Host Defender Auto Deployment from SaaS based Prisma Cloud User Guide

Introduction

Prisma Cloud provides comprehensive visibility and threat detection for cloud workload in Google Cloud. Prisma Cloud software consists of two components: Console and Defender. Console is Prisma Cloud's management interface. It lets you define policy and monitor your environment. For the Prisma Cloud SaaS edition, the Console is hosted by Palo Alto Networks. Defender is deployed to Google Cloud environment to secure the cloud workload. Defender protects your environment according to the policies set in Console. There are a number of Defender types, Host Defender utilizes Prisma Cloud's model-based approach for protecting hosts that do not run containers.

Host Defender Auto Deployment allows Prisma Cloud customers to deploy Prisma Cloud Host Defender (Security agent) from SaaS based Prisma Cloud Console to the virtual machines (VM) aka compute engine instances in your Google Cloud project automatically. Google Cloud Guest Policy manages Host Defender Auto Deployment to VMs. You can choose the target VMs based on the Guest Policy Assignment. The auto deployment use two of Google Cloud Guest Policy Assignments to allow you to target a group of VMs by using one of the following characteristics:

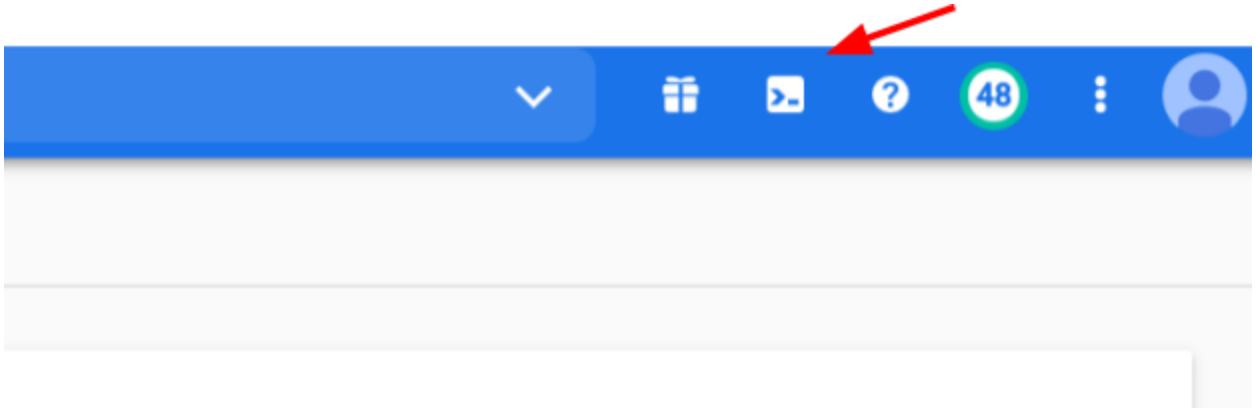
- Instance name prefix. See example 2.
- Instance label. See example 3.

Guest Policy supported various versions of Linux and Window OS, you can find a completed list of OS at LINK,

Prepare your Google Cloud Project

You will need to prepare your Google Cloud Project with required API, Service Account and Secret before launching the Host Defender Auto Deployment.

In the upper-right corner of the Cloud Console, click the Cloud Shell icon to open the Cloud Shell.



Start by setting a variable for your project ID with this command:

```
export project_id=[YOUR PROJECT ID]
```

Verify the success of your variable creation by echoing the value:

```
echo $project_id
```

Next, create an environment variable for your username:

```
export user_id=[YOUR USER ID]
```

Verify the success of your variable creation by echoing the value:

```
echo $user_id
```

Select your account and project

```
gcloud config set account $user_id
gcloud config set project $project_id
```

Permission for creating Guest Policy

Owners of a project have full access to create and manage policies. For all other users, you need to grant GuestPolicy Admin permission for managing the Guest Policies with the following gcloud command.

- GuestPolicy Admin (`roles/osconfig.guestPolicyAdmin`). Contains permissions to create, delete, update, get, and list guest policies.

```
gcloud projects add-iam-policy-binding $project_id \
    --member user:$user_id \
    --role roles/osconfig.guestPolicyAdmin
```

Enable the OS Config API in your project

```
gcloud services enable osconfig.googleapis.com
```

Configure the project metadata

```
gcloud compute project-info add-metadata \
--metadata=enable-guest-attributes=true,enable-osconfig=true,enable-os-config-debug=true,
osconfig-log-level=debug
```

```
Your Cloud Platform project in this session is set to paloaltonetworksgcp-public.
Use "gcloud config set project [PROJECT_ID]" to change to a different project.
myan@cloudshell:~ (paloaltonetworksgcp-public)$ gcloud services enable osconfig.googleapis.com
myan@cloudshell:~ (paloaltonetworksgcp-public)$ gcloud compute project-info add-metadata  \
>    --metadata=enable-osconfig=true
Updated [https://www.googleapis.com/compute/v1/projects/paloaltonetworksgcp-public].
myan@cloudshell:~ (paloaltonetworksgcp-public)$
```

Run the following command to confirm the project metadata is setup properly

```
gcloud compute project-info describe
```

```
Use "gcloud config set project [PROJECT_ID]" to change to a different project.
myan@cloudshell:~ (paloaltonetworksgcp-public)$ gcloud compute project-info describe
commonInstanceMetadata:
  fingerprint: Qwx3Im4G9UI=
  items:
  - key: enable-guest-attributes
    value: 'TRUE'
  - key: enable-osconfig
    value: 'true'
  - key: gke-cluster-daniel1-b25f0804-secondary-ranges
    value: services:default:default:gke-cluster-daniel1-services-b25f0804,pods:default:default:gke-cluster-daniel1-pods-b25f0804
  - key: gke-cluster-test-pcc-mp-12e0fb96-secondary-ranges
    value: services:default:default:gke-cluster-test-pcc-mp-services-12e0fb96,pods:default:default:gke-cluster-test-pcc-mp-pods-12e0fb96
  - key: osconfig-log-level
    value: debug
```

Deploy Token Refresher

The Prisma Cloud Compute API Token used to securely retrieve software is valid up to 60 minutes. The Host Defender Auto Deployment deploys Prisma Cloud Host Defender (Security agent) from Prisma Cloud Console to the virtual machines (VM) aka compute engine instances in your Google Cloud project per the guess policy you configure. This process requires a valid Prisma Cloud Compute API Token from time to time to ensure the automatic deployment of Host Defender when new compute instances spin up. To address this you will deploy a Token Refresher script to keep the token valid.

The Token Refresher script is a community supported script that refreshes the Prisma Cloud Console token associated with cert download for auto-install of defender agent.

Prepare for Token Refresher

Token Refresher refreshes the token through refreshing a secret stored in Secret Manager.

Enable the Following APIs using the gcloud command below:

1. Compute Engine API
2. Cloud Functions API
3. Cloud Logging API
4. Cloud Pub/Sub API
5. Cloud Build API (required by GCP for the Functions API)
6. Cloud Scheduler API
7. Cloud Secretes Manager
8. Cloud Storage API (Different than Cloud Storage)

```
gcloud services enable compute.googleapis.com cloudfunctions.googleapis.com
logging.googleapis.com pubsub.googleapis.com cloudscheduler.googleapis.com
cloudbuild.googleapis.com storage.googleapis.com secretmanager.googleapis.com
```

Create a Service Account for the Token Refresher script.

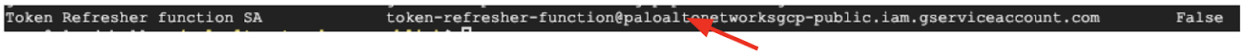
Enter the following command in Google Cloud Shell and select “Enter”. This will create a service account named “Token-Refresher-Function” and assign the service account a description of “Token Refresher function SA”

NOTE: If you get a request to authorize cloud shell to execute this script, click “Authorize.”

```
gcloud iam service-accounts create token-refresher-function \
 --description "Token Refresher function SA" \
 --display-name="Token Refresher function SA"
```

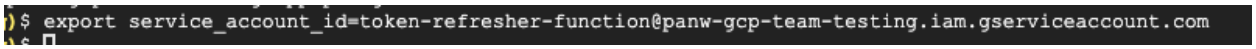
Verify the service account is created. Run the following command and locate the service account:

```
gcloud iam service-accounts list
```



From the cloudshell output, copy the full email ID of your newly created service account, and add it to an environment variable with the following command:

```
export service_account_id=[FULL SERVICE ACCOUNT EMAIL ID]
```



The Token Refresher script will require access to a storage bucket to be created in a later step. To grant permission for the Token Refresher script to access secretmanager admin role to the function service account use this gcloud command:

```
gcloud projects add-iam-policy-binding $project_id \
--member serviceAccount:$service_account_id \
--role roles/secretmanager.admin
```

Operational Steps Outline

1. Cloud Scheduler is a Cron tool that utilizes AppEngine to process Google-managed scheduled events. The Cloud Scheduler job will be used to trigger the Pub/Sub topic that will trigger a Cloud Function to refresh the API Token.
2. The Pub/Sub Topic is the trigger that activates the Cloud Function.
3. The Cloud Function runs the Python code to refresh the token by running a “requests.get” against the token endpoint on your Prisma instance.
4. The Prisma instance returns the token to the Cloud Function.

All of these steps will need to be completed by an individual with GCP organization admin or project admin rights plus token management access to the Prisma Cloud Compute Console.

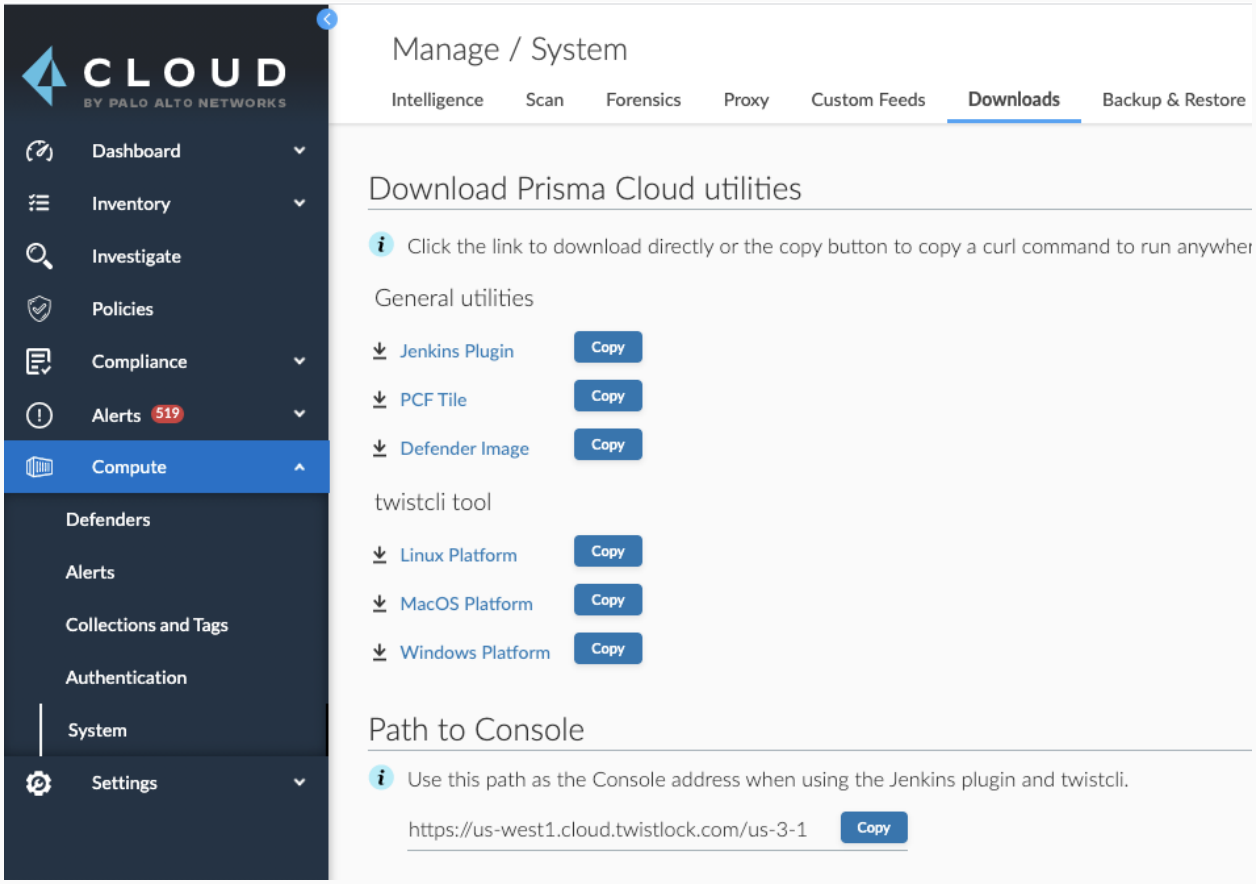
Retrieve Prisma Cloud Compute Console URL and Token

1. Login to your Prisma Cloud Console and get the access information:
    a. The URL to access Prisma Cloud Console

    Prisma Cloud > Compute > Manage > System, select **Downloads** tab, locate “**Path to Console**” at the bottom of the page, click **copy** button and paste to your notepad. The URL should be in this format:

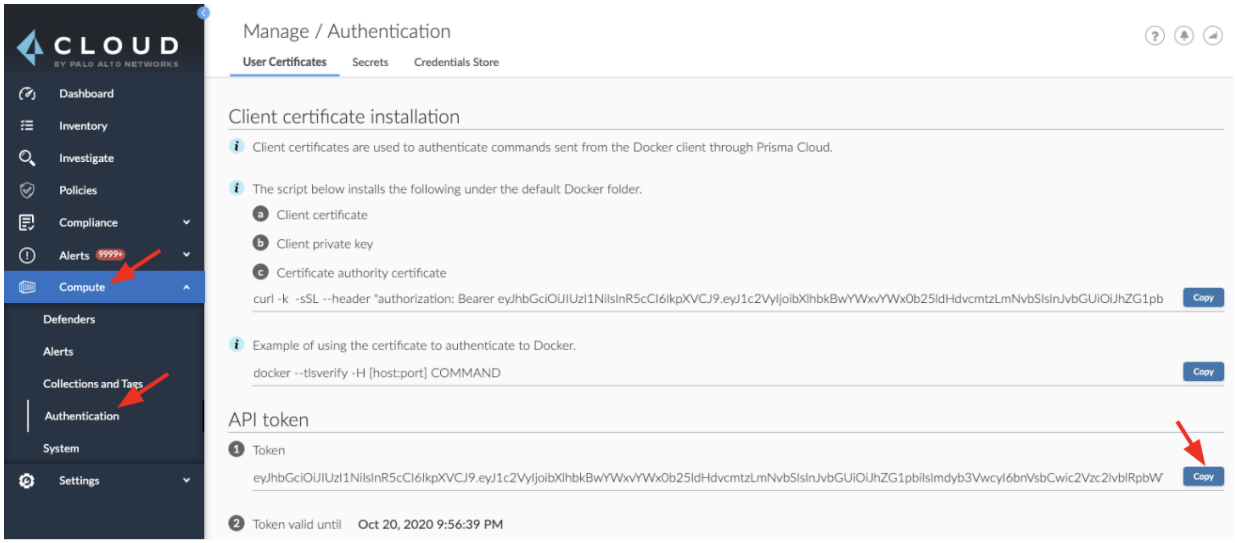
    https://us-west1.cloud.twistlock.com/us-3-159237196

    where “us-3-159237196” is the tenant ID. You will need the console address to configure the Cloud Function in a later step.



b. Authentication API Token

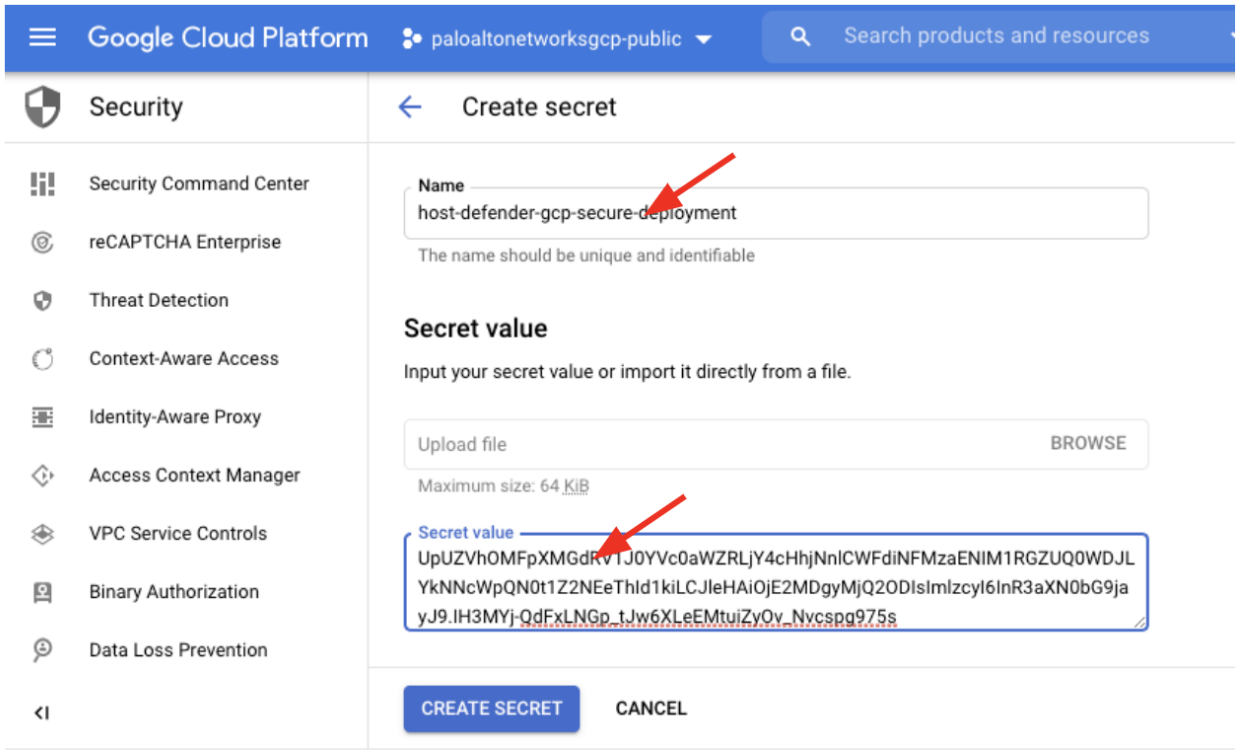
Prisma Cloud > Compute > Manage > Authentication, copy API Token



Create Secret for Prisma Cloud Token.

Navigate to the GCP Secrete Manager in the GCP Console, create a secret with the name **host-defender-gcp-secure-deployment**, the value is the token you copied in the previous step.

Note: You must use the exact secret name above, or the auto deployment would fail.



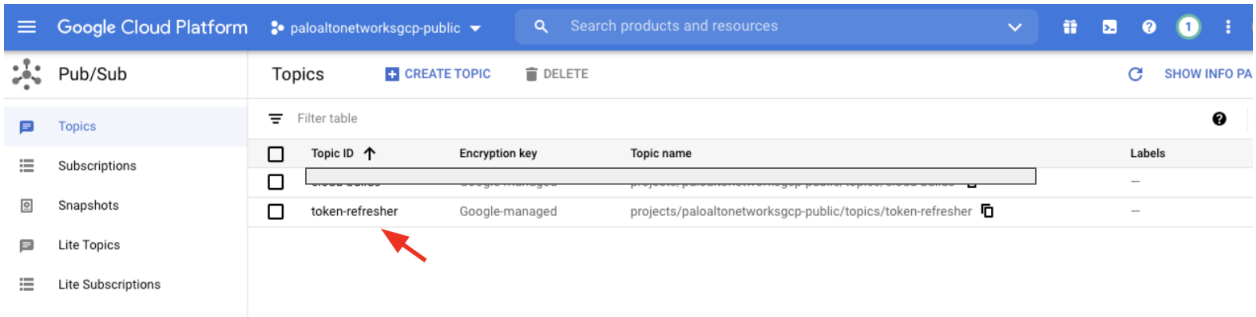
Create a Pub/Sub Topic

Now that you have the service account for your Cloud Function, you'll need to create a Pub/Sub Topic that acts as the trigger for the function.

```
gcloud pubsub topics create token-refresher
```

```
myan@cloudshell:~/host-defender-token (paloaltonetworksgcp-public)$ gcloud pubsub topics create token-refresher
Created topic [projects/paloaltonetworksgcp-public/topics/token-refresher].
```

From the hamburger menu at the top left, select Pub/Sub > Topics, and verify that your topic is listed.



Create a GCP Function

Clone the repository containing the Token Refresher source code:

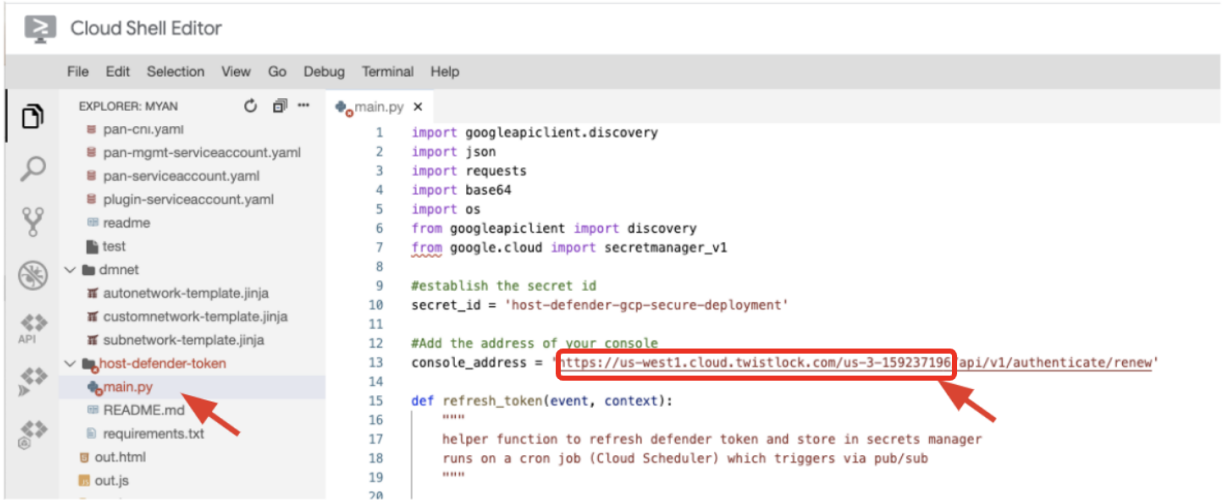
```
git clone https://github.com/PaloAltoNetworks/host-defender-token-refresher.git
cd host-defender-token-refresher
```

If you'd like, spend some time looking over the code. A Cloud Function requires two main code files - main.py, requirements.txt. Google Cloud has API-specific Python libraries to interact with their services. You use the requirements.txt to add the required libraries and APIs reference.

Configure Cloud Function

The cloud function will use a python script named main.py. We will update main.py script with your Prisma Cloud Compute Console **URL**. Make sure to replace the URL only and leave the rest in place.

Open the Editor  and navigate to main.py under host-defender-token folder



Replace the **Console Address** with the URL you copied from Prisma Cloud Console.

Run the following command to deploy the Cloud Function:

```
gcloud functions deploy pcc-token-refresher --region "us-central1"
--trigger-topic=token-refresher --entry-point=refresh_token --runtime=python37
--service-account $service_account_id
```

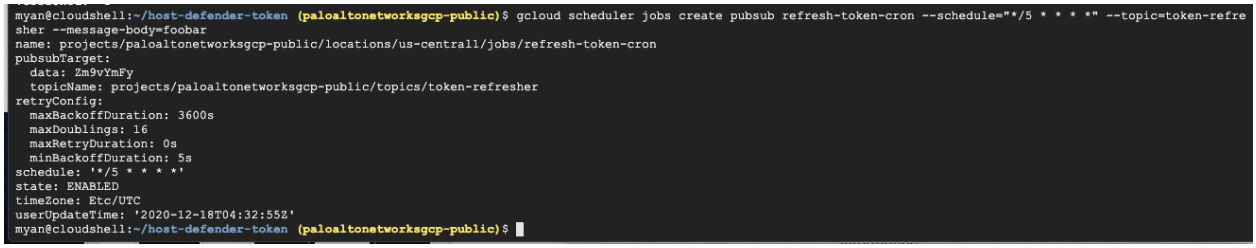
It may take several minutes for your Cloud Function to complete deploying. Important pieces of the command above are the entry point, which simply means the first function that should run during the operation; your service account; and trigger topic, which is the PubSub topic created previously.

```
myan@cloudshell:~/host-defender-token (paloaltonetworksgcp-public)$ gcloud functions deploy pcc-token-refresher --region "us-central1" --trigger-topic=token-refresher
--entry-point=refresh_token --runtime=python37 --service-account $service_account_id
Created .gcloudignore file. See `gcloud topic gcloudignore` for details.
Deploying function (may take a while - up to 2 minutes)...⠹
For Cloud Build Stackdriver Logs, visit: https://console.cloud.google.com/logs/viewer?project=paloaltonetworksgcp-public&advancedFilter=resource.type%3Dbuild%0Aresourc
e.labels.build_id%3D77f039e6-258a-4f7d-807d-81759986c58c%0AlogName%3Dprojects%2Fpaloaltonetworksgcp-public%2Flogs%2Fcloudbuild
Deploying function (may take a while - up to 2 minutes)...done.
availableMemoryMb: 256
buildId: 77f039e6-258a-4f7d-807d-81759986c58c
entryPoint: refresh_token
eventTrigger:
  eventType: google.pubsub.topic.publish
  failurePolicy: {}
  resource: projects/paloaltonetworksgcp-public/topics/token-refresher
  service: pubsub.googleapis.com
ingressSettings: ALLOW_ALL
labels:
  deployment-tool: cli-gcloud
name: projects/paloaltonetworksgcp-public/locations/us-central1/functions/pcc-token-refresher
runtime: python37
serviceAccountEmail: token-refresher-function@paloaltonetworksgcp-public.iam.gserviceaccount.com
sourceUploadUrl: https://storage.googleapis.com/gcf-upload-us-central1-14b8fc50-d3f9-4389-a84d-a1af942e607c/7ee90317-30cc-4f5f-913b-d81cb8af6346.zip?GoogleAccessId=ser
vice-1026607622258@gcf-admin-robot.iam.gserviceaccount.com&Expires=1608266949&Signature=EdX7ssDGVuGj1ec91Bmxn521kcwByvIejKGg42f1jCgoMHGqAleNr2RK1Rp6JduOd%2F3DMyXRQ5SxO
TFolG24zi6faLzpdzFGo4%2FOii3IoKObyFsOxXmTcteILHuMiXWnqjMv0iTeVNDxVzDbz3zsuiB%2F7BHn65LH3Jz51kKF1CJ0vaJib3vVIv%2BRMo3jGrwi39pnaR8tjHP%2B8Hs%2B7GY4yVJdJbYqIcGsQaUWg3Adux
DCD2ek2TNvx9zBGYfrc8gD10fKuUohzMcErOTvlxl45s0pGmN%2BHtaytSkUV%2FMrvBRylCEvQ%2B3XEJOKm5ww4f%2FuZxUG7PhE1FFfW1uGUsIsXQ%3D%3D
status: ACTIVE
timeout: 60s
updateTime: '2020-12-18T04:21:15.929Z'
versionId: '1'
```

Create cron job via Cloud Scheduler

Cloud Scheduler is a Cron tool that utilizes AppEngine to process Google-managed scheduled events. The Cloud Scheduler job will be used to trigger the Pub/Sub topic that acts as the trigger for the Cloud Function.

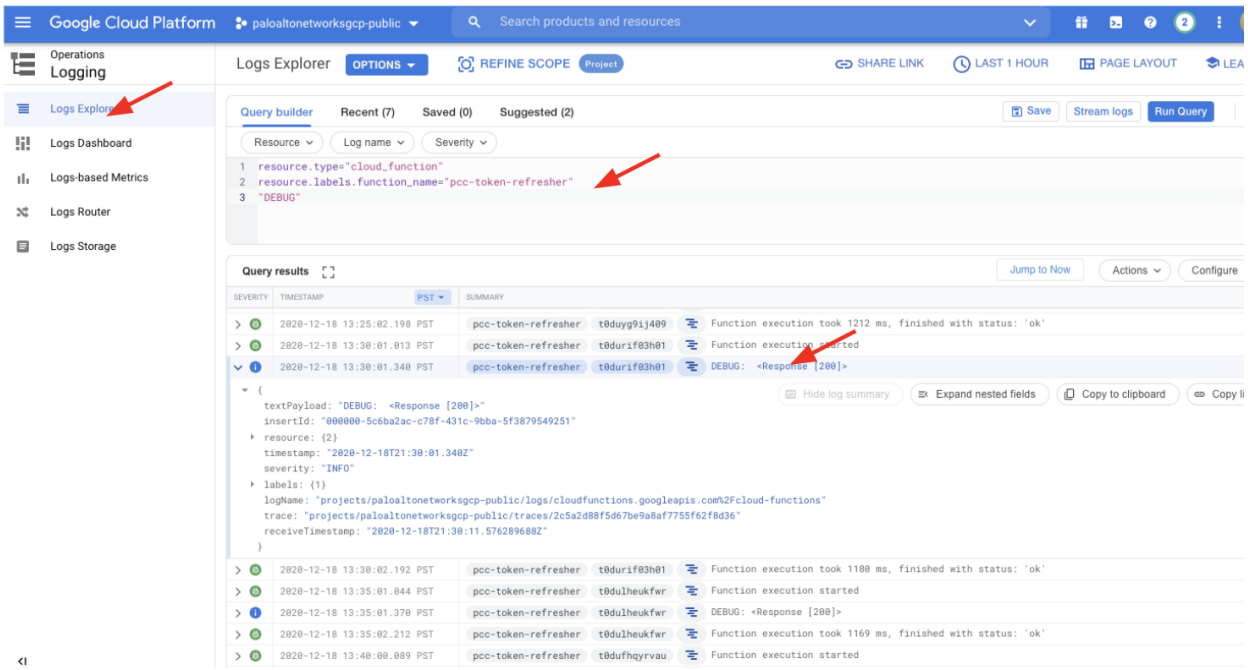
```
gcloud scheduler jobs create pubsub refresh-token-cron --schedule="*/5 * * * *"
--topic=token-refresher --message-body=foobar
```



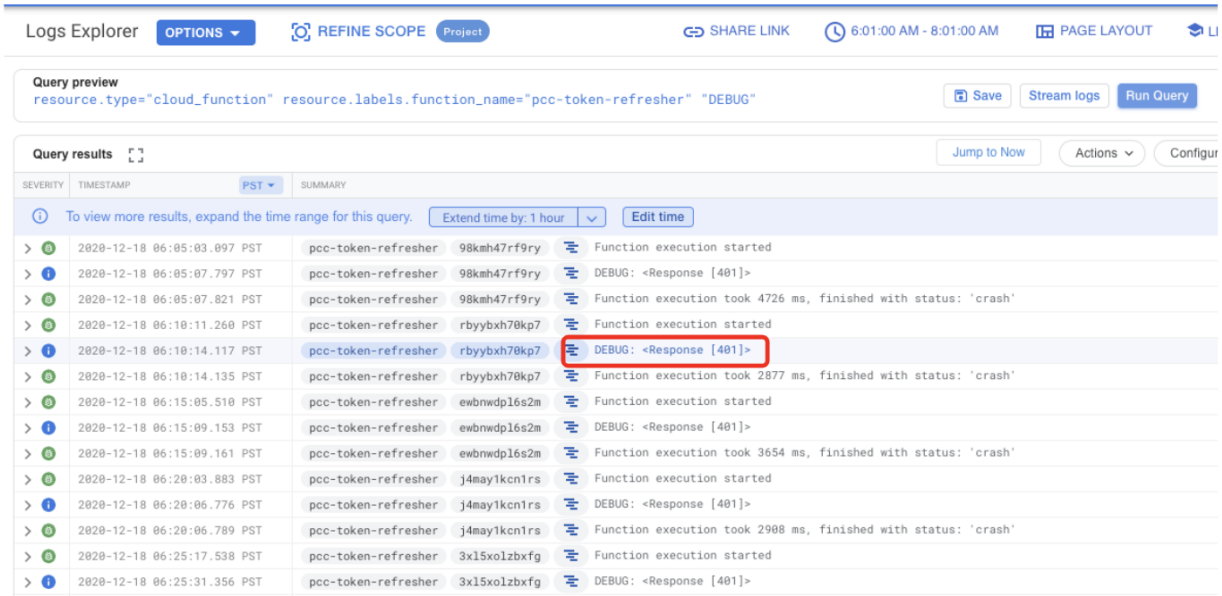
Verify the token is refreshing

Navigate to **Logging > Log Explorer,** enter the following into the **Query builder** and click **Run Query**. Locate “DEBUG: <Response [200]>”, it is the indicator of a success run of "**pcc-token-refresher**”

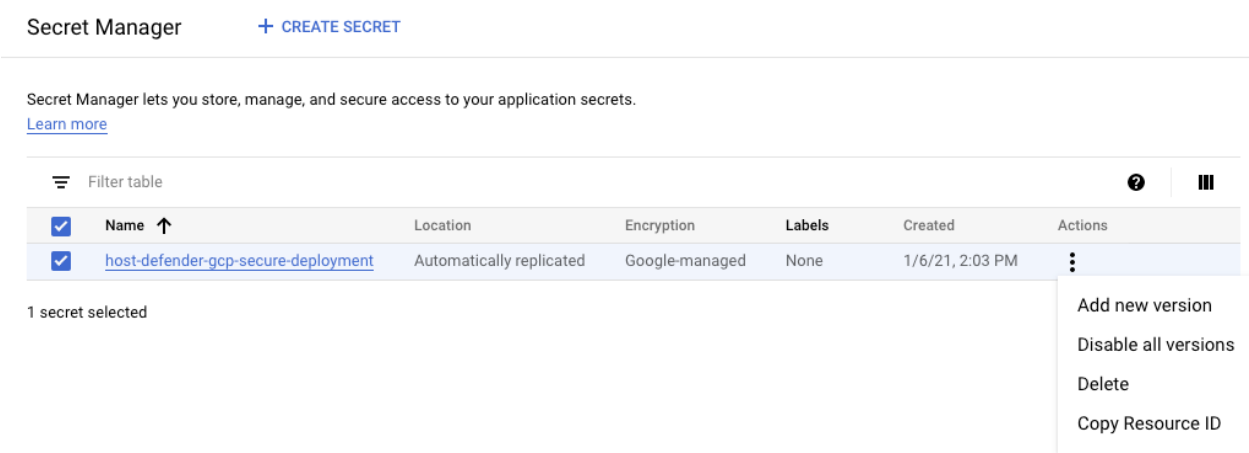
```
resource.type="cloud_function"
resource.labels.function_name="pcc-token-refresher"
"DEBUG"
```



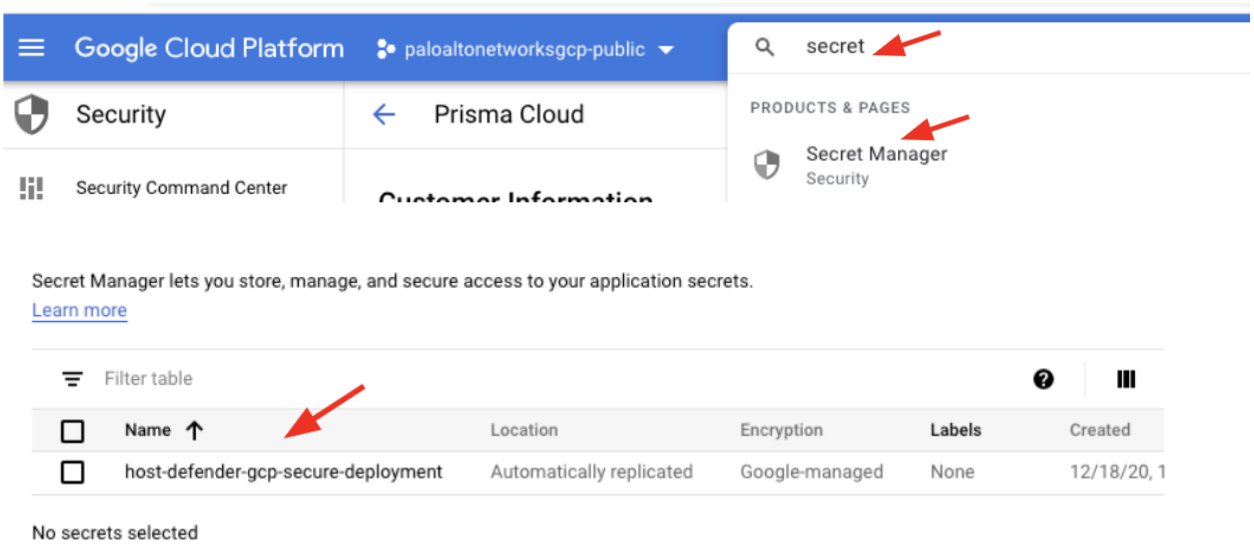
Debug response code 401 indicates a failed Cloud Function run.



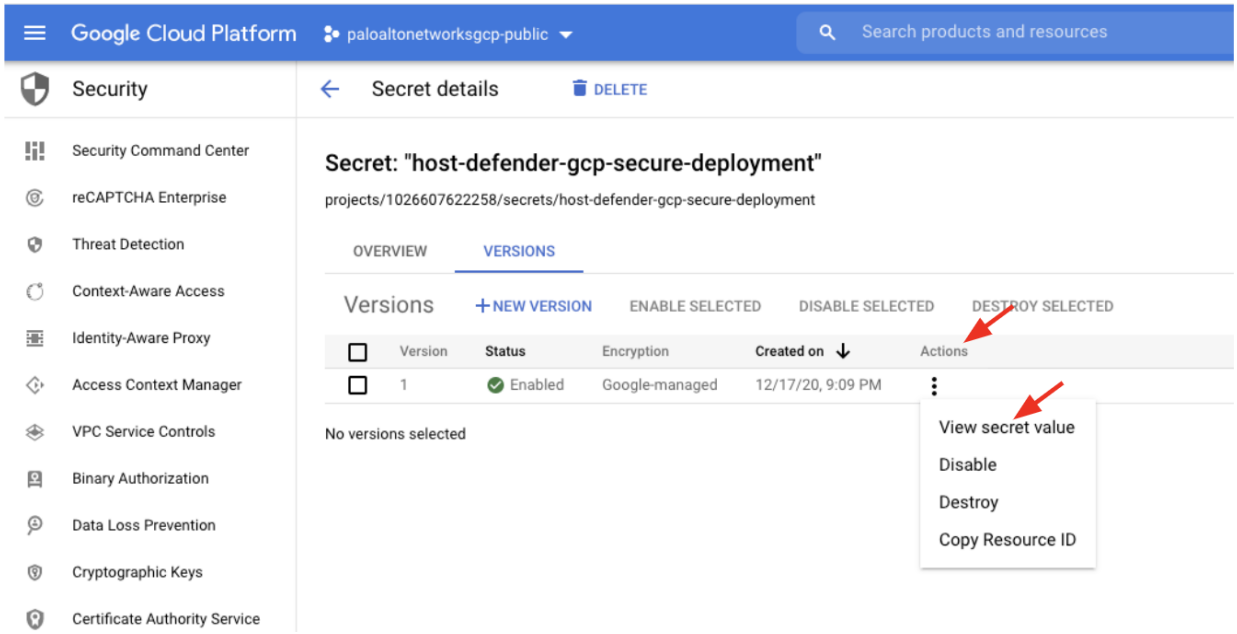
Delete the secret **host-defender-gcp-secure-deployment** and recreate it with updated token from Prisma Cloud

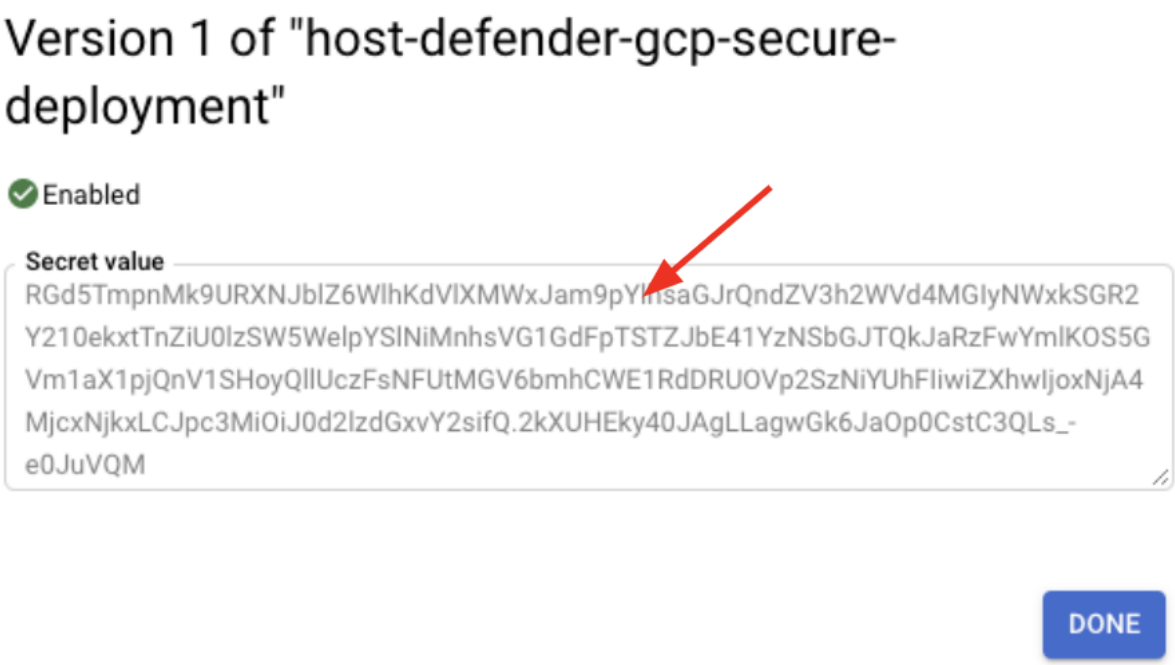


You may also see the secrets get updated every 5 minutes at Security Manager



Select secret "**host-defender-gcp-secure-deployment**", click **Action** and **View secret value**





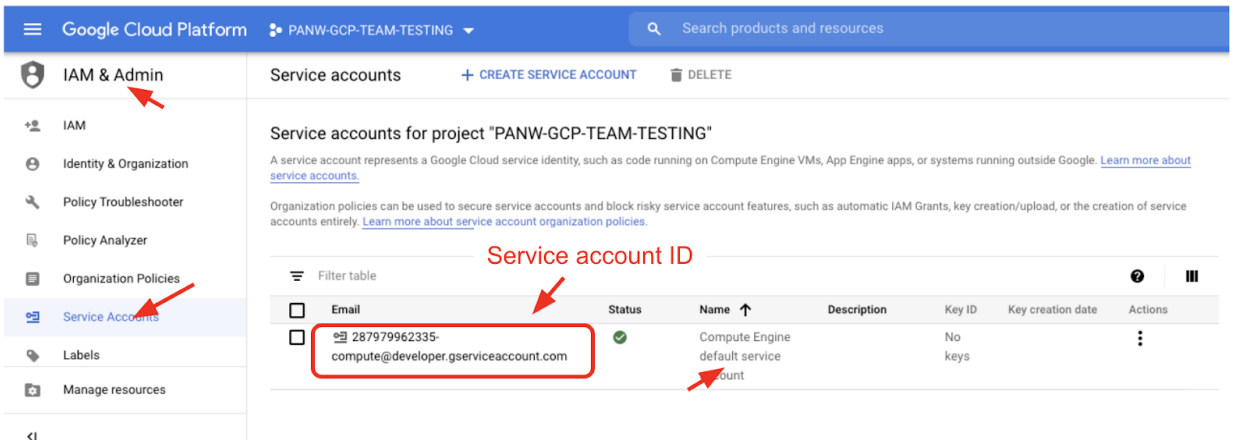
Confirm that the secret value has been updated with a new token.

Deploy Host Defender from Marketplace

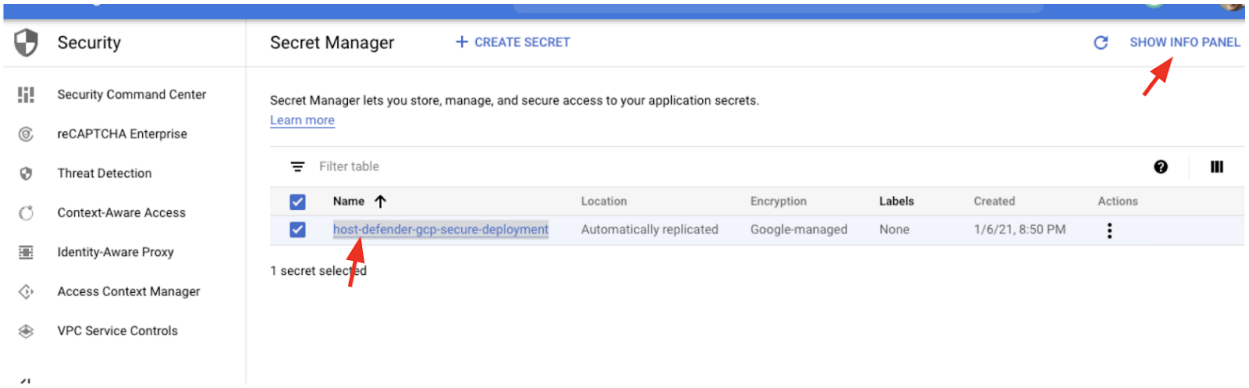
Prepare Compute Instance

VM instances need the permission to access the secret **host-defender-gcp-secure-deployment** you created in the section above. You will add the Secret Admin role to the service account you use for VM instances. The following is using the default service account, you may use another service account.

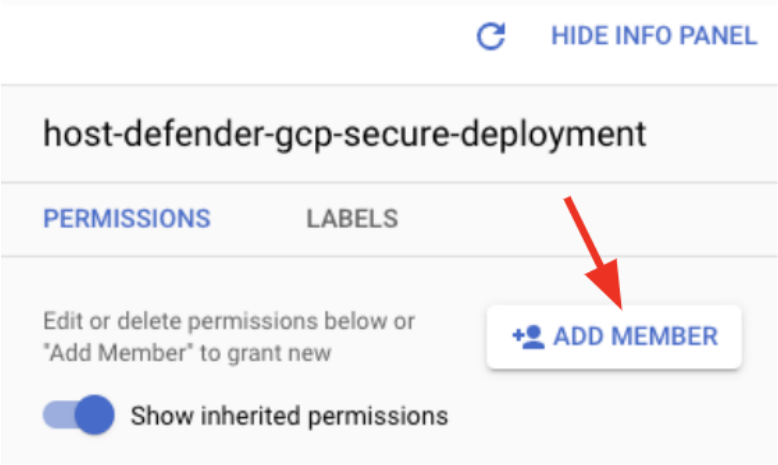
Navigate to IAM & Admin > Service accounts, locate the default service account for VM instance



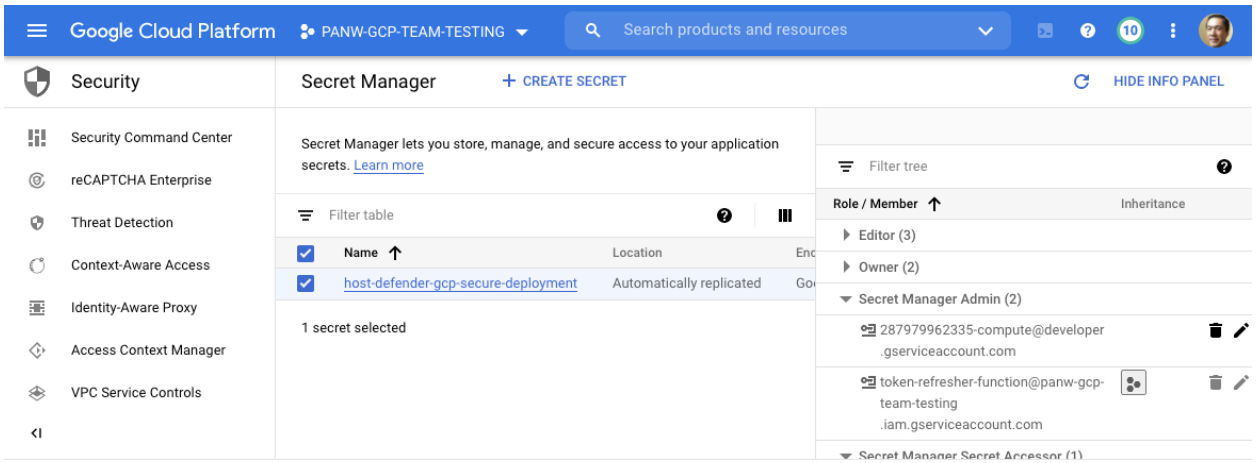
Navigate to Secret Manager, select the secret **host-defender-gcp-secure-deployment,** click show info panel at up right corner



Click **Add Member**



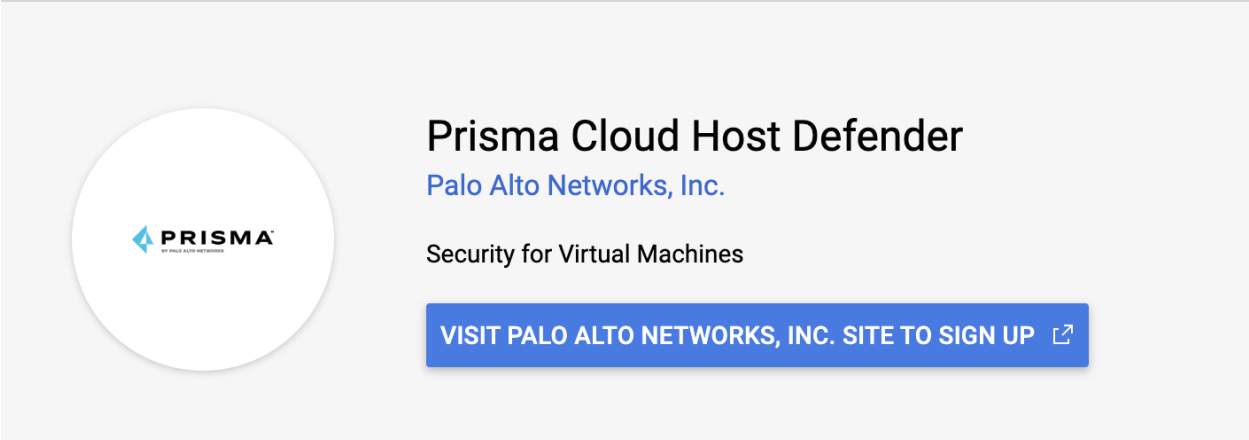
Add the Compute Engine service account under New members. Select a role of **Secret Manager Admin** and **Save**



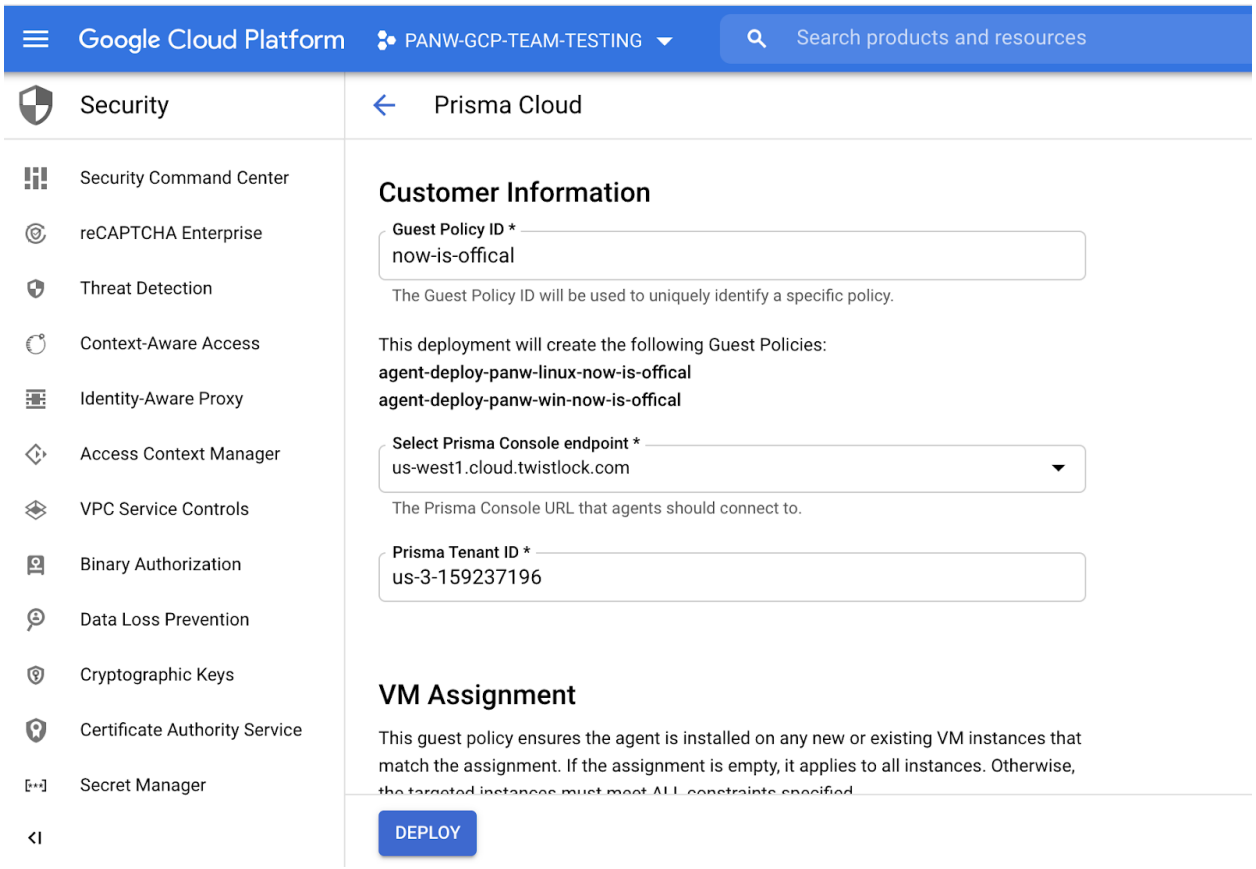
Deploy Host Defender

Create a Guest Policy

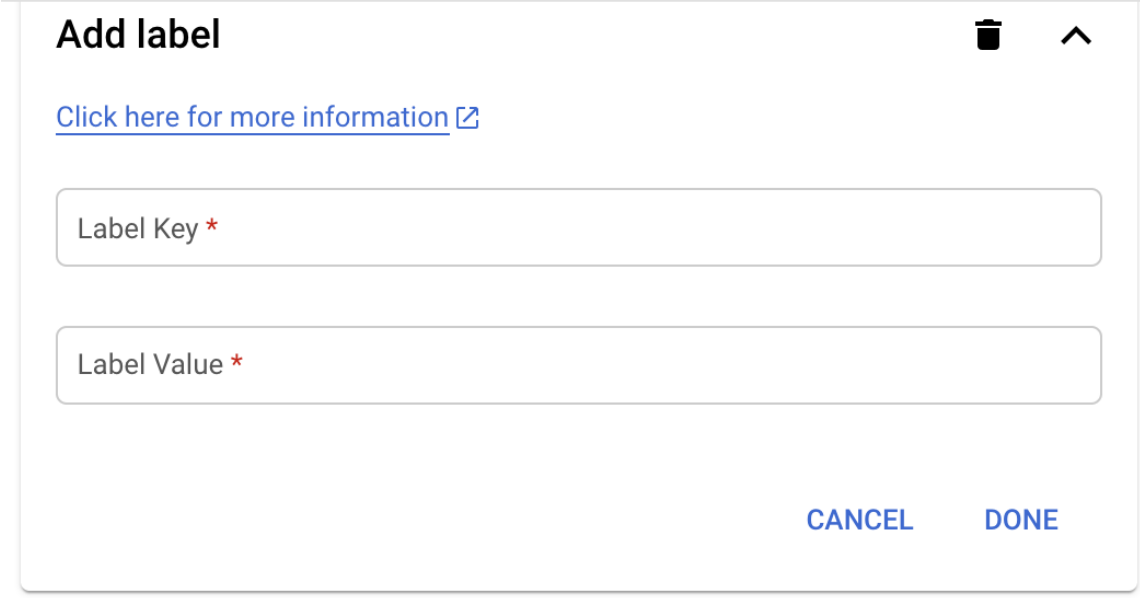
Navigate to marketplace and search for Prisma Cloud Host Defender

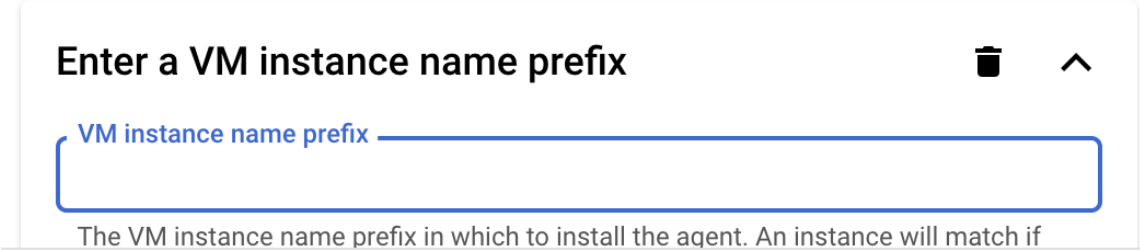


Enter the name of the Guest Policy ID you would like to use. Select Prisma Console endpoint and Prisma Tenant ID (you can find the information at your **Prisma Cloud Console**, navigate to **Compute > Manage > System** you did in a previous step).

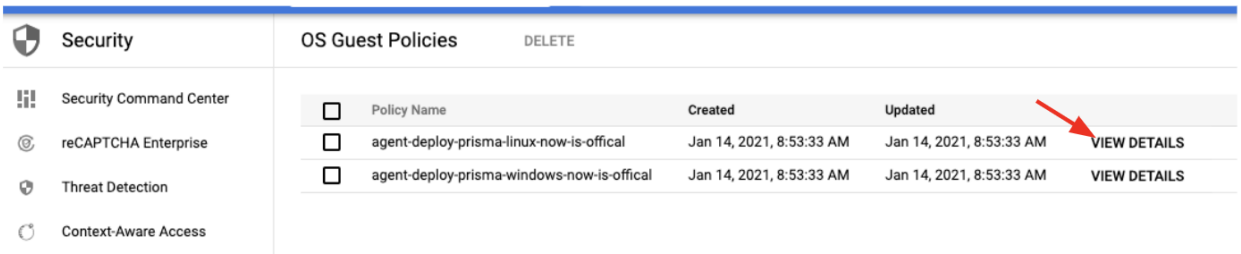


You may choose Add label or add VM name prefix to your guest policy. Refer to the Google Cloud link for more details.





Click **Deploy,** it will bring you to the Guest Policy page. You may click **VIEW DETAILS** at right to view the details of the guest policy created.



Review the guest policy status with gcloud command

You may use the gcloud command-line tool to inspect the guest policies associated with your project and the compute instance.

1. Use the os-config guest-policies list command to list all your guest policies.

```
gcloud beta compute os-config guest-policies list
```

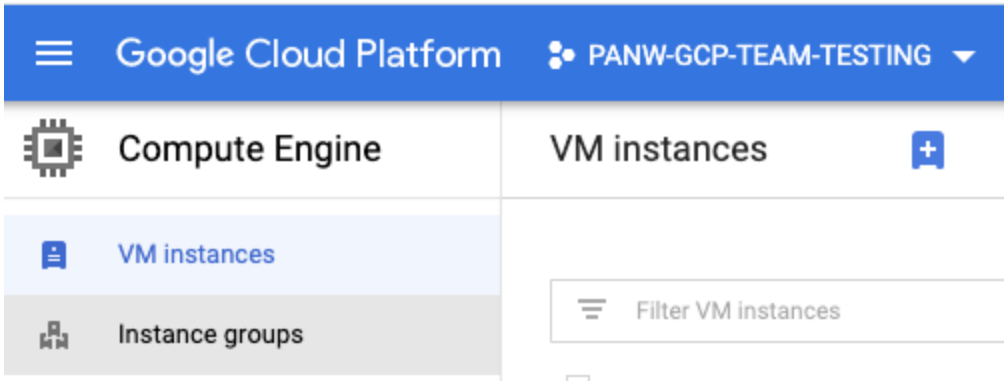
To review the guest policy in Google Cloud Console
https://console.cloud.google.com/security/agent/deployment/policies?project={project_id}
Deploying Host Defender creates OS Guest Policy. Click **VIEW DETAILS** to review the details of the Guest Policy.

2. From the list of guest policies, copy the IDs of the guest policies you would like to inspect, and then run the command to inspect each of the guest policies. Replace POLICY_ID with the policy ID that you want to review.

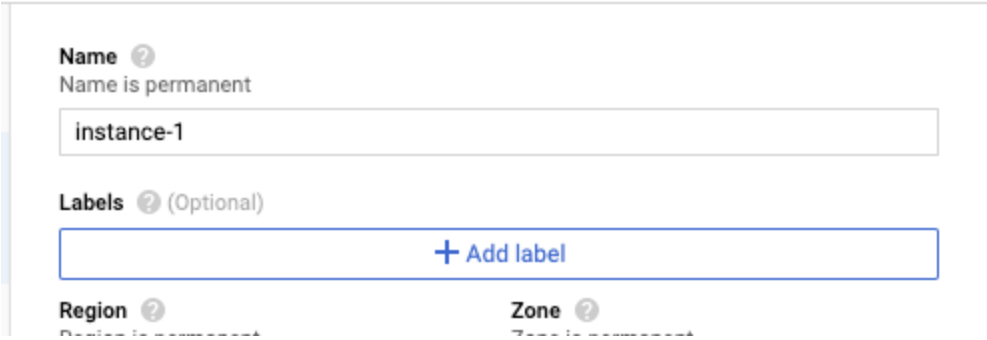
```
gcloud beta compute os-config guest-policies describe POLICY_ID
```

Create a VM instance

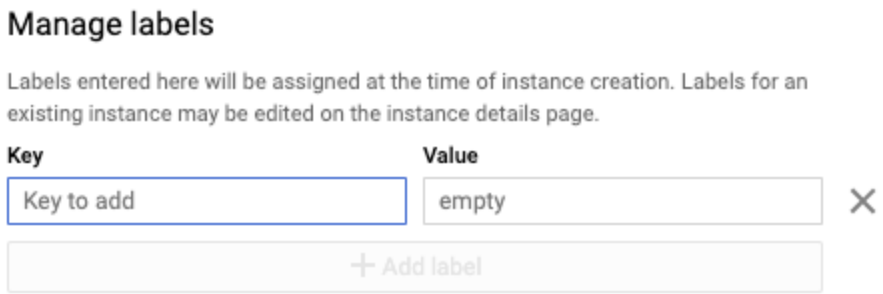
Navigate to Google Cloud Console, select **Compute Engine** > **VM Instance**



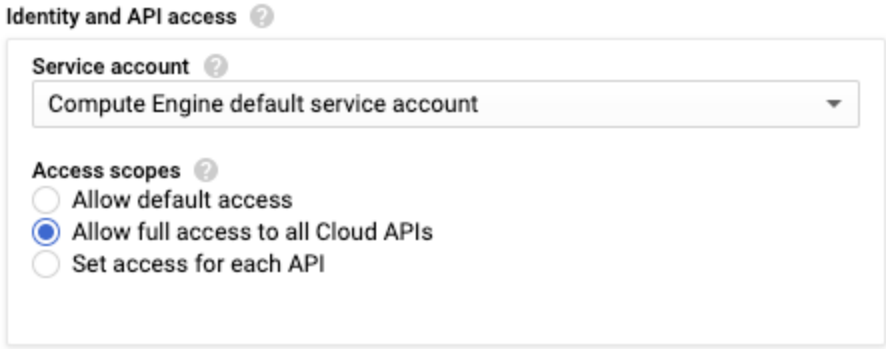
Select a name for your VM instance, make sure it matches the prefix if you use it when creating the guest policy in the previous step.



If you use a label when creating the guest policy in the previous step, make sure you add the matched label to your VM instance.



If you use the Compute Engine default Service Account for your VM instance, make sure select **Allow full access to all Cloud APIs**

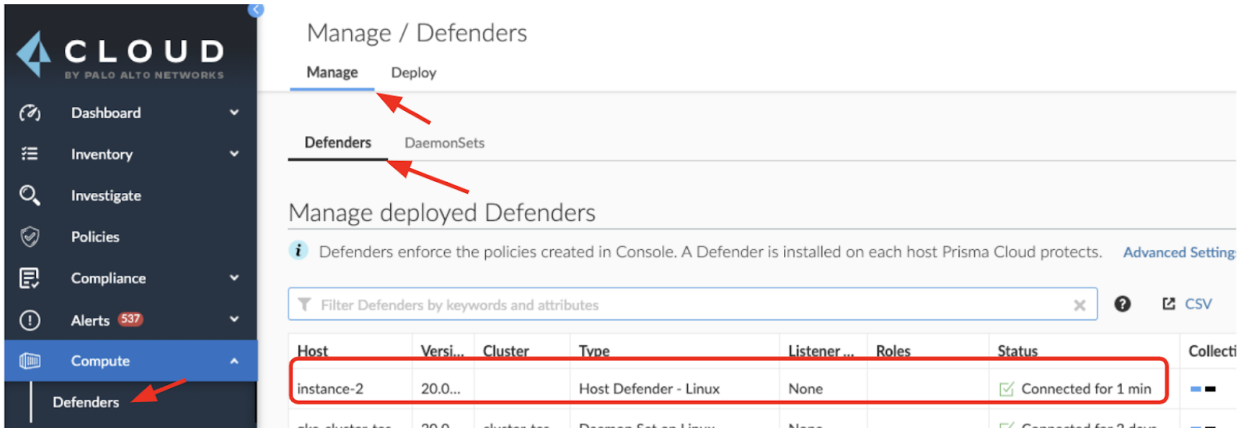


Once the instance is created, the Host Defender should be installed automatically.

Valid the Host Defender is deployed

Login to your Prisma Cloud Console and confirm if the defender is deployed:

Prisma Cloud > Compute > Manage > Defender, select **Manage, Defenders**



Troubleshooting

If the VM doesn't show up at Prisma Cloud Compute console defender list. Go through the following steps:

Confirm the guest-policy desired state at the VM instance is "Installed"

You can also run the lookup command for a given instance to see which configurations apply to that VM instance.

```
gcloud beta compute os-config guest-policies lookup VM_NAME
```

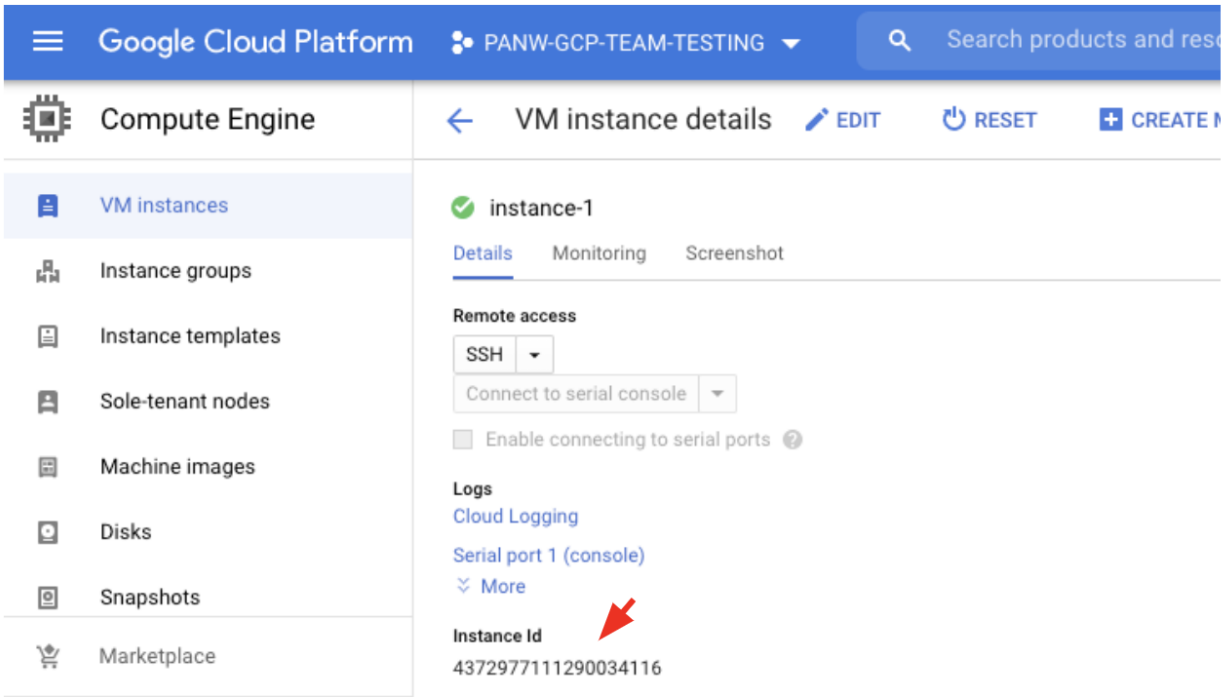
```
myan@cloudshell:~ (paloaltonetworksgcp-public)$ gcloud beta compute os-config guest-policies lookup guest-policy
Did you mean zone [us-west1-b] for instance: [guest-policy] (Y/n)?  n

No zone specified. Using zone [us-central1-a] for instance: [guest-policy].
```

SOFTWARE RECIPES			
SOURCE	NAME	VERSION	DESIRED_STATE
projects/1026607622258/guestPolicies/test	agent-deploy-prisma-linux-mypolicyid	1.0	INSTALLED

Review Guest Policy Logs

Review OSConfigAngent (Guest Policy) logs if any error.

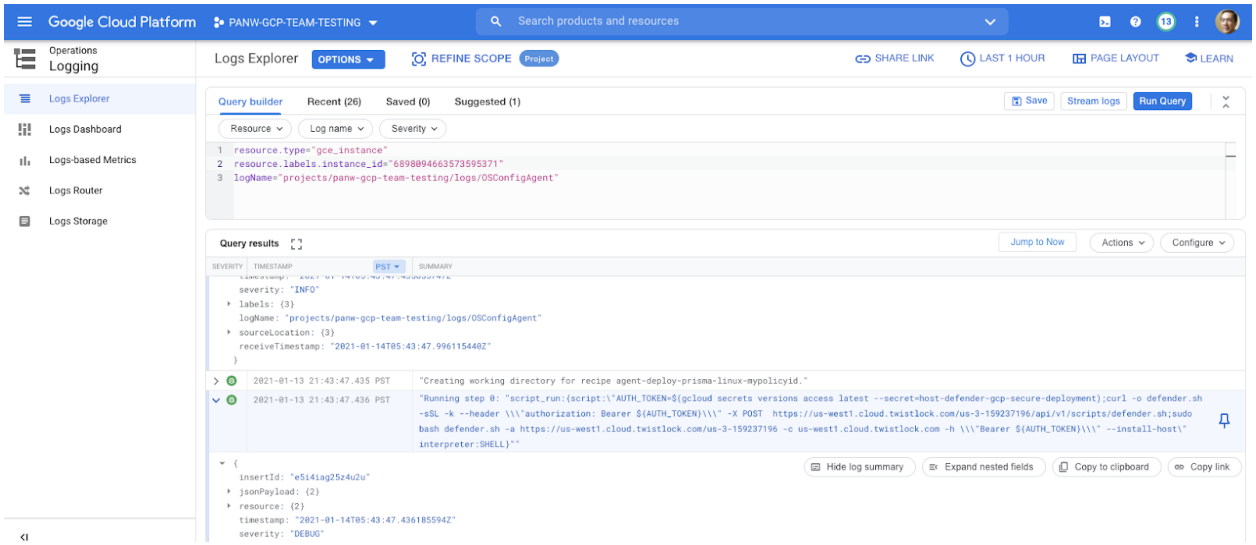
Locate the Instance ID



Navigate to Logs Explore, run query:

```
resource.type="gce_instance"
resource.labels.instance_id="your instance ID"
logName="projects/your-project_id/logs/OSConfigAgent"
```

Here is an example of a successful run:



Possible errors

Connection timeout error

`Failed downloading twistlock.cfg - curl: (7) Failed to connect to us-west1.cloud.twistlock.com port 443: Connection timed out.`

Solution

There was a connection issue. May be caused by the guest policy not fully deployed. Please wait for a couple of minutes and deploy the VM again.

Permission denied error

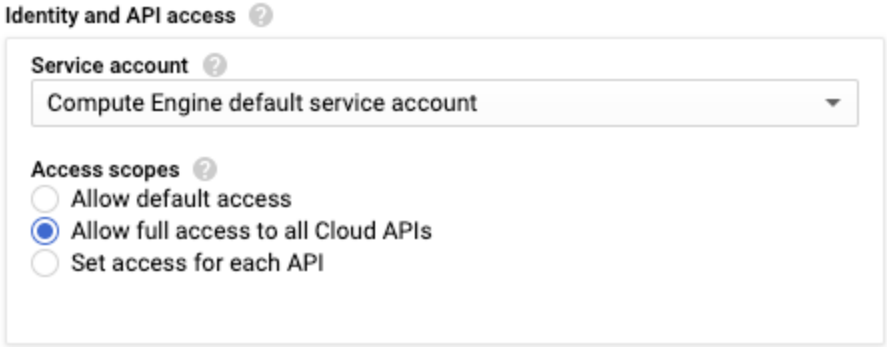
```
2021-01-06 21:07:37.524 PST   "Running step 0: "script_run:{script:\"AUTH_TOKEN=$(gcloud secrets versions access latest --secret=host-defender-gcp-secure-deployment);curl -o defender.sh -sSL -k --header \\\"authorization: Bearer ${AUTH_TOKEN}\\\" -X POST  https://us-west1.cloud.twistlock.com/us-3-159237196/api/v1/scripts/defender.sh;sudo bash defender.sh;sudo bash defender.sh -a https://us-west1.cloud.twistlock.com/us-3-159237196 -c us-west1.cloud.twistlock.com -h \\\"Bearer ${AUTH_TOKEN}\\\" --install-host\" interpreter:SHELL}""

Hide log summary | Expand nested fields | Copy to clipboard | Copy link

{
  insertId: "kjmlneg8ht6m4b"
  jsonPayload: {2}
  resource: {2}
  timestamp: "2021-01-07T05:07:37.524093507Z"
  severity: "DEBUG"
  labels: {3}
  logName: "projects/panw-gcp-team-testing/logs/OSConfigAgent"
  sourceLocation: {3}
  receiveTimestamp: "2021-01-07T05:07:38.437231124Z"
}

2021-01-06 21:07:39.936 PST   "Combined output for "/bin/sh" command:
ERROR: (gcloud.secrets.versions.access) PERMISSION_DENIED: Request had insufficient authentication scopes.
```

Solution

1. Make sure the secret is refreshing, you should see code 200. If you see 401, recreate the secret with the current token. Use the following for log query:

```
resource.type="cloud_function"
resource.labels.function_name="pcc-token-refresher"
"DEBUG"
```

2. Give the full API access when deploying VM if use the default service account. You get Access Scope error if you select default



3. Provide VM service account the access of secret manager admin to the secret **host-defender-gcp-secure-deployment** before you deploy the VM

Appendix

Installing the Cloud Logging agent on a single VM

Use this procedure if the Cloud Logging agent is not installed automatically. The Logging agent streams logs from your VM instances and from selected third-party software packages to Cloud Logging. It is a best practice to run the Logging agent on all your VM instances. Note there is additional cost associated with the logging.

1. Open a terminal connection to your VM instance using SSH or a similar tool and ensure you have sudo access.
2. Change to a directory you have write access to, for example your home directory.
3. Add the agent's package repository:

```
curl -sSO https://dl.google.com/cloudagents/add-logging-agent-repo.sh
sudo bash add-logging-agent-repo.sh
sudo apt-get update
```

4. Install the agent:

List the available versions of the agent in order to select which version to install:

```
sudo apt-cache madison google-fluentd
```

To install the latest version of the agent, run:

```
  sudo apt-get install google-fluentd
```

Install the Configuration files. For unstructured logging, run:

```
sudo apt-get install -y google-fluentd-catch-all-config
```

5. Assign proper roles

You need to make sure your instance service-account has proper rights to edit/write data to StackDriver. Simply run following commands to assign proper roles:

```
gcloud projects add-iam-policy-binding PROJECT_NAME
--member="serviceAccount:SERVICE_ACCOUNT_EMAIL"
--role="roles/logging.logWriter"

gcloud projects add-iam-policy-binding PROJECT_NAME
--member="serviceAccount:SERVICE_ACCOUNT_EMAIL"
--role="roles/monitoring.metricWriter"
```

URL to Create a Guest Policy

You may use this link to create a guest policy directly
console.cloud.google.com/security/agent/deployment/prisma

URL to Guest Policy

You may use this link to access a guest policy directly
https://console.cloud.google.com/security/agent/deployment/policies?project=*{project ID}*