

VM-Series for GCP



GCP Terraform Template Deployment Guide Single Project

Deploys a single or multiple VM-Series next generation firewall in a GCP Project
<https://www.paloaltonetworks.com>

Table of Contents

Version History	3
1. About GCP Terraform Template SP(Single Project)	4
2. Support Policy.....	4
3. Instances used	5
4. Prerequisites.....	5
4.1 Create GCP account	5
4.2 Install the Google Cloud SDK.....	5
4.3 Accept the EULA (if required).....	5
4.4 Create a Project.....	5
4.5 Enable the API	7
4.6 Create a Bootstrap Bucket.....	8
4.7 Download the Terraform Template Files.....	10
4.8 Gather Information and Update the Template File	10
5. Launch the Terraform Template	11
6. Review what was created.....	13
7. Access the firewall	14
8. Cleanup.....	15
8.1 Delete the deployment	15
9. Conclusion.....	16
Appendix A	16
Troubleshooting tips	16

Version History

Version number	Comments
1.0	Initial Draft

1. About GCP Terraform Template SP (Single Project)

GCP Terraform Templates SP(Single Project), are files that can deploy, configure, and launch GCP resources such as VPC networks & subnets, security groups, firewall rules, route tables, and more. These templates are used for ease of deployment and are key to any cloud deployment model.

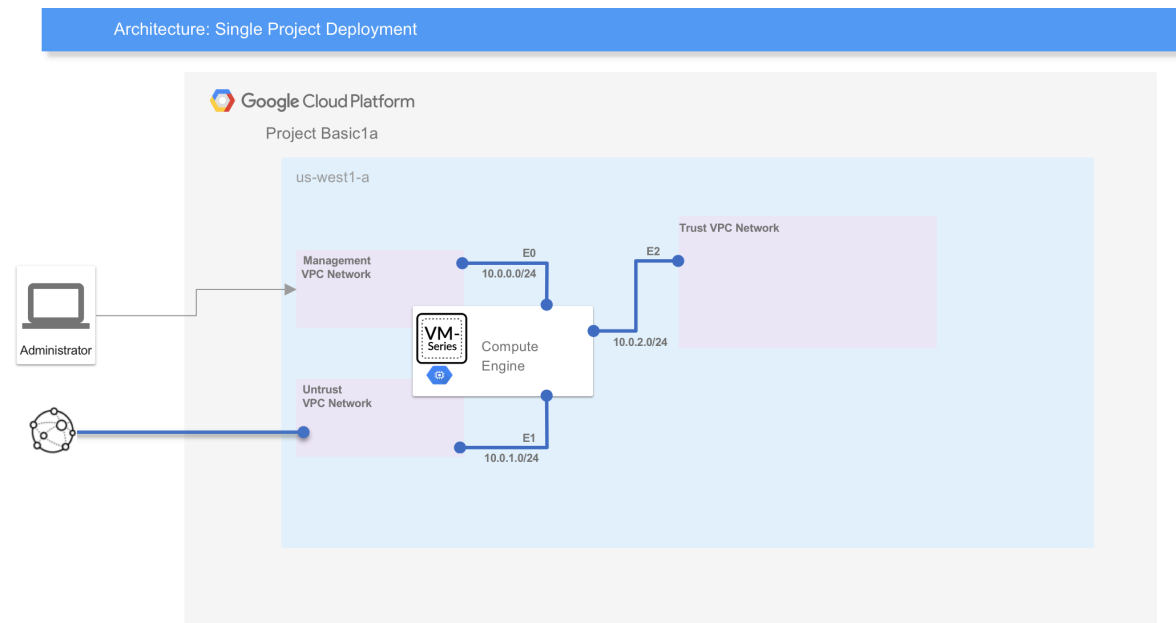
For more information on Templates refer to Google's documentation

<https://cloud.google.com/community/tutorials/managing-gcp-projects-with-terraform>

There are also many Terraform template s available here:

<https://github.com/GoogleCloudPlatform/terraform-google-examples>

This document will explain how to deploy a Terraform template that launches everything that is shown below in the diagram. This includes a VM-Series firewall and the subnets. In addition, the Terraform template can enable the VM-Series firewall to perform a native bootstrapping feature that allows for additional configuration of the VM-Series firewall (such as routes, security policies, etc.) Once the Terraform template has been deployed, the network topology will align with the following diagram:



2. Support Policy

This template is released under an as-is, best effort, support policy. These scripts should be seen as community supported and Palo Alto Networks will contribute our expertise as and when possible. We do not provide technical support or help in using or troubleshooting the components of the project through our normal support options such as Palo Alto Networks support teams, or ASC (Authorized Support Centers) partners and backline support options. The underlying product used (the VM-Series firewall) by the scripts or templates are still supported, but the support is only for the product functionality and not for help in deploying or using the template or script itself.

Unless explicitly tagged, all projects or work posted in our GitHub repository (at <https://github.com/PaloAltoNetworks/googlecloud>) or sites other than our official Downloads page on <https://support.paloaltonetworks.com> are provided under the best effort policy.

3. Instances used

When deploying this Terraform template the following machine types are used:

Instance name	Machine Type
VM Series Firewall	n1-standard-4

Note: There are GCP costs associated with each machine type launched, please refer to the Google instance pricing page <https://cloud.google.com/compute/pricing>

4. Prerequisites

Here are the prerequisites required to successfully launch this template:

- Terraform installed

4.1 Create GCP account

If you do not have a GCP account already, go to <https://cloud.google.com/free/> and create an account.

4.2 Install the Google Cloud SDK

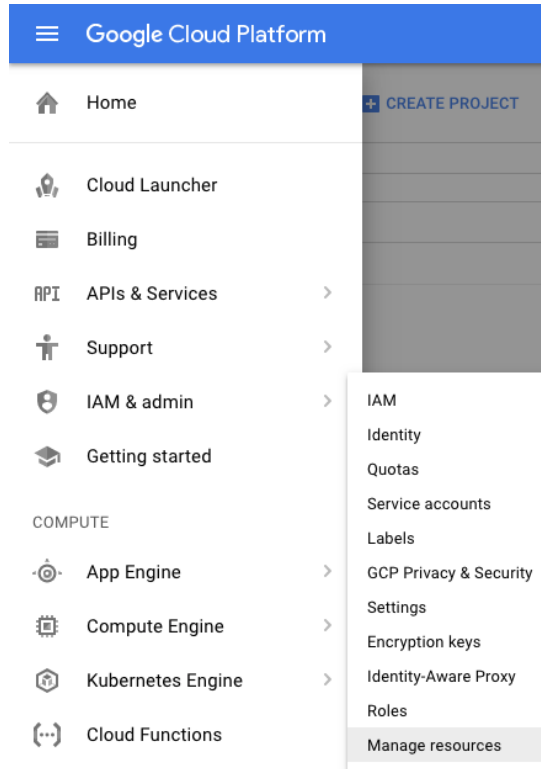
Template installations in GCP are performed from the CLI. Install the SDK/CLI by selecting the relevant platform from the following link and following the installation instructions:

<https://cloud.google.com/sdk/>

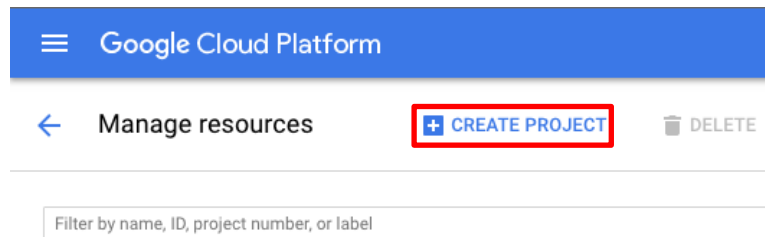
4.3 Accept the EULA (if required)

4.4 Create a Project

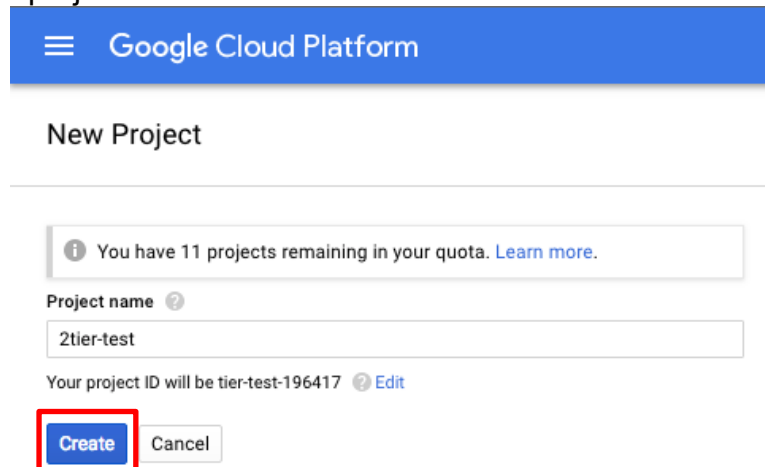
All GCP resources are deployed to a project, which is an organizational boundary that separates users, resources, billing information, etc. It is similar to an AWS VPC or an Azure Resource Group. By default, GCP will create a project upon creation of an account. To create a dedicated project, use the drop-down on the left and select **IAM & admin > Manage Resources**:



Click **Create Project**:



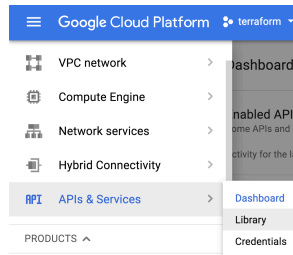
Specify a name for the project and click **Create**:



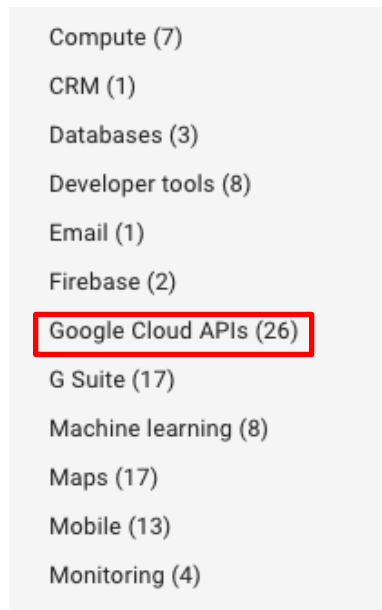
Note that project creation will take a few minutes.

4.5 Enable the API

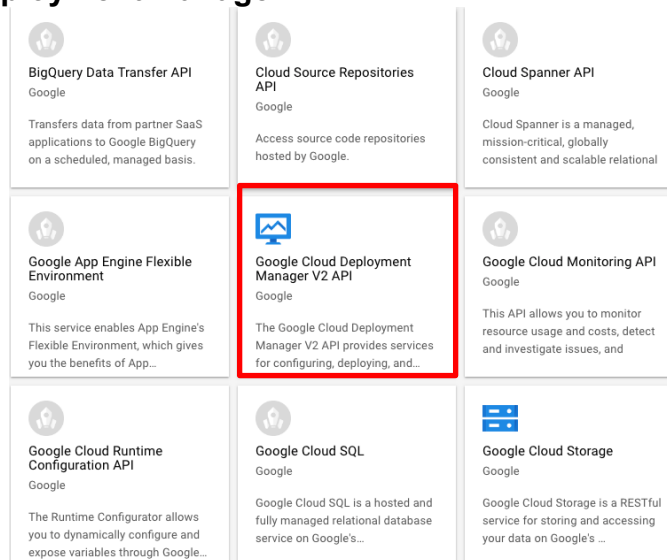
Deploying a template requires the API be enable on the project. Navigate to **APIs & Services > Library**:



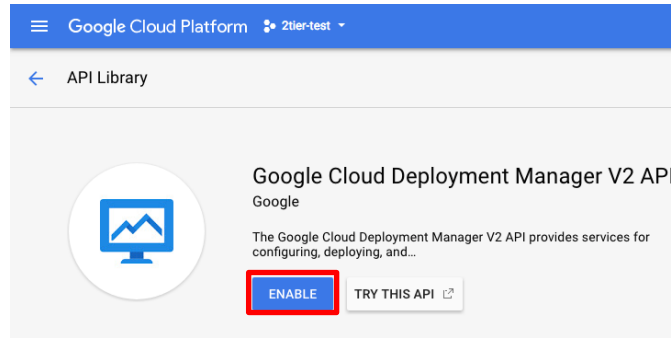
Select Google Cloud APIs on the left-hand-side:



Select **Google Cloud Deployment Manager V2 API**:



Select **Enable**:

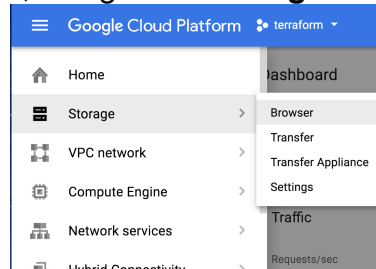


Enabling the API for the project will take a few minutes to complete.

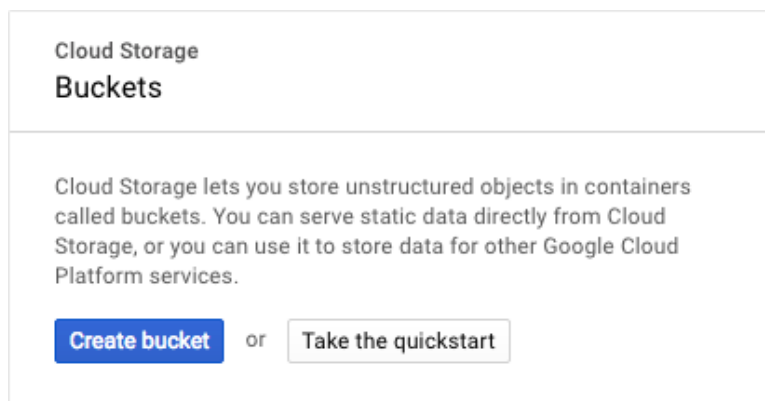
4.6 Create a Bootstrap Bucket

Bootstrapping is a feature of the VM-Series firewall that allows you to load a pre-defined configuration into the VM-Series firewall during boot-up. This ensures that the VM-Series firewall is configured and ready at initial boot-up, thereby removing the need for manual configuration. The bootstrapping feature also enables automating deployment of the VM-Series firewall.

In order to create a Bootstrap bucket, navigate to **Storage > Browser**:



Click Create Bucket:



Specify a globally-unique bucket name and regional settings and click **Create**:

Palo Alto Networks GCP Terraform Template Deployment Guide Single Project

The screenshot shows the 'Create a bucket' form in the Google Cloud Storage console. The left sidebar contains navigation links: Storage, Browser, Transfer, Transfer Appliance, and Settings. The main form area has the following fields and options:

- Name:** A text input field containing 'tf-single-project'. A note below states: 'Must be unique across Cloud Storage. Privacy: Do not include sensitive information in your bucket name. Others can discover your bucket name if it matches a name they're trying to use.'
- Default storage class:** A radio button selection with four options:
 - Multi-Regional:** Selected. 'Use to stream videos and host hot web content. Best for data accessed frequently around the world.'
 - Regional:** 'Use to store data and run data analytics. Best for data accessed frequently in one part of the world.'
 - Nearline:** 'Use to store rarely accessed documents. Best for data accessed less than once per month.'
 - Coldline:** 'Use to store very rarely accessed documents. Best for data accessed less than once per year.'
- Multi-Regional location:** A dropdown menu showing 'United States'. A note states: 'Redundant across 2+ regions within your selected location.'
- Specify labels:** A link to expand the section.
- Create/Cancel buttons:** At the bottom of the form.

You will need to enter a globally unique bucket name. GCP will warn you if the name is not unique. Once the bucket is created, click on the newly created bucket and add four folders called **config**, **license**, **software** and **content** by clicking on **Create Folder**:

The screenshot shows the 'Browser' view of the 'bootstrap-tf' bucket. At the top, there are buttons for 'UPLOAD FILES', 'UPLOAD FOLDER', and 'CREATE FOLDER'. Below the buttons is a search bar labeled 'Filter by prefix...'. The breadcrumb path is 'Buckets / bootstrap-tf'. A table lists the contents of the bucket:

Name	Size	Type
config/	—	Folder
content/	—	Folder
license/	—	Folder
software/	—	Folder

Download the following files and save them in a known location:

<https://raw.githubusercontent.com/PaloAltoNetworks/googlecloud/master/two-tier-template/bootstrap.xml>

<https://raw.githubusercontent.com/PaloAltoNetworks/googlecloud/master/two-tier-template/init-cfg.txt>

Now click on the **config** folder in the console and click **UPLOAD FILES**:

The screenshot shows the 'Browser' view of the 'config' folder within the 'bootstrap-tf' bucket. At the top, there are buttons for 'UPLOAD FILES', 'UPLOAD FOLDER', 'CREATE FOLDER', 'REFRESH', and 'SHARE PUB'. Below the buttons is a search bar labeled 'Filter by prefix...'. The breadcrumb path is 'Buckets / bootstrap-tf / config'. A table lists the contents of the folder:

Name	Size	Type	Storage class	Last modified
bootstrap.xml	20.68 KB	text/xml	Multi-Regional	2/10/18, 11:21 AM
init-cfg.txt	363 B	text/plain	Multi-Regional	2/10/18, 11:21 AM





Select the two files (bootstrap.xml and init-cft.txt) downloaded previously and upload them to the **config** folder.

The **content**, **license**, & **software** folders can be left empty.

NOTE: Please create the folders above using the GCP GUI console. Creating folders locally on your machine and uploading them may not work as expected.

4.7 Download the Terraform Template Files

Click on each file and select Raw and save all of the files to a known location:

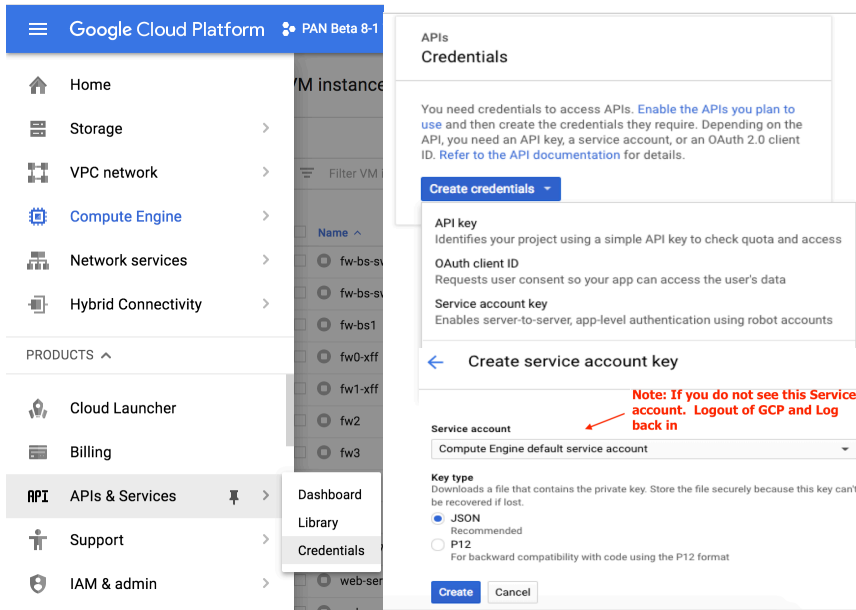
..		
 GCP_Terraform_Template_Single_Project_How_To_Guide.pdf	Add files via upload	4 hours ago
 Main.tf	Add files via upload	4 hours ago
 Variables.tf	Add files via upload	4 hours ago
 readme	Create readme	4 hours ago
<div> readme</div> <div>This Terraform template deploys single or multiple VM-Series firewalls into a single GCP Project.</div>		

4.8 Gather Information and Update the Template File

Deploying the Terraform template in GCP requires modification of the template Main and Variable files to include deployment-specific information. The required information is:

```
credentials = "${file("Your_Project_Credentials.json")}"
project     = "${Your_Project_ID}"
region      = "${Your_Project_Region}"
zone        = "${Your_Project_Zone}"
sshKey      = "${Your_Public_SSHKey}"
```

To create the credentials to access the APIs in JSON format. In GCP console go to (APIs & Services > Credentials > Create Credentials > Service Account Key), and download the file (client_secrets.json). Put the .json credential file in your Terraform template folder.



Once the information has been gathered, update the Main and Variable files with the information. Save the Files.

5. Launch the Terraform Template

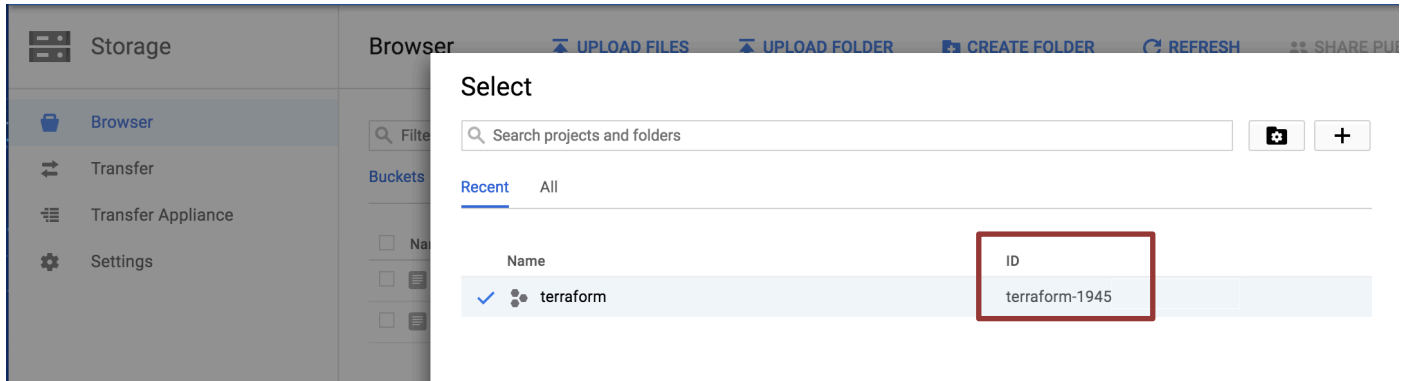
Navigate to a command shell navigate to the directory containing the downloaded template files:

Authenticate to the GCP environment from the command line with the command:

```
$ gcloud auth login
```

- Copy/paste the link into a browser and select the account to authenticate if a browser does not automatically launch:
- Review the requested permissions and click **Allow**:
- Copy the one-time verification code:
- Paste it into the window to complete the authentication request (ignore the warning):

Note the Project ID:



Set the target project for template deployment via command line:

```
$ gcloud config set project my_Project_id
```

Run Terraform Commands:

Initiate template deployment using command “**terraform init**”.

```
Terraform has been successfully initialized!
```

```
You may now begin working with Terraform. Try running "terraform plan" to see  
any changes that are required for your infrastructure. All Terraform commands  
should now work.
```

```
If you ever set or change modules or backend configuration for Terraform,  
rerun this command to reinitialize your working directory. If you forget, other  
commands will detect it and remind you to do so if necessary.
```

Once the terraform init has completed run the command “**terraform plan**”.

```
Plan: 12 to add, 0 to change, 0 to destroy.
```

```
Note: You didn't specify an "-out" parameter to save this plan, so Terraform  
can't guarantee that exactly these actions will be performed if  
"terraform apply" is subsequently run.
```

You will see if there are any errors and what terraform will be deploying. Now run the “**terraform apply**” command and say **yes** when prompt.

```
Plan: 12 to add, 0 to change, 0 to destroy.
```

```
Do you want to perform these actions?  
Terraform will perform the actions described above.  
Only 'yes' will be accepted to approve.
```

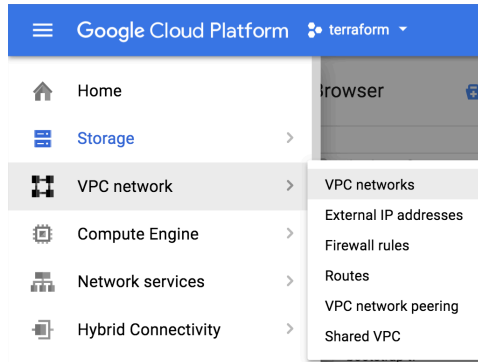
```
Enter a value: yes
```

If all goes well, Terraform will report success (“Apply Complete!” and no errors):

```
Apply complete! Resources: 12 added, 0 changed, 0 destroyed.
```

6. Review what was created

Let's review what the Terraform template has launched. The newly created networks can be viewed via **VPC Networks > VPC Network**:

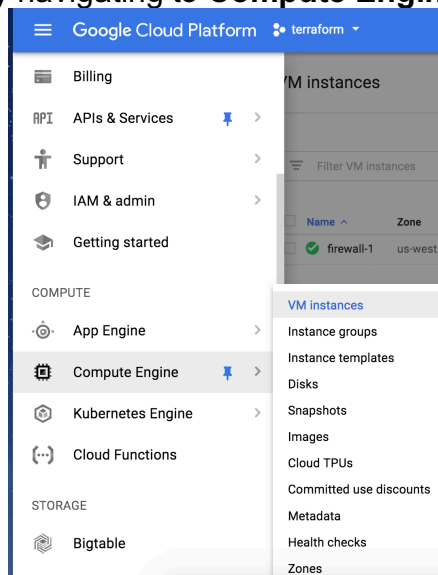


The Terraform template creates three networks with 1 subnet each: management, untrust, and trust:


VPC network	VPC networks CREATE VPC NETWORK REFRESH						
	Name	Region	Subnets	Mode	IP addresses ranges	Gateways	Firewall Rules
	management	us-west1	1	Custom	10.0.0.0/24	10.0.0.1	1
	trust	us-west1	1	Custom	10.0.2.0/24	10.0.2.1	1
	untrust	us-west1	1	Custom	10.0.1.0/24	10.0.1.1	1

Note: The default network was automatically created when the project was instantiated. It can be ignored or deleted.

Deployed hosts can be viewed by navigating to **Compute Engine > VM Instances**:



High-level information regarding the deployed instances are available with the default view:



Compute Engine

VM instances

Instance groups

Instance templates

Disks

Snapshots

Images

Cloud TPUs

Committed use discounts

CREATE INSTANCE

IMPORT VM

REFRESH



START

STOP

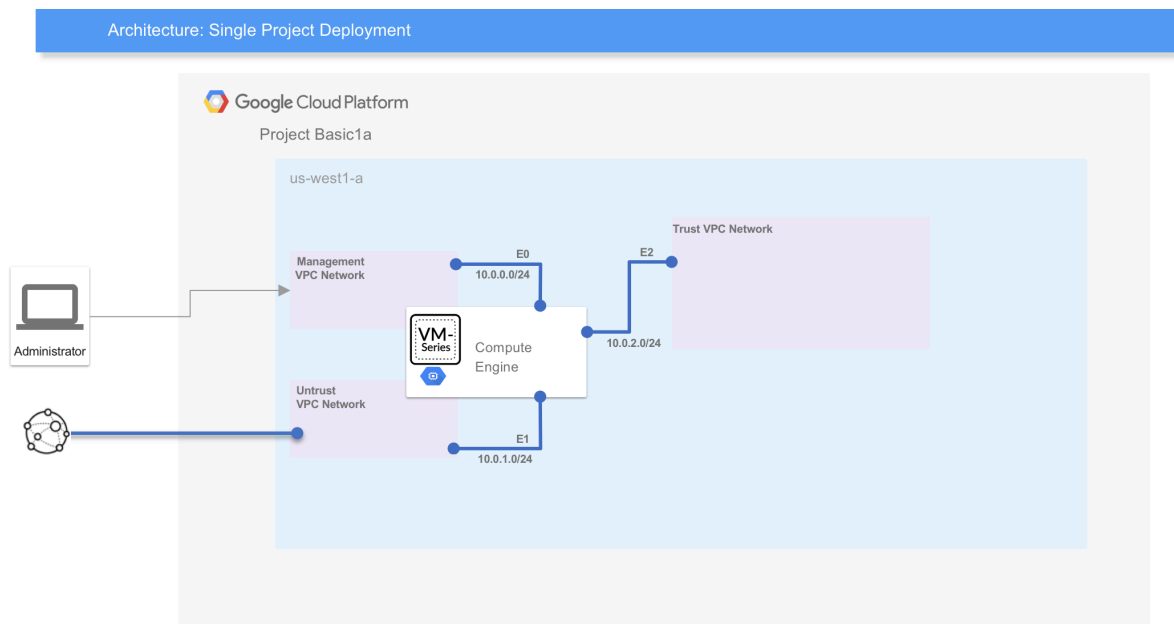
RESET

Filter VM instances

Columns

<input type="checkbox"/>	Name ^	Zone	Recommendation	Internal IP	External IP	Connect
<input type="checkbox"/>	 firewall-1	us-west1-a		10.0.0.2	35.230.45.56	SSH 

All of this matches the topology shown below:

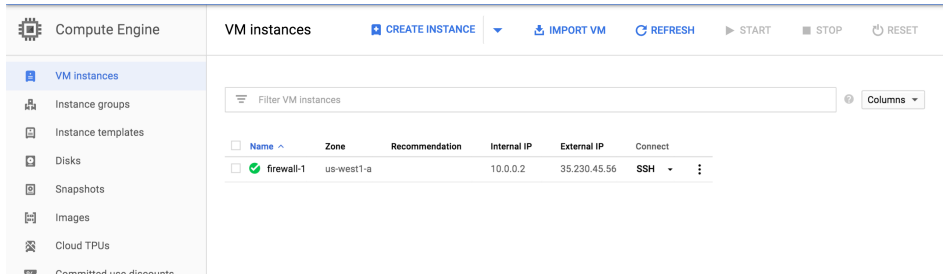


7. Access the firewall

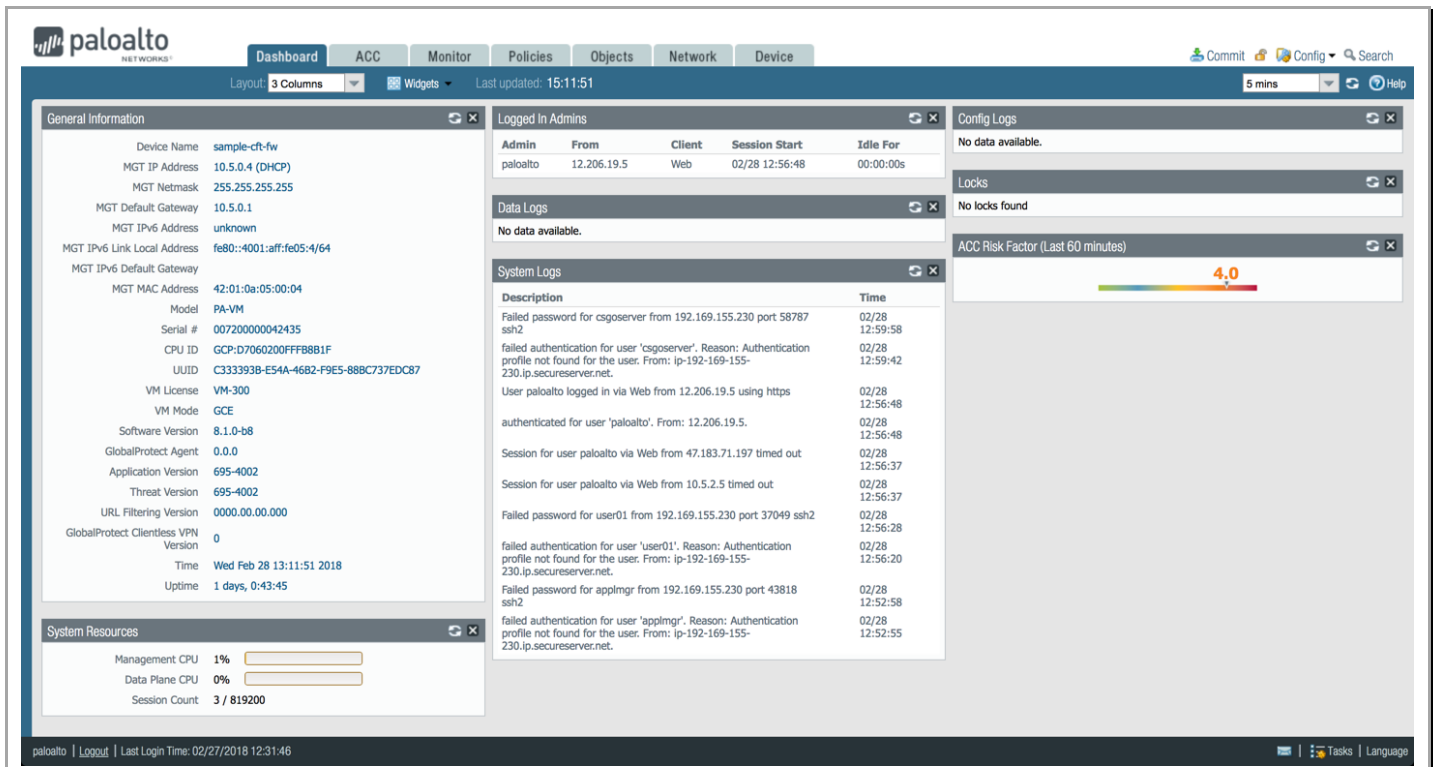
NOTE: Bootstrapping a VM-Series firewall takes approximately 9 minutes. Be patient😊 once the template has been deployed successfully, it may be a while before the VM-Series firewall is up and you are able to log into the VM-Series firewall.

Once the template deployment is complete, the VM-Series firewall will show a green checkmark indicating it is running. On the right side is the management IP address of the firewall:

Palo Alto Networks GCP Terraform Template Deployment Guide Single Project



You should now be able to login to the firewall using the **username: paloalto** and password: **Pal0Alt0@123**



8. Cleanup

8.1 Delete the deployment

Once done, cleanup as follows:

- If you licensed the VM-Series firewall perform the De-License function.
 - https://www.paloaltonetworks.com/documentation/71/virtualization/virtualization/license-the-vm-series-firewall/deactivate-vm#_87329

- From the CLI, issue the command “**terraform destroy**”
 - This will delete all the resources created via the Terraform template.

9. Conclusion

You have successfully deployed a Terraform template in GCP and demonstrated how the Palo Alto Next Generation VM-Series firewall can be deployed via Terraform automation to not only secure traffic throughout your GCP Project, but throughout your Enterprise Google Cloud Infrastructure.

Appendix A

Troubleshooting tips

Bootstrapping not working

If the VM-Series firewall is up and you are able to access the login page, but unable to login using the username/password: paloalto/Pal0Alt0@123, then chances are bootstrapping has failed. See below:

a. Corrupt configuration files

Please ensure that the bootstrap.xml and init-cft.txt files mentioned in [Section 4.6](#) are not corrupted.

b. Incorrect bootstrap bucket-name or bucket out of region

Another reason for bootstrapping to fail is that the bootstrap bucket name (Parameter: <your-bootstrapbucket>) was incorrectly entered in the template file. Please make sure the bucket name created in [Section 4.6](#) is mentioned when launching the template.