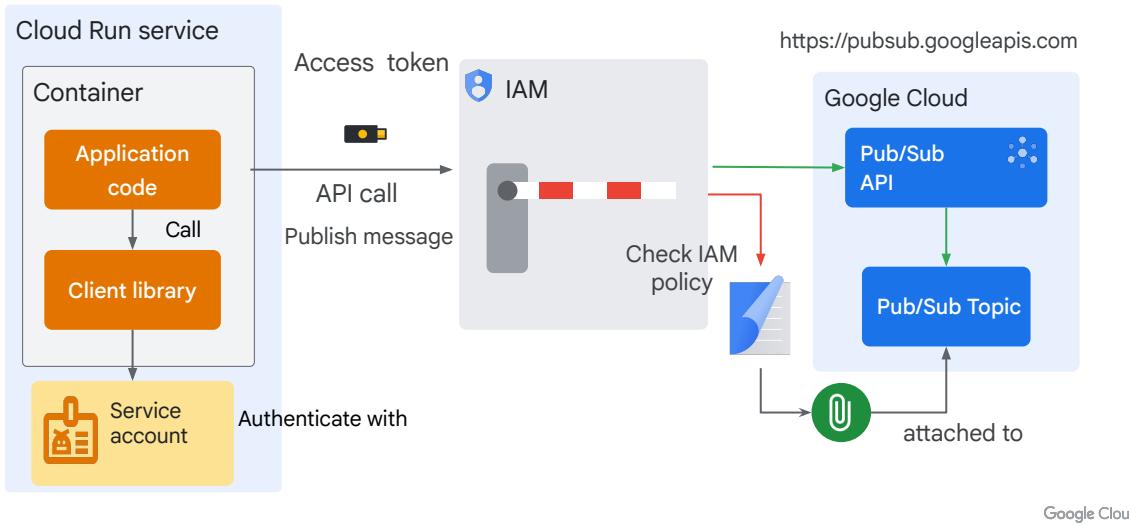
Every Cloud Run service or job is linked to a service account, that is also known as the “service identity.”

By default, Cloud Run services or jobs run as the [default Compute Engine service account](#) with the Editor role.

It's recommended to use a user-managed service account with the most minimal set of permissions required for the service to perform its functions.

A best practice is to use a service account for each service identity, and grant selective permissions to the account.

Service identity on Cloud Run



In practice, a container with your application runs as part of a Cloud Run service.

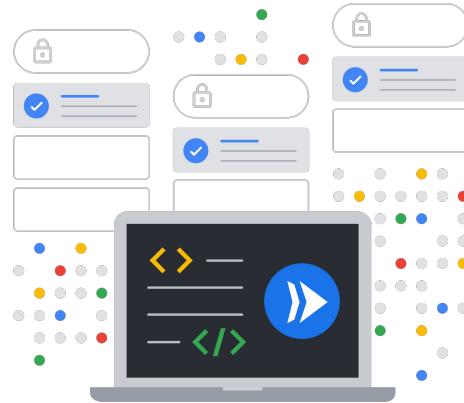
If you use a client library in your application code to publish a message to a Pub/Sub topic, the library automatically acquires appropriate tokens to authenticate your code's requests using the service's runtime service account. When accessing most Google APIs, OAuth 2.0 access tokens are used.

The access token is used to call the Pub/Sub API.

IAM verifies the access token, and uses the identity in the access token to check if there is a policy binding with the required roles to publish a message to the attached Pub/Sub topic.

Remember

- 1 An IAM policy contains a list of policy bindings that bind members to a role.
- 2 A service account is a type of member identity that is used by machines, applications, or services.
- 3 Every Cloud Run service or job is linked to a service account.
- 4 Use a user-managed service account for each Cloud Run service with a minimum set of permissions.

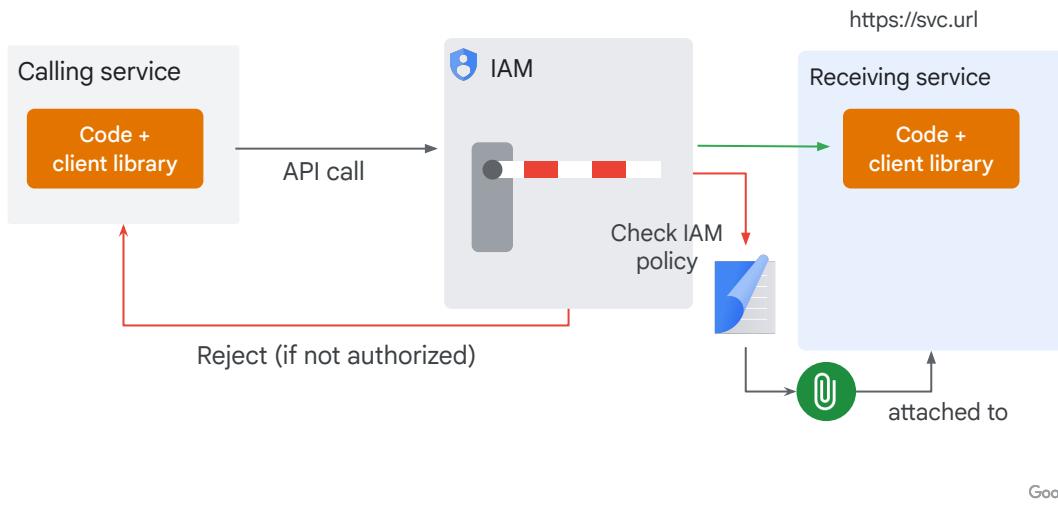


Google Cloud

In summary:

- An IAM policy contains a list of policy bindings that bind members to a role. To authorize API and service calls, IAM reads the IAM policy that is attached to a resource.
- A service account is a type of member identity that is used by machines, applications, or services.
- Every Cloud Run service or job is linked to a service account.
- Use a user-managed service account for each Cloud Run service with a minimum set of permissions.

Service to service communication



If your application architecture uses multiple Cloud Run services, these services likely need to communicate with each other. This communication could be either asynchronous or synchronous. Many of these services may be private and therefore require credentials for access.

For asynchronous communication, you can use various Google Cloud services such as Cloud Tasks, Pub/Sub, Cloud Scheduler, or Eventarc.

For synchronous communication, your service calls another service's endpoint URL directly over HTTP. In this case, it's a best practice to use IAM and an individual service identity for the calling service. The service account is granted the minimum set of permissions required.

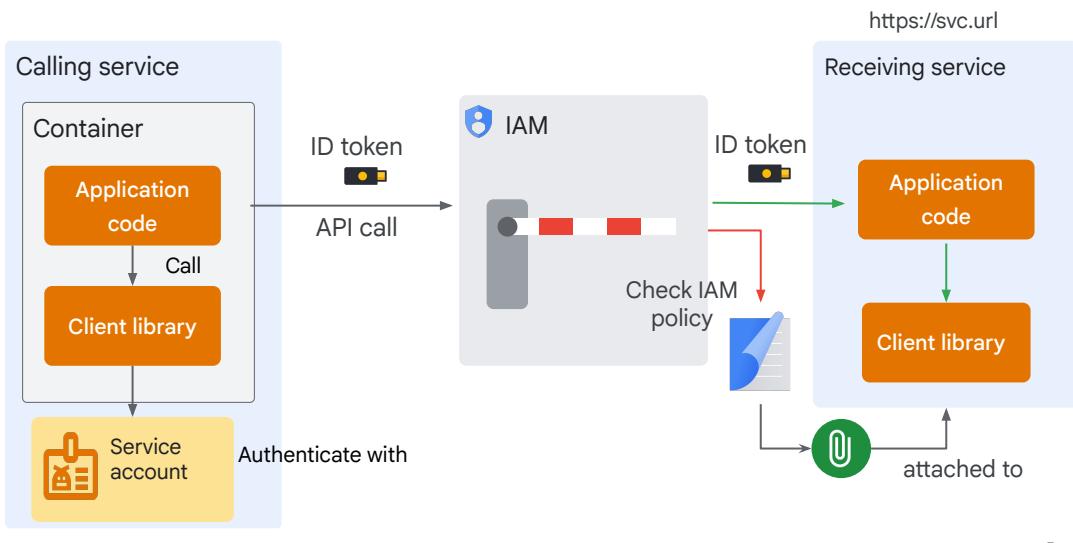
To set up a service account, you configure the receiving service to accept requests from the calling service by making the calling service's service account a *principal* on the receiving service. Then you grant that service account the *Cloud Run Invoker* (`roles/run.invoker`) role.

You can do this in the Google Cloud console, with the gcloud CLI, or with Terraform. Here's the gcloud CLI command:

```
gcloud run services add-iam-policy-binding RECEIVING_SERVICE
```

```
--member='serviceAccount:CALLING_SERVICE_IDENTITY' --role='roles/run.invoker'
```

Service to service communication



Google Cloud

The request made by the calling service must present proof of this identity in the form of a Google-signed [OpenID Connect](#) token.

OpenID Connect is an identity protocol based on OAuth 2.0 that enables identity verification of a client based on the authentication performed by an authorization server. It's also used to obtain basic profile information about the client.

One way to acquire this ID token is to use the Google authentication client libraries in your calling service's application code.

In the receiving service, your application code can use Google's authentication libraries to parse the request, and extract and verify information from the ID token.