ConductorOne

Implementing Just-in-Time Access in Google Cloud Platform (GCP)

Just-in-time (JIT) access is a security best practice that grants users access to resources only when they require them and for a limited amount of time. JIT access helps minimize data breaches and unauthorized cloud resource use, effectively providing users with the permissions they need when needed—as opposed to conventional static access models, which offer access without any time limitation.

JIT access can help significantly decrease Google Cloud Platform's attack surface by issuing permissions as needed, thus limiting entry points and the potential for unauthorized access. Users operate with only as much privilege as is necessary for their roles. This reduces the risks of accidental data exposure or system misuse and keeps accounts secure by restricting an attacker's scope and duration of access. GCP also supports port-level access control via JIT access, allowing for fine-grained control over network ports or services on virtual machines.

JIT access introduces an automated or routed approval system where users must request access when needed and approvals can be automated or routed through the appropriate channels. Not only can JIT access increase security, but it can also help organizations maintain more visibility and control over access requests.

JIT access solutions usually feature comprehensive auditing features that monitor all access activities and create an audit trail of when resources were accessed and by whom. In the unfortunate event of a security incident, JIT access simplifies the incident response process. Access revocation can be triggered immediately, limiting potential damage by cutting off unauthorized access. This rapid response capability is critical to containing and mitigating security breaches.

In this article, you'll learn how to implement JIT access in Google Cloud Platform.

How to Implementing Just-in-Time Access in Google Cloud Platform

You can follow the step-by-step instructions below to implement JIT access in GCP.

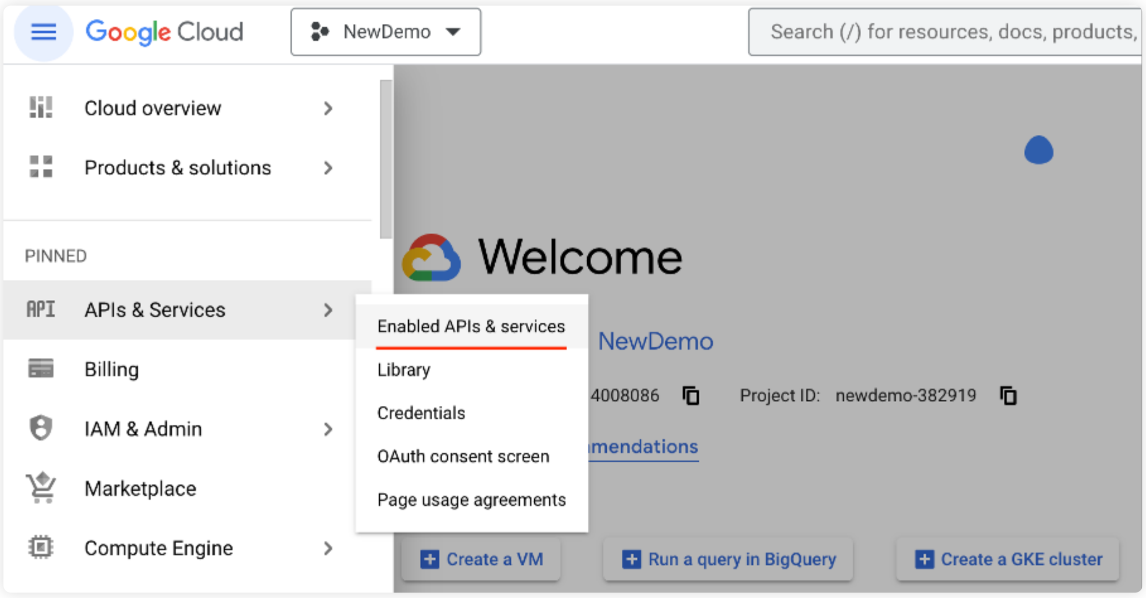
Enabling Required APIs in Your GCP Project for the JIT Access App

Before using the JIT Access app for GCP, you'll need to enable the following APIs for it:

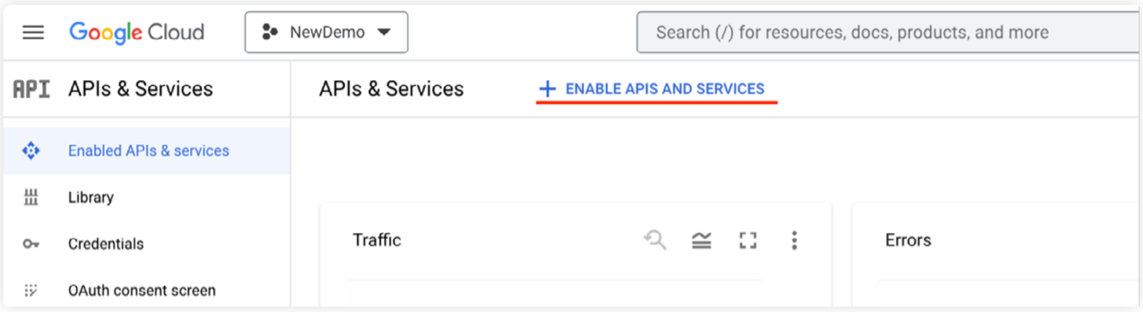
- Cloud Asset Inventory
- Resource Manager
- Identity-Aware Proxy (IAP)
- Artifact Registry
- Cloud Build

These APIs are required to manage resources, authenticate users, store container images, and build and deploy applications in GCP.

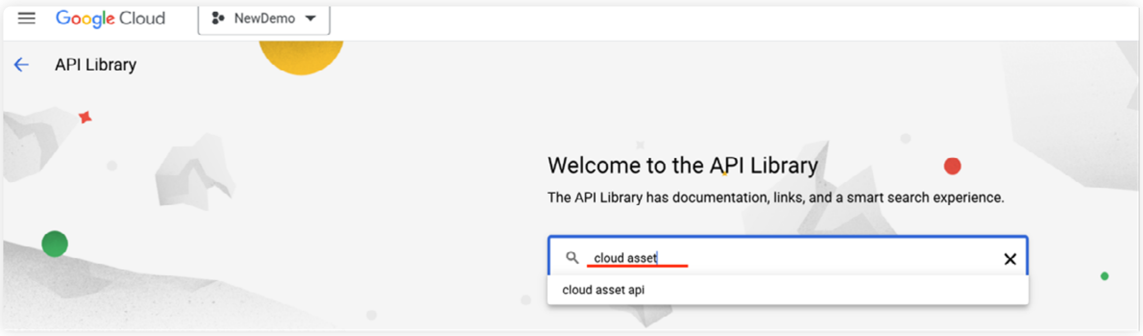
Go to the **APIs & Services > Enabled APIs & services** page in the GCP console:



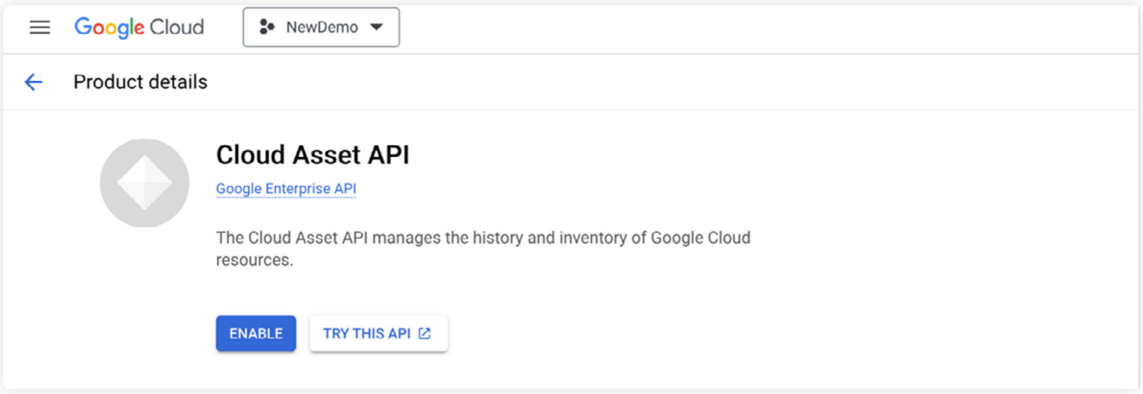
Click **ENABLE APIS AND SERVICES:**



In the search bar, enter the name of the API you want to enable (for example, “Cloud Asset Inventory”):


Google Cloud
NewDemo
API Library
Welcome to the API Library
The API Library has documentation, links, and a smart search experience.
cloud asset
cloud asset api

Select the name of the API, then click the **ENABLE** button on the details page to enable the API:

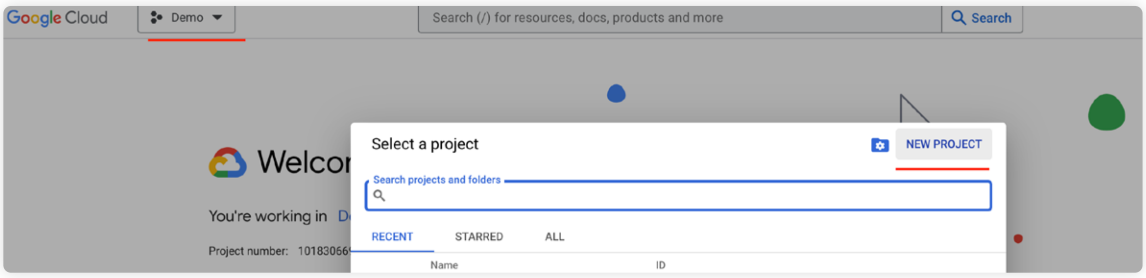


The Cloud Asset API manages the history and inventory of Google Cloud resources.

Once you've enabled all the required APIs, you can deploy the JIT Access app.

Creating a Service Account for the JIT Access App

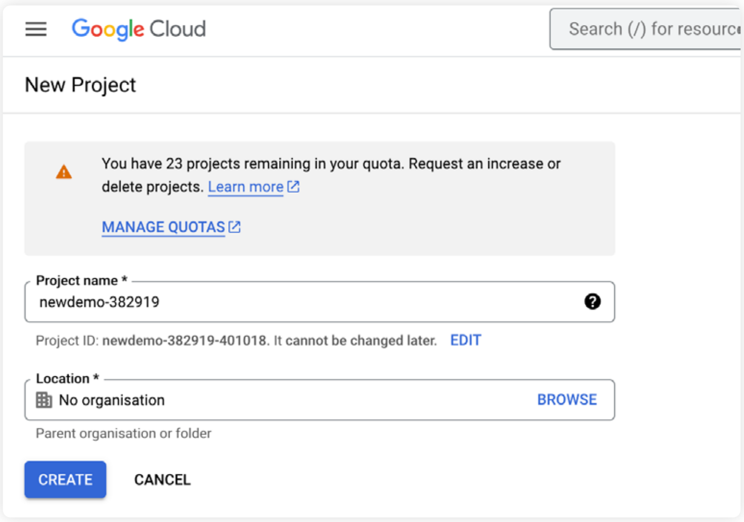
Before creating a service account, you'll need to create a new project if you don't already have one. Click the drop-down menu in the top left and select **NEW PROJECT:**



Enter a project name and click **CREATE:**

Open [Google Cloud Shell]() and set the environment variable for your project. Run the following command, replacing `newdemo-382919` below with your project name if you used a different name:

```
gcloud config set
project newdemo-382919
```



You have 23 projects remaining in your quota. Request an increase or delete projects. Learn more

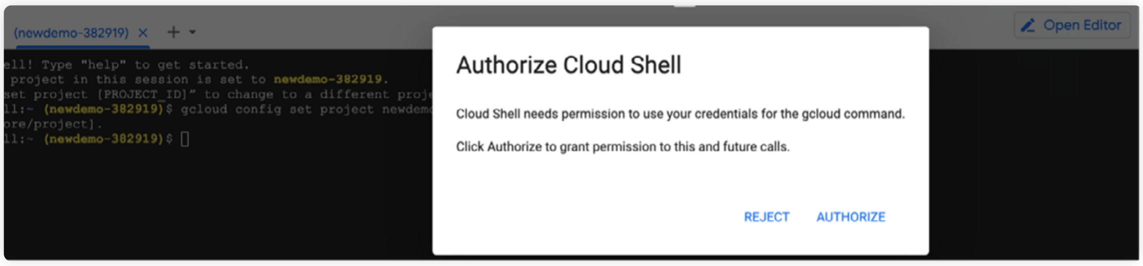
Project name *
newdemo-382919

Project ID: newdemo-382919-401018. It cannot be changed later. EDIT

Location *
No organisation

Parent organisation or folder

Click **AUTHORIZE** on the resulting pop-up window:



```
ell! Type "help" to get started.
 project in this session is set to newdemo-382919.
set project [PROJECT_ID]" to change to a different proj
ll:~ (newdemo-382919)$ gcloud config set project newdem
ore/project].
ll:~ (newdemo-382919)$ 
```

Authorize Cloud Shell

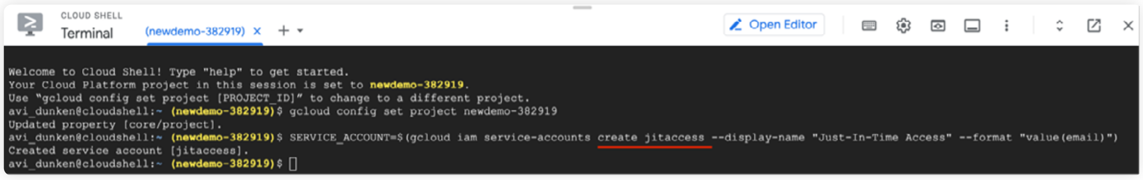
Cloud Shell needs permission to use your credentials for the gcloud command.

Click Authorize to grant permission to this and future calls.

REJECT AUTHORIZE

You'll need a service account for your application. Run the following command to create one named `Just-In-Time Access:`

```
SERVICE_ACCOUNT=$(gcloud iam service-accounts create jitaccess
--display-name “Just-In-Time Access” --format “value(email)”)
```



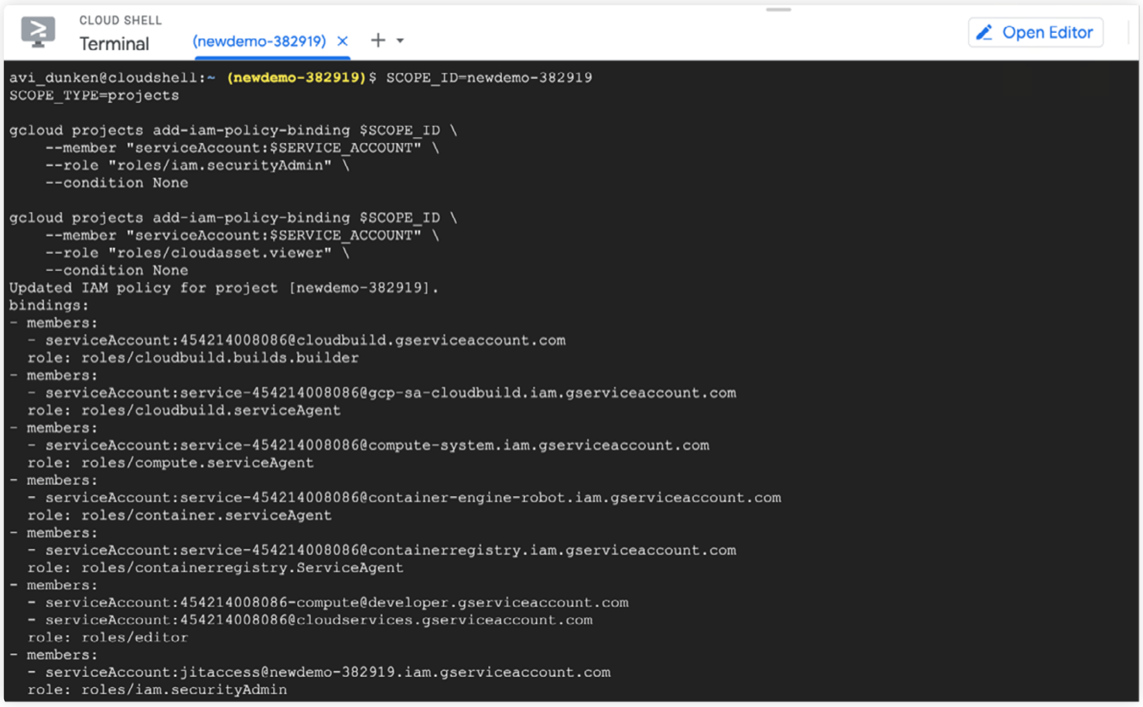
```
Welcome to Cloud Shell! Type "help" to get started.
Your Cloud Platform project in this session is set to newdemo-382919.
Use "gcloud config set project [PROJECT_ID]" to change to a different project.
avi_dunken@cloudshell:~ (newdemo-382919)$ gcloud config set project newdemo-382919
Updated property [core/project].
avi_dunken@cloudshell:~ (newdemo-382919)$ SERVICE_ACCOUNT=$(gcloud iam service-accounts create jitaccess --display-name "Just-In-Time Access" --format "value(email)")
Created service account [jitaccess].
avi_dunken@cloudshell:~ (newdemo-382919)$ 
```

Allowing the JIT Access App to Manage IAM Bindings

Run the following command in Cloud Shell to provide the relevant permissions for the JIT Access app to manage IAM bindings:

```
SCOPE_ID=newdemo-382919
SCOPE_TYPE=projects

gcloud projects add-iam-policy-binding $SCOPE_ID \
    --member “serviceAccount:$SERVICE_ACCOUNT” \
    --role “roles/iam.securityAdmin” \
    --condition None

gcloud projects add-iam-policy-binding $SCOPE_ID \
    --member “serviceAccount:$SERVICE_ACCOUNT” \
    --role “roles/cloudasset.viewer” \
    --condition None
```

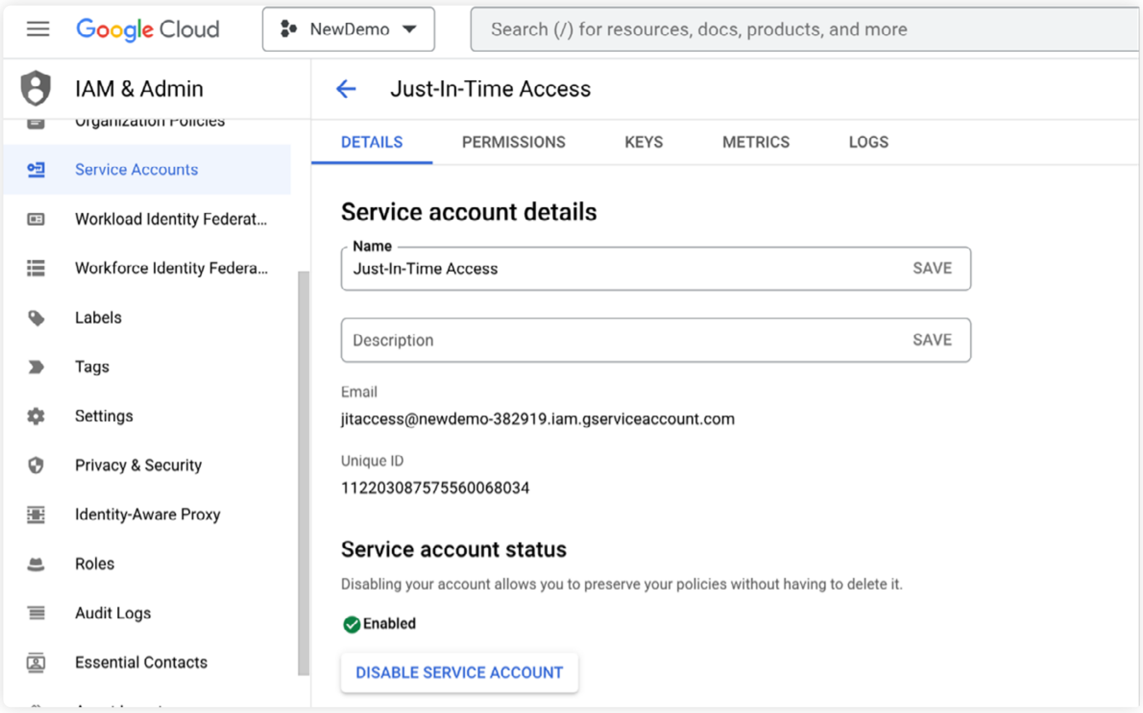
This provides the necessary access and assigns both Security Admin (`roles/iam.securityAdmin`) and Cloud Asset Viewer (`roles/cloudasset.viewer`) roles to the relevant section of your resource hierarchy:



```
avi_dunken@cloudshell:~ (newdemo-382919)$ SCOPE_ID=newdemo-382919
SCOPE_TYPE=projects

gcloud projects add-iam-policy-binding $SCOPE_ID \
    --member "serviceAccount:$SERVICE_ACCOUNT" \
    --role "roles/iam.securityAdmin" \
    --condition None

gcloud projects add-iam-policy-binding $SCOPE_ID \
    --member "serviceAccount:$SERVICE_ACCOUNT" \
    --role "roles/cloudasset.viewer" \
    --condition None
Updated IAM policy for project [newdemo-382919].
bindings:
- members:
  - serviceAccount:454214008086@cloudbuild.gserviceaccount.com
  role: roles/cloudbuild.builds.builder
- members:
  - serviceAccount:service-454214008086@gcp-sa-cloudbuild.iam.gserviceaccount.com
  role: roles/cloudbuild.serviceAgent
- members:
  - serviceAccount:service-454214008086@compute-system.iam.gserviceaccount.com
  role: roles/compute.serviceAgent
- members:
  - serviceAccount:service-454214008086@container-engine-robot.iam.gserviceaccount.com
  role: roles/container.serviceAgent
- members:
  - serviceAccount:service-454214008086@containerregistry.iam.gserviceaccount.com
  role: roles/containerregistry.ServiceAgent
- members:
  - serviceAccount:454214008086-compute@developer.gserviceaccount.com
  - serviceAccount:454214008086@cloudservices.gserviceaccount.com
  role: roles/editor
- members:
  - serviceAccount:jitaccess@newdemo-382919.iam.gserviceaccount.com
  role: roles/iam.securityAdmin
```

Once you've completed the above step, you will have successfully granted the JIT Access application permission to manage IAM bindings in Cloud Shell.

To verify this, navigate to the **IAM & Admin** page in the GCP console and click **Service Accounts**. You can see that the "Just-In-Time Access" service account has been created and enabled:



Service account details

Name: Just-In-Time Access

Description

Email
jitaccess@newdemo-382919.iam.gserviceaccount.com

Unique ID
112203087575560068034

Service account status

Disabling your account allows you to preserve your policies without having to delete it.

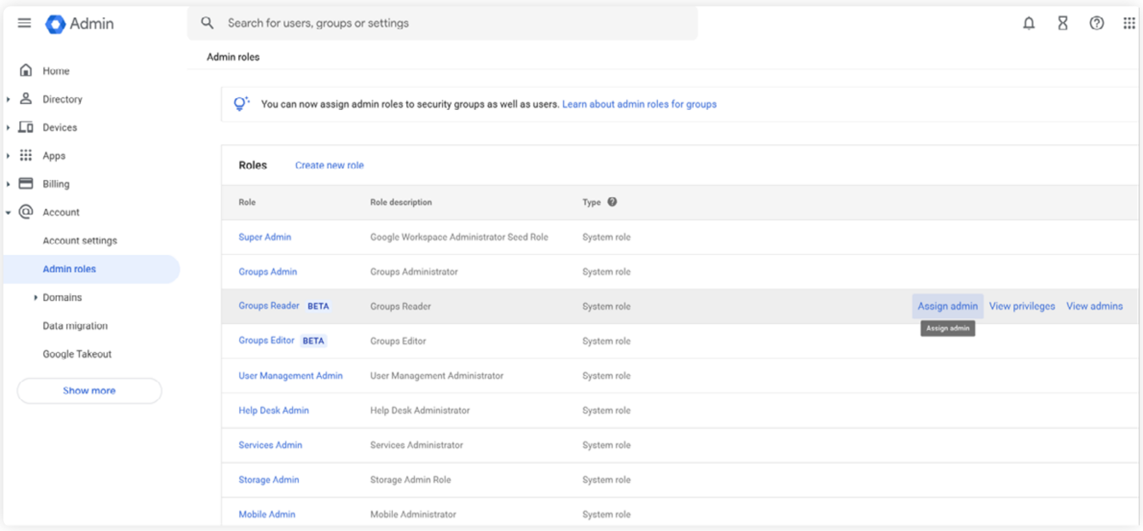
✓ Enabled

Allowing the JIT Access App to Resolve Group Memberships

You next need to give the app access to group memberships for the service account.

Start by signing in with a super admin user account in the Google Admin console.

In the Admin console, expand the **Account** drop-down menu and click **Admin roles**.
Then, under the **Groups Reader** role, click **Assign admin:**

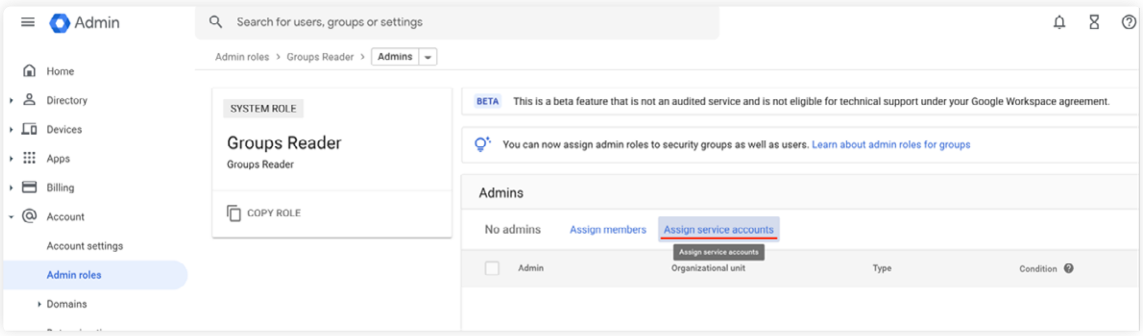


You can now assign admin roles to security groups as well as users. Learn about admin roles for groups

Roles

Role	Role description	Type
Super Admin	Google Workspace Administrator Seed Role	System role
Groups Admin	Groups Administrator	System role
Groups Reader BETA	Groups Reader	System role
Groups Editor BETA	Groups Editor	System role
User Management Admin	User Management Administrator	System role
Help Desk Admin	Help Desk Administrator	System role
Services Admin	Services Administrator	System role
Storage Admin	Storage Admin Role	System role
Mobile Admin	Mobile Administrator	System role

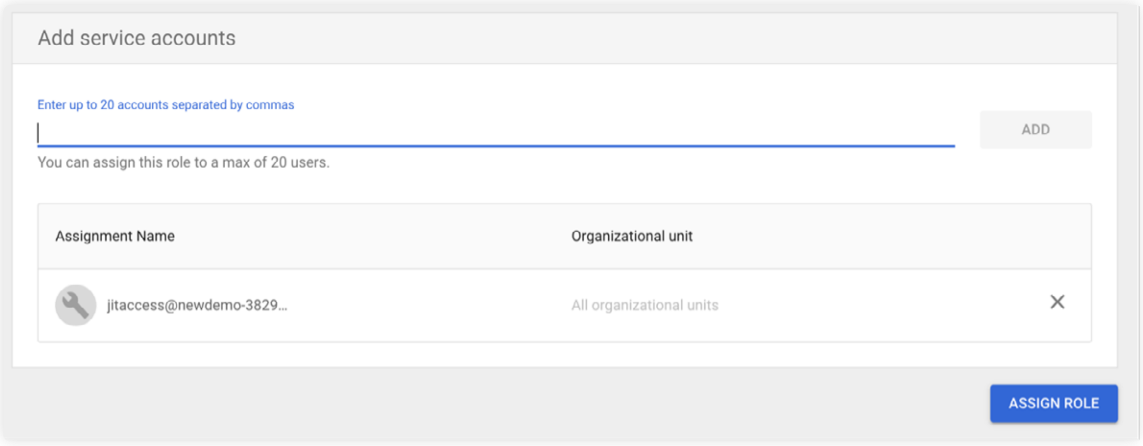
Select **Assign service accounts:**



BETA This is a beta feature that is not an audited service and is not eligible for technical support under your Google Workspace agreement.

You can now assign admin roles to security groups as well as users. Learn about admin roles for groups

Enter `jitaccess@newdemo-382919.iam.gserviceaccount.com`, replacing `newdemo-382919` with your project name, then click **ADD**, followed by **ASSIGN ROLE**:


Add service accounts
Enter up to 20 accounts separated by commas
ADD
You can assign this role to a max of 20 users.
Assignment Name
Organizational unit
jitaccess@newdemo-3829...
All organizational units
ASSIGN ROLE

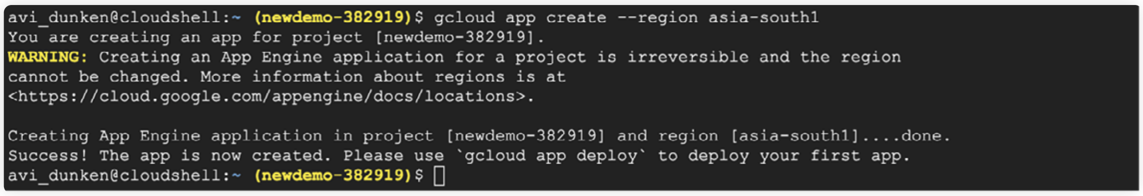
Once you have completed these steps, the application will have access to resolve the group memberships of the users in the group.

Deploying the JIT Access App to App Engine

You'll now deploy the JIT Access application on App Engine.

First, open Cloud Shell and create an App Engine application. Choose the region closest to your physical location, then use the `gcloud app create` command:

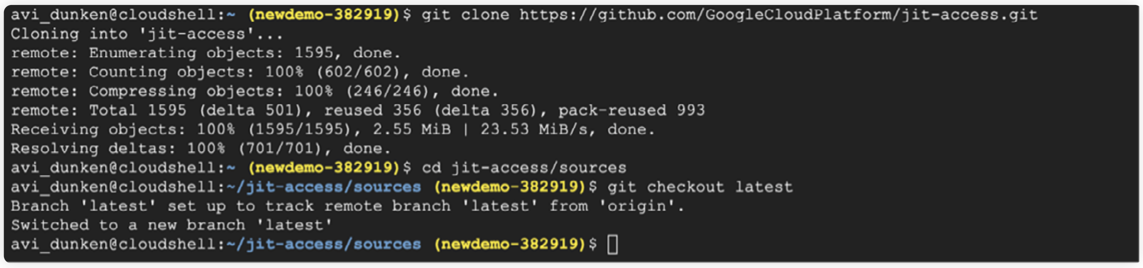
```
gcloud app create --region asia-south1
```



```
avi_dunken@cloudshell:~ (newdemo-382919)$ gcloud app create --region asia-south1
You are creating an app for project [newdemo-382919].
WARNING: Creating an App Engine application for a project is irreversible and the region
cannot be changed. More information about regions is at
<https://cloud.google.com/appengine/docs/locations>.

Creating App Engine application in project [newdemo-382919] and region [asia-south1]....done.
Success! The app is now created. Please use `gcloud app deploy` to deploy your first app.
avi_dunken@cloudshell:~ (newdemo-382919)$
```

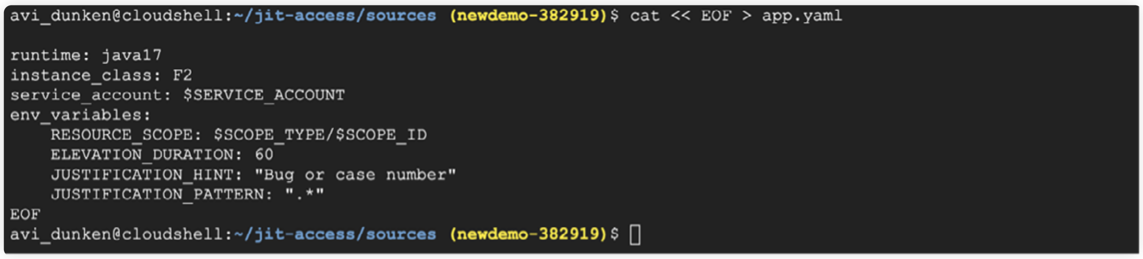
Clone the JIT access GitHub repository in Cloud Shell and `checkout` to the most recent branch:

```
git clone https://github.com/GoogleCloudPlatform/jit-access.git
cd jit-access/sources
git checkout latest
```


```
avi_dunken@cloudshell:~ (newdemo-382919)$ git clone https://github.com/GoogleCloudPlatform/jit-access.git
Cloning into 'jit-access'...
remote: Enumerating objects: 1595, done.
remote: Counting objects: 100% (602/602), done.
remote: Compressing objects: 100% (246/246), done.
remote: Total 1595 (delta 501), reused 356 (delta 356), pack-reused 993
Receiving objects: 100% (1595/1595), 2.55 MiB | 23.53 MiB/s, done.
Resolving deltas: 100% (701/701), done.
avi_dunken@cloudshell:~ (newdemo-382919)$ cd jit-access/sources
avi_dunken@cloudshell:~/jit-access/sources (newdemo-382919)$ git checkout latest
Branch 'latest' set up to track remote branch 'latest' from 'origin'.
Switched to a new branch 'latest'
avi_dunken@cloudshell:~/jit-access/sources (newdemo-382919)$
```

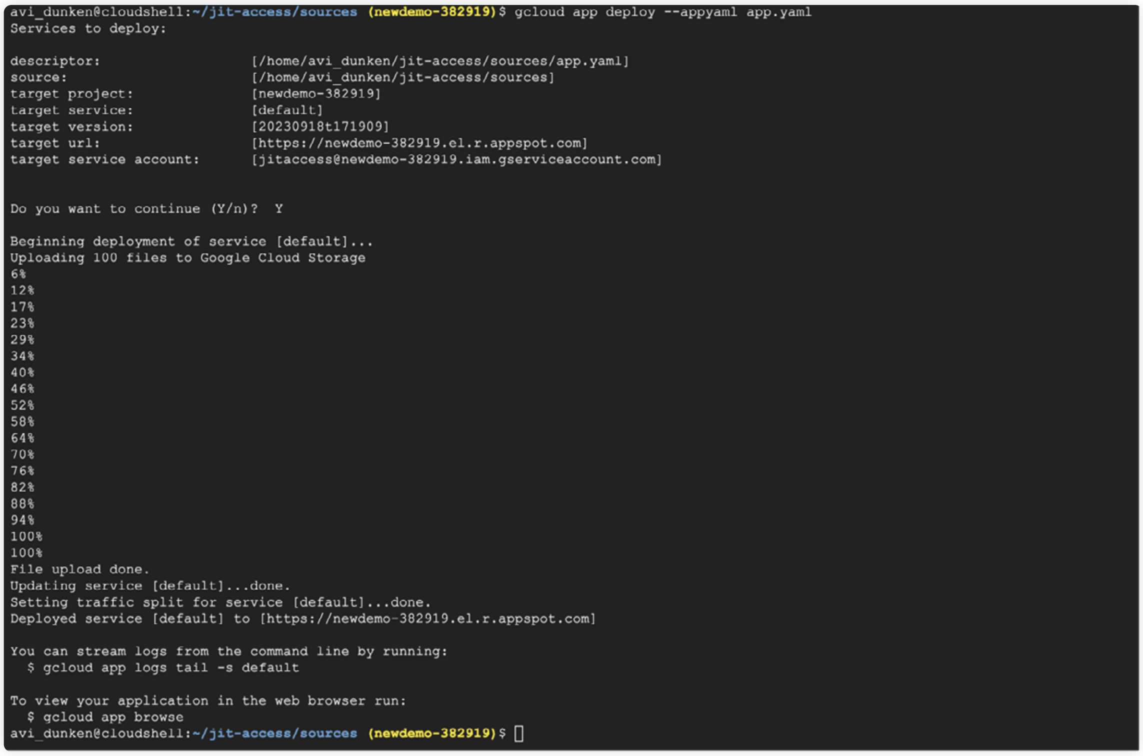
You'll also need a configuration file for the JIT Access app:

```
cat << EOF > app.yaml

runtime: java17
instance_class: F2
service_account: $SERVICE_ACCOUNT
env_variables:
    RESOURCE_SCOPE: $SCOPE_TYPE/$SCOPE_ID
    ELEVATION_DURATION: 60
    JUSTIFICATION_HINT: “Bug or case number”
    JUSTIFICATION_PATTERN: “.*”
EOF
```


```
avi_dunken@cloudshell:~/jit-access/sources (newdemo-382919)$ cat << EOF > app.yaml

runtime: java17
instance_class: F2
service_account: $SERVICE_ACCOUNT
env_variables:
    RESOURCE_SCOPE: $SCOPE_TYPE/$SCOPE_ID
    ELEVATION_DURATION: 60
    JUSTIFICATION_HINT: "Bug or case number"
    JUSTIFICATION_PATTERN: ".*"
EOF
avi_dunken@cloudshell:~/jit-access/sources (newdemo-382919)$
```

Finally, use the following command to deploy the application:

```
gcloud app deploy --appyaml app.yaml
```



```
avi_dunken@cloudshell:~/jit-access/sources (newdemo-382919)$ gcloud app deploy --appyaml app.yaml
Services to deploy:

descriptor:                  [/home/avi_dunken/jit-access/sources/app.yaml]
source:                      [/home/avi_dunken/jit-access/sources]
target project:              [newdemo-382919]
target service:              [default]
target version:              [20230918t171909]
target url:                  [https://newdemo-382919.el.r.appspot.com]
target service account:      [jitaccess@newdemo-382919.iam.gserviceaccount.com]


Do you want to continue (Y/n)?  Y

Beginning deployment of service [default]...
Uploading 100 files to Google Cloud Storage
6%
12%
17%
23%
29%
34%
40%
46%
52%
58%
64%
70%
76%
82%
88%
94%
100%
100%
File upload done.
Updating service [default]...done.
Setting traffic split for service [default]...done.
Deployed service [default] to [https://newdemo-382919.el.r.appspot.com]

You can stream logs from the command line by running:
  $ gcloud app logs tail -s default

To view your application in the web browser run:
  $ gcloud app browse
avi_dunken@cloudshell:~/jit-access/sources (newdemo-382919)$ 
```

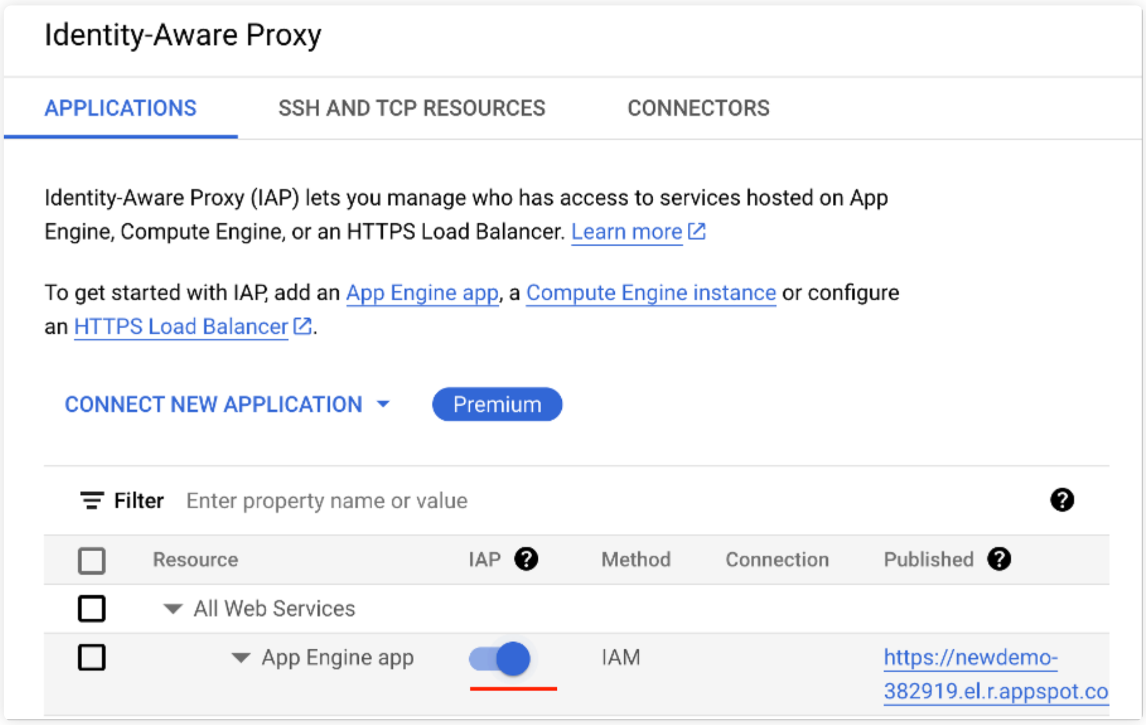
In the above screenshot, you can see the public URL that the service is deployed to. In this case, it's `https://newdemo-382919.el.r.appspot.com`, but your URL will reflect your specific project name. This tutorial uses the default settings for the automatically configured load balancer, but you can adjust them to your liking.

Configuring Identity-Aware Proxy (IAP)

You next have to configure IAP for the JIT Access app to improve security, access management, and compliance. First, use the following code to set up an OAuth consent screen:

```
gcloud iap oauth-brands create \
    --application_title “Just-In-Time Access” \
    --support_email=$(gcloud config get core/account)
```

Then, in the GCP console, first navigate to **Security** and then **Identity-Aware Proxy**.
Enable the **IAP** toggle under the **APPLICATIONS** tab:


Identity-Aware Proxy

APPLICATIONS | SSH AND TCP RESOURCES | CONNECTORS

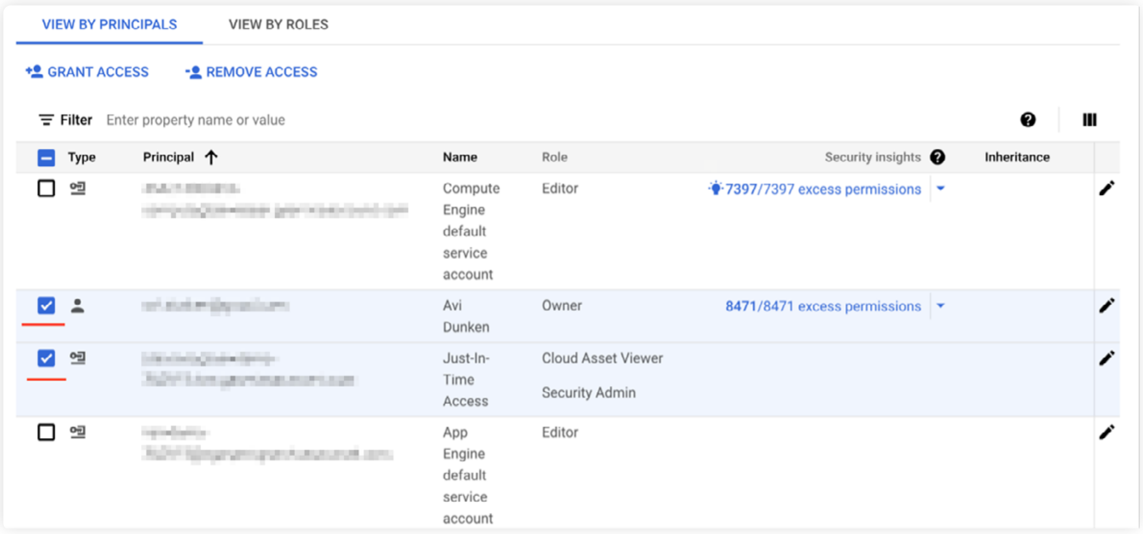
Identity-Aware Proxy (IAP) lets you manage who has access to services hosted on App Engine, Compute Engine, or an HTTPS Load Balancer. Learn more

To get started with IAP, add an App Engine app, a Compute Engine instance or configure an HTTPS Load Balancer.

☐	Resource	IAP	Method	Connection	Published
☐	All Web Services				
☐	App Engine app	●	IAM		https://newdemo-382919.el.r.appspot.co

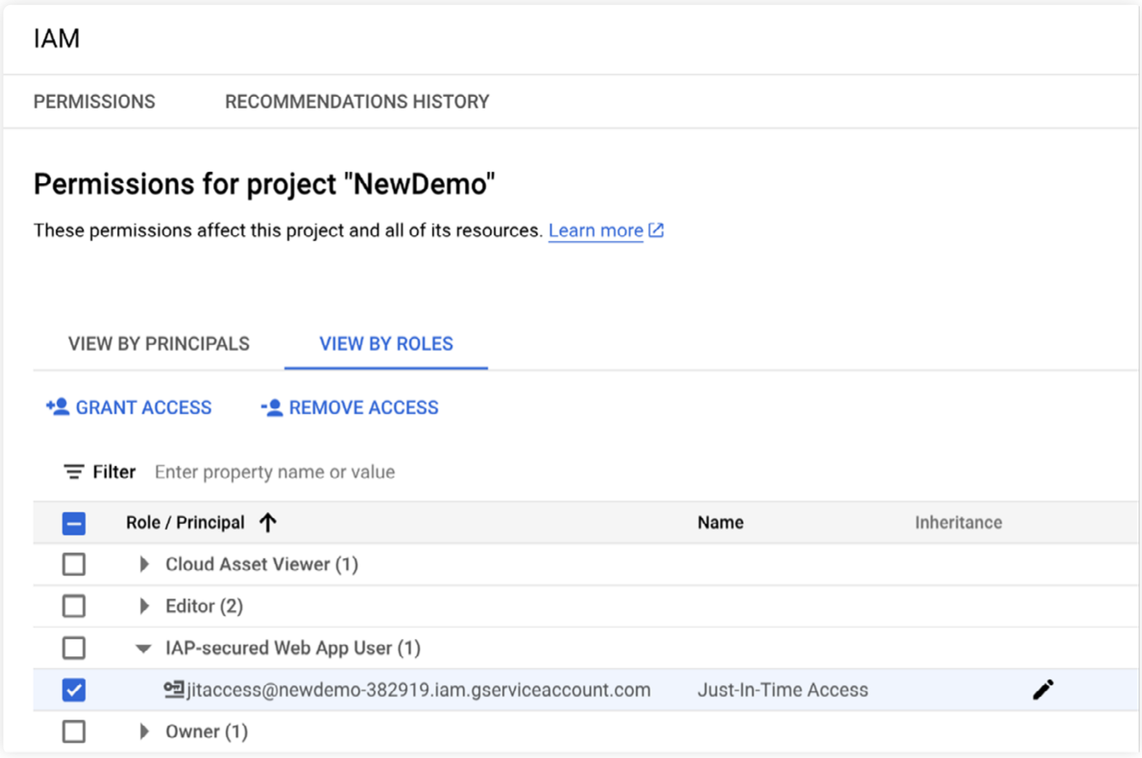
You now have to specify which users can access your JIT Access app. You can give access permission to individuals, groups, or entire domains.

Back in the console, navigate to **IAM & Admin** section, and, under **IAM**, click **GRANT ACCESS**. Then, select users, groups, or domains in the list of principals by clicking the checkbox to the left of each item:



	Type	Principal	Name	Role	Security insights	Inheritance
☐		[redacted]	Compute Engine default service account	Editor	7397/7397 excess permissions	
✓		[redacted]	Avi Dunken	Owner	8471/8471 excess permissions	
✓		[redacted]	Just-In-Time Access	Cloud Asset Viewer Security Admin		
☐		[redacted]	App Engine default service account	Editor		

Expand **IAP-secured Web App User** in the list of roles and select the appropriate user by clicking the checkbox to its left:



IAM

PERMISSIONS RECOMMENDATIONS HISTORY

Permissions for project "NewDemo"

These permissions affect this project and all of its resources. Learn more

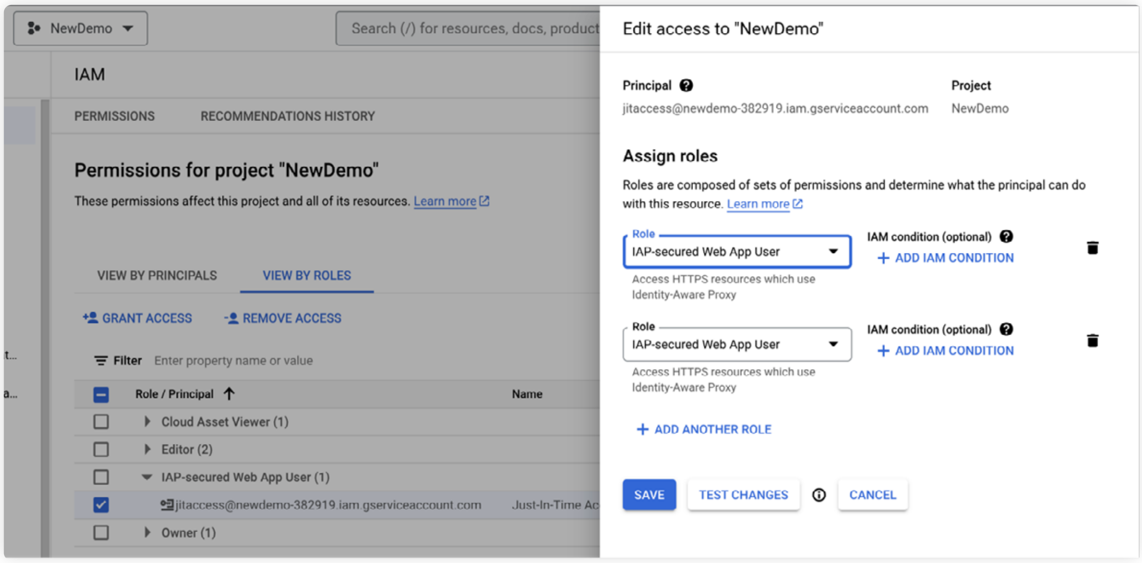
VIEW BY PRINCIPALS VIEW BY ROLES

GRANT ACCESS REMOVE ACCESS

Filter Enter property name or value

	Role / Principal ↑	Name	Inheritance
☐	Cloud Asset Viewer (1)		
☐	Editor (2)		
☐	IAP-secured Web App User (1)		
✓	jitaccess@newdemo-382919.iam.gserviceaccount.com	Just-In-Time Access	
☐	Owner (1)		

Click **SAVE**:



Edit access to "NewDemo"

Principal: jitaccess@newdemo-382919.iam.gserviceaccount.com

Project: NewDemo

Assign roles

Roles are composed of sets of permissions and determine what the principal can do with this resource. Learn more

Role: IAP-secured Web App User — Access HTTPS resources which use Identity-Aware Proxy — IAM condition (optional): ADD IAM CONDITION

Role: IAP-secured Web App User — Access HTTPS resources which use Identity-Aware Proxy — IAM condition (optional): ADD IAM CONDITION

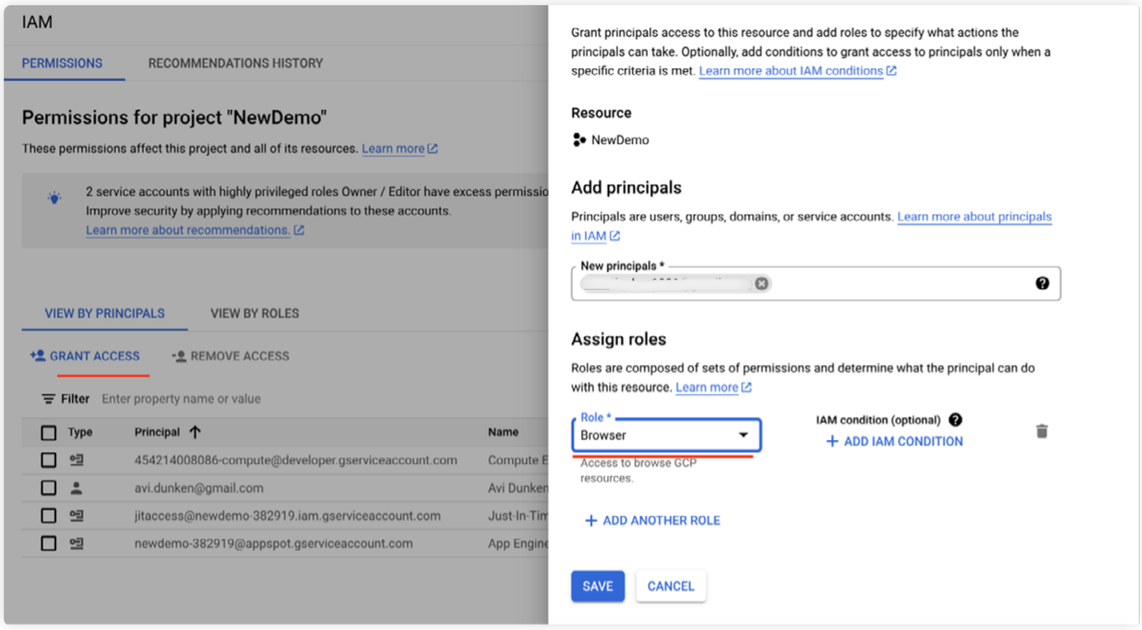
ADD ANOTHER ROLE

SAVE TEST CHANGES CANCEL

Testing JIT Access

You’ll now test the process by granting eligible access and activating it using the JIT Access app.

As before, click **GRANT ACCESS** on the IAM page in the GCP console. You’ll be prompted to enter an email address. Use your Google Workspace user or a second cloud identity, and expand the drop-down menu labeled **Select a role** to select **Project > Browser**:



IAM

PERMISSIONS RECOMMENDATIONS HISTORY

Permissions for project "NewDemo"

These permissions affect this project and all of its resources. Learn more

2 service accounts with highly privileged roles Owner / Editor have excess permissio
Improve security by applying recommendations to these accounts.
Learn more about recommendations.

VIEW BY PRINCIPALS VIEW BY ROLES

GRANT ACCESS REMOVE ACCESS

Filter Enter property name or value

Type	Principal	Name
	454214008086-compute@developer.gserviceaccount.com	Compute E
	avi.dunken@gmail.com	Avi Dunken
	jitaccess@newdemo-382919.iam.gserviceaccount.com	Just-In-Tim
	newdemo-382919@appspot.gserviceaccount.com	App Engin

Grant principals access to this resource and add roles to specify what actions the principals can take. Optionally, add conditions to grant access to principals only when a specific criteria is met. Learn more about IAM conditions

Resource

NewDemo

Add principals

Principals are users, groups, domains, or service accounts. Learn more about principals in IAM

New principals * [redacted]

Assign roles

Roles are composed of sets of permissions and determine what the principal can do with this resource. Learn more

Role * Browser

Access to browse GCP resources.

IAM condition (optional)

ADD IAM CONDITION

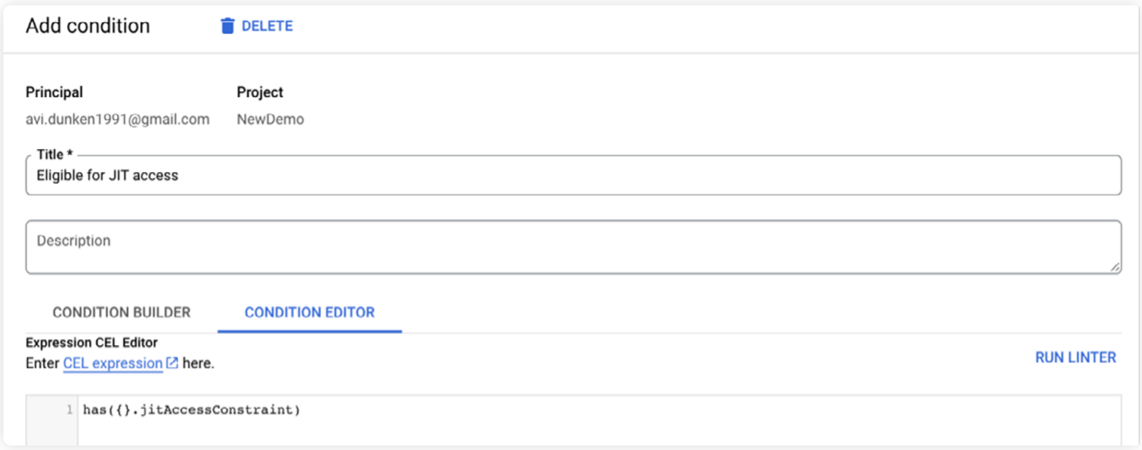
ADD ANOTHER ROLE

SAVE CANCEL

Then, click **ADD IAM CONDITION**, enter an email ID in the **New principal** input field, and under **Role**, choose **Browser**.

Add a descriptive title for the condition, such as “Eligible for JIT access.” Next, click **CONDITION EDITOR** and enter the CEL expression below:

```
has({}.jitAccessConstraint)
```



Add condition DELETE

Principal: avi.dunken1991@gmail.com

Project: NewDemo

Title *: Eligible for JIT access

Description

CONDITION BUILDER CONDITION EDITOR

Expression CEL Editor

Enter CEL expression here.

RUN LINTER

```
has({}.jitAccessConstraint)
```

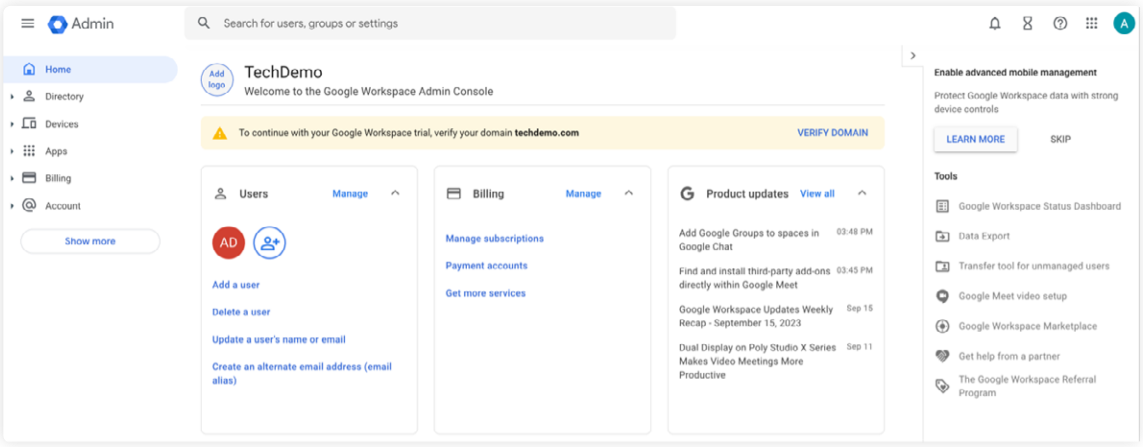
Click **SAVE**. Based on this condition you just added, eligible access will be granted for the JIT Access application.

Requesting JIT Access Using the JIT Access App

You can now try changing to a different user and requesting temporary access to a resource.

Go to your JIT Access app's URL in an incognito browser window. If you didn't take note of it before, simply replace `newdemo-382919` with your project name in `https://newdemo-382919.el.r.appspot.com`.

Sign in with the user that you just granted the access to:



TechDemo
Welcome to the Google Workspace Admin Console

To continue with your Google Workspace trial, verify your domain **techdemo.com** VERIFY DOMAIN

Users — Manage
Add a user
Delete a user
Update a user's name or email
Create an alternate email address (email alias)

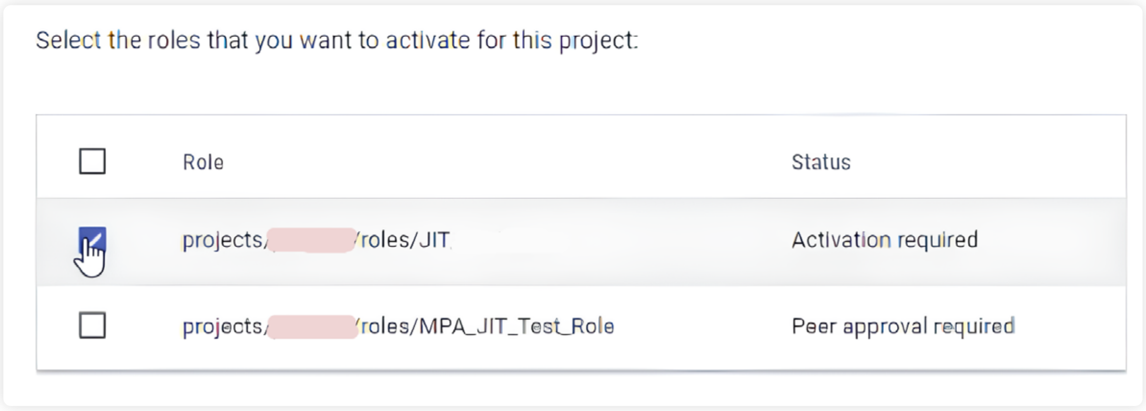
Billing — Manage
Manage subscriptions
Payment accounts
Get more services

Product updates — View all
Add Google Groups to spaces in Google Chat — 03:48 PM
Find and install third-party add-ons directly within Google Meet — 03:45 PM
Google Workspace Updates Weekly Recap - September 15, 2023 — Sep 15
Dual Display on Poly Studio X Series Makes Video Meetings More Productive — Sep 11

Enable advanced mobile management
Protect Google Workspace data with strong device controls
LEARN MORE SKIP

Tools
Google Workspace Status Dashboard
Data Export
Transfer tool for unmanaged users
Google Meet video setup
Google Workspace Marketplace
Get help from a partner
The Google Workspace Referral Program

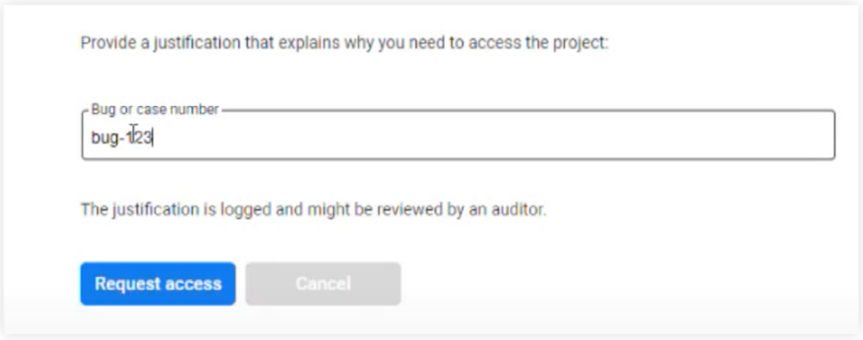
Select a role for which you want to activate access in the JIT Access application:



Select the roles that you want to activate for this project:

☐	Role	Status
☑	projects/[redacted]/roles/JIT	Activation required
☐	projects/[redacted]/roles/MPA_JIT_Test_Role	Peer approval required

Provide a justification and click **Request access**:



Provide a justification that explains why you need to access the project:

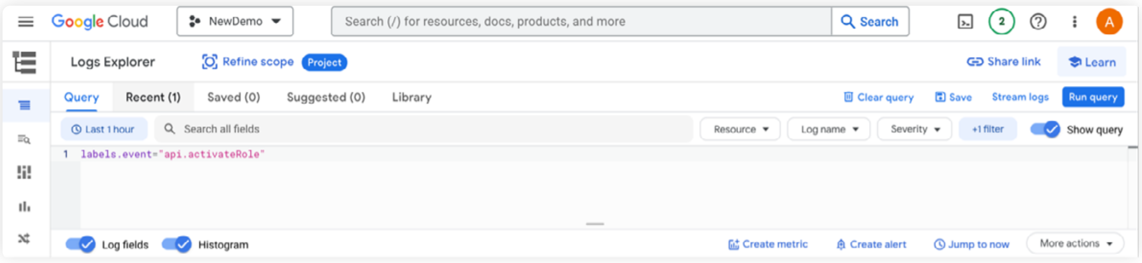
Bug or case number: bug-123

The justification is logged and might be reviewed by an auditor.

Return to your browser window where the administrative user is logged in and review the log.

Go to Logs Explorer in the Logging section of your GCP console and enable the **Show query** option.
Then, enter the query below:

```
labels.event=”api.activateRole”
```



```
labels.event="api.activateRole"
```

Clicking **Run query** should produce an output similar to the following:

~~~
```json
{
“textPayload”: “User test123@gmail.
com activated role ‘ROLE’ on ‘//
cloudresourcemanager.googleapis.com/projects/
newdemo-382919’ for themselves”,
“severity”: “INFO”,
“labels”: {
       “resource”: “//cloudresourcemanager.
googleapis.com/projects/newdemo-382919”,
       “event”: “api.activateRole”,
       “role”: “ROLE”,
       “clone_id”: “00d8...”,
       “user”: “test123@gmail.com”,
       “justification”: “testing”,
  },
}
~~~

You can see that there’s now a log record for the activated role.

You’ve now successfully set up and tested JIT access in Google Cloud Platform.

Tips to Optimize Access Security in GCP

Configuring JIT access is just one part of ensuring access security in GCP. Below are some general tips for enhancing access security in GCP.

Implement Multifactor Authentication

Enabling multifactor authentication (MFA) adds another level of protection for user accounts in GCP by requiring users to supply two or more authentication factors, such as a password and a time-based, one-time password generated on the user's mobile phone. MFA significantly lowers the risk of unauthorized access, even when login credentials become compromised.

Use an Identity and Access Management Solution

Identity and access management solutions provide robust features to simplify user access, roles, and permissions in GCP. ConductorOne, for instance, guarantees enhanced security for GCP with minimal effort on your end. Its centralized control, audit trails, and automation features effectively uphold access security.

Enable Audit Logging and Monitoring to Track and Detect Suspicious Activities

GCP offers comprehensive audit logging and monitoring capabilities. By activating these features, you can quickly observe user activities, API usage, system events, and any suspicious activity, allowing you to quickly respond to threats.

Utilize Service Accounts and the Principle of Least Privilege

By allocating permissions using service accounts and adhering to the principle of least privilege, users and applications will only have the permissions necessary for performing their tasks, which reduces accidental data exposure or misuse of resources.

Implement VPC Service Controls to Secure Data within GCP Services

VPC Service Controls provide an additional layer of security by allowing you to define security perimeters for specific Google Cloud services. This helps prevent data exfiltration and unauthorized access to resources.

Use Cloud Identity-Aware Proxy (IAP) for Fine-Grained Access Control to Applications

IAP provides context-aware access control for web applications running on GCP. You learned how to configure IAP earlier in this article. Users are granted access based on their identity and context, providing increased security without needing VPNs or complex firewall rules.

Implement Strong and Regularly Rotated Encryption Keys for Data Protection

Data encryption is vital for access security. Implement strong encryption keys and regularly rotate them to protect data during transit and at rest. You can rotate your keys manually or set up a schedule using GCP's Cloud Key Management service.

Follow Security Best Practices for Virtual Machine Instances, Containers, and Serverless Functions

GCP compute resources require their own set of security considerations to prevent vulnerabilities from emerging and secure access. GCP has some recommended best practices for protecting virtual machines, containers, and serverless functions to minimize vulnerabilities while safeguarding access.

Regularly Review and Update Access Permissions

Security should be an ongoing process, and access permissions should be regularly evaluated and updated. Remove unnecessary permissions, make sure roles reflect job responsibilities, and ensure your security policies align with evolving organizational needs and best practices.

Conclusion

You now know how to implement JIT access in GCP. JIT access within the Google Cloud Platform represents an exciting paradigm shift in cloud security. By only granting access when needed and following the principle of least privilege, this approach significantly enhances organizations operating within the platform's security posture—not simply protecting access but securing it intelligently.

ConductorOne plays an essential part in access control by helping to implement security best practices. For instance, ConductorOne allows you to implement JIT access for GCP resources, enforce the least privilege principle for users with GCP accounts, and implement risk-based authentication for these accounts. These practices can help reduce the risks of data breaches or unauthorized access to GCP resources.