**VM-Series for GCP**



**GCP Terraform Template**

**Deployment Guide**

Deploys a Load Balancer Sandwich of an Autoscaling Managed Instance Group for VM-Series in GCP. Only works use with PAYGO Images with Predefined NAT Rules for HealthCheck

<https://www.paloaltonetworks.com>

Table of Contents

[Version History 3](#_Toc516168676)

[1. About Terraform Templates 4](#_Toc516168677)

[2. Support Policy 5](#_Toc516168678)

[3. Instances used 6](#_Toc516168679)

[4. Prerequisites 6](#_Toc516168680)

[4.1 Create GCP account 6](#_Toc516168681)

[4.2 Install the Google Cloud SDK 6](#_Toc516168682)

[4.3 Accept the EULA (If Required) 6](#_Toc516168683)

[4.4 Create a Project 6](#_Toc516168684)

[4.5 Enable the API 8](#_Toc516168685)

[4.6 Create a Bootstrap Bucket 9](#_Toc516168686)

[4.7 Download the Terraform Template Files 12](#_Toc516168687)

[4.8 Gather Information and Update the Template File 12](#_Toc516168688)

[5. Launch the Template 13](#_Toc516168689)

[6. Review what was created 15](#_Toc516168690)

[7. Access the firewall 20](#_Toc516168691)

[8. Access the Webservers via ELB 25](#_Toc516168692)

[9. Cleanup 25](#_Toc516168693)

[9.1 Delete the deployment 25](#_Toc516168694)

[10. Conclusion 26](#_Toc516168695)

[Appendix A 26](#_Toc516168696)

[Troubleshooting tips 26](#_Toc516168697)

# Version History

|  |  |
| --- | --- |
| Version number | Comments |
| 1.0 | Initial Draft |

# About Terraform Templates

GCP Terraform Templates, are files that can deploy, configure, and launch GCP resources such as VPC networks & subnets, security groups, firewall rules, route tables, load balancers, 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

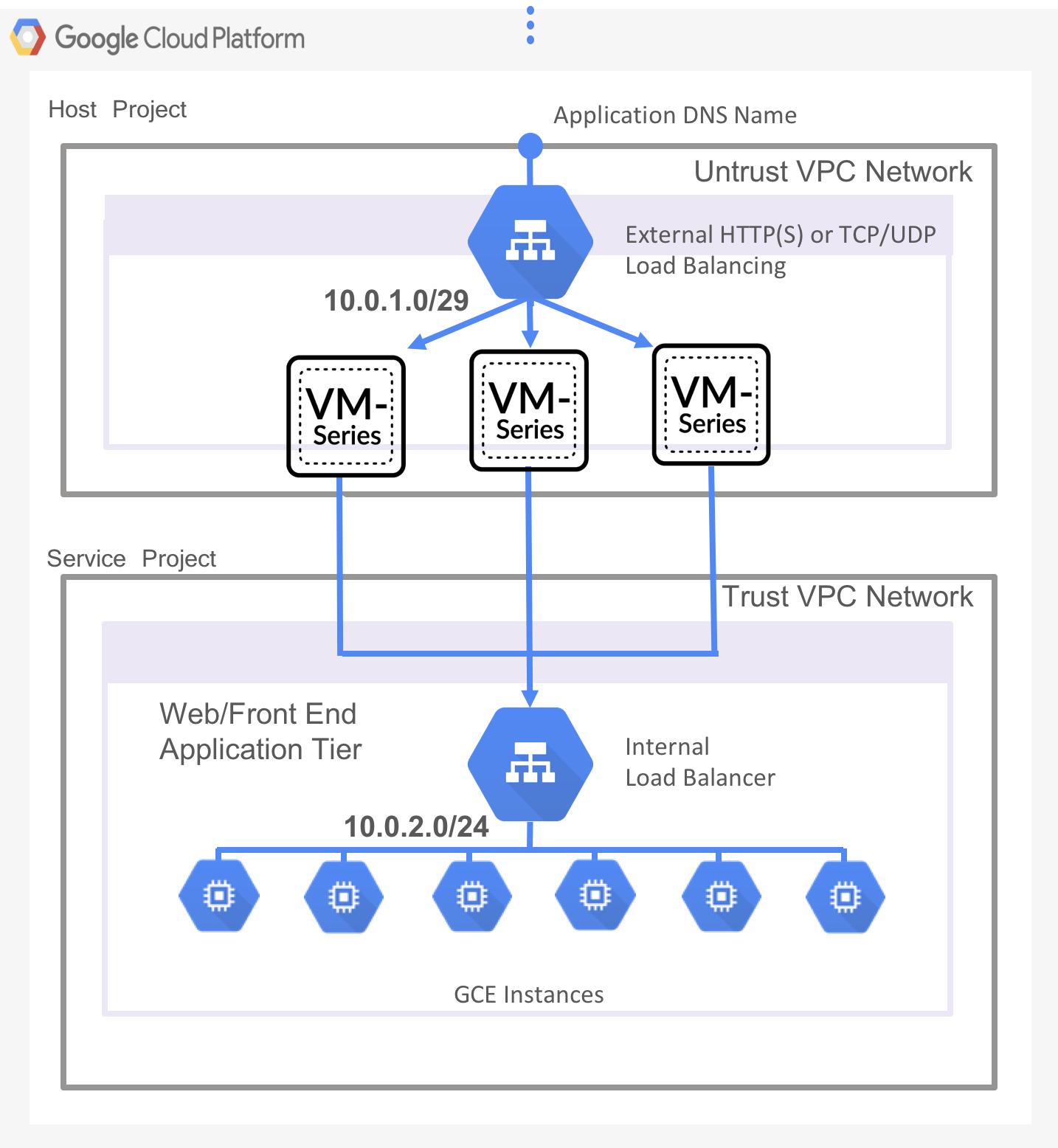
<https://cloud.google.com/community/tutorials/managing-gcp-projects-with-terraform>

There are also many Terraform templates available here:

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

<https://github.com/PaloAltoNetworks>

This document will explain how to deploy a Terraform template that launches everything that is shown below in the diagram. This includes, GCP Instance Template, GCP Managed Instance Group (MIG), VM-Series firewall(s) and subnets, HTTP ELB, and TCP ILB with HealthChecks. In addition, the Terraform template performs a native bootstrapping feature for the VM-Series firewall that allows for additional configuration of the VM-Series firewall (such as routes, security policies, NAT polices, & Management Interface Swap, etc.) Once the Terraform template has been deployed, the network topology will align with the following diagram:



# Support Policy

This template is released under an as-is, best effort, community support policy. This script should be seen as community supported. 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/or backline support options. The underlying product used (the VM-Series firewall) by the scripts or templates are 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>) or sites other than our official Downloads page on <https://support.paloaltonetworks.com> are provided as community scripts.

# Instances used

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

|  |  |
| --- | --- |
| **Instances** | **Machine Types** |
| Apache Web Servers | f1-micro |
| VM Series Firewall | n1-standard-4 |

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

# Prerequisites

Here are the prerequisites required to successfully launch this template:

* Terraform installed

## Create GCP account

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

## Install the Google Cloud SDK

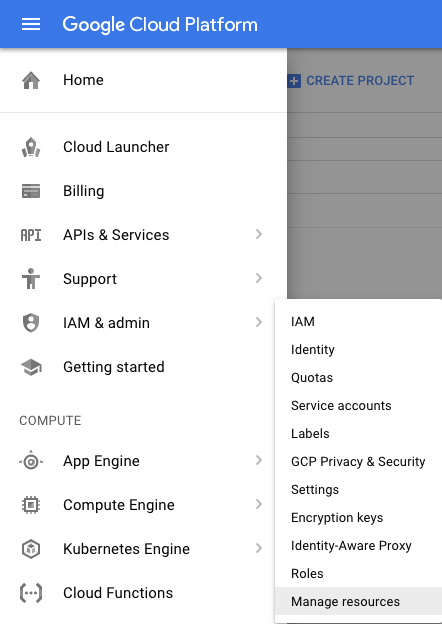
Template installations in GCP are performed from the CLI. Install the SDK/CLI by selecting the relevant platform from the following link and following the installation instructions:

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

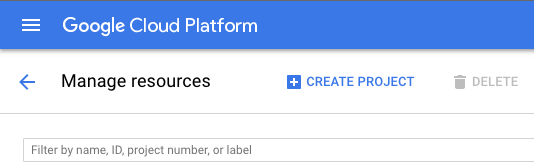
## Accept the EULA (If Required)

## Create a Project

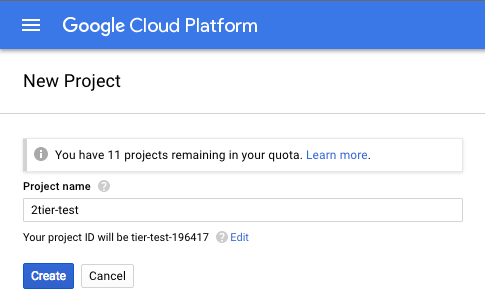
All GCP resources are deployed to a GCP Project. A GCP Project is an organizational boundary that separates users, resources, billing information, etc. By default, GCP will create a Project upon creation of an account. If that is not the case or to manually create a dedicated project, use the drop-down on the left and select **IAM & admin > Manage Resources**:



Click **Create Project**:



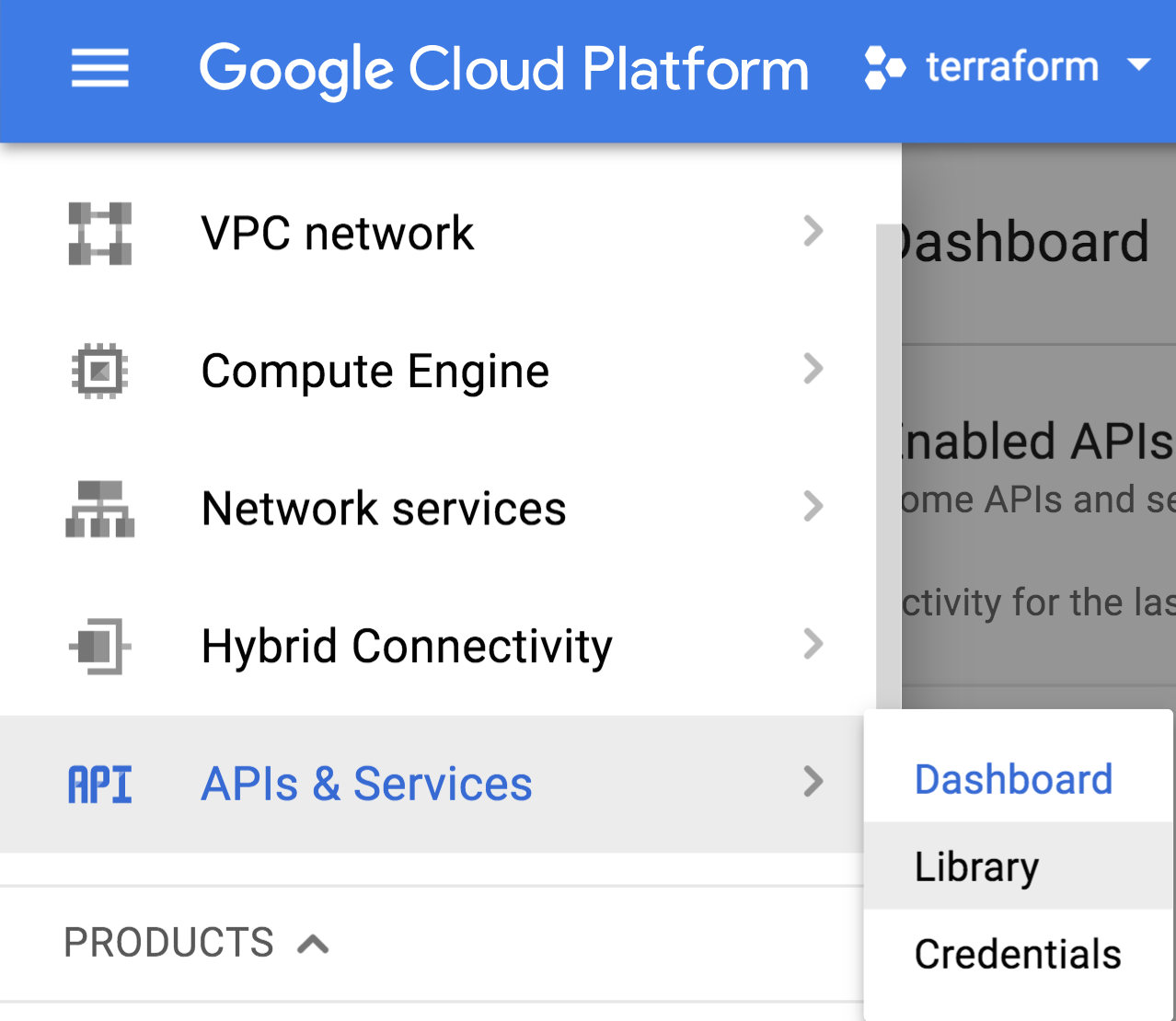
Specify a name for the project and click **Create**:



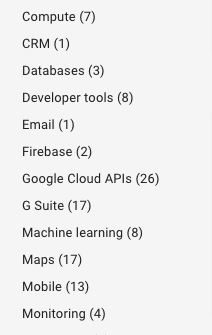
**Note: GCP Project creation will take a few minutes.**

## Enable the API

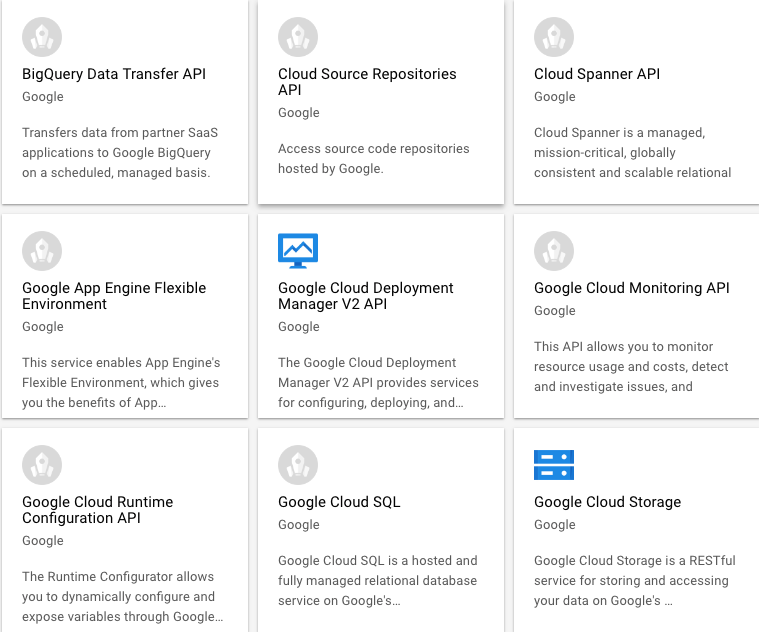
Deploying a template requires the API be enable on the project. Navigate to **APIs & Services > Library**:



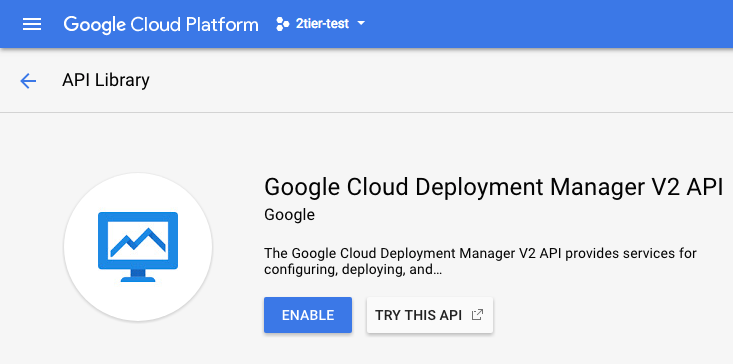
Select Google Cloud APIs on the left-hand-side:



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



Select **Enable**:

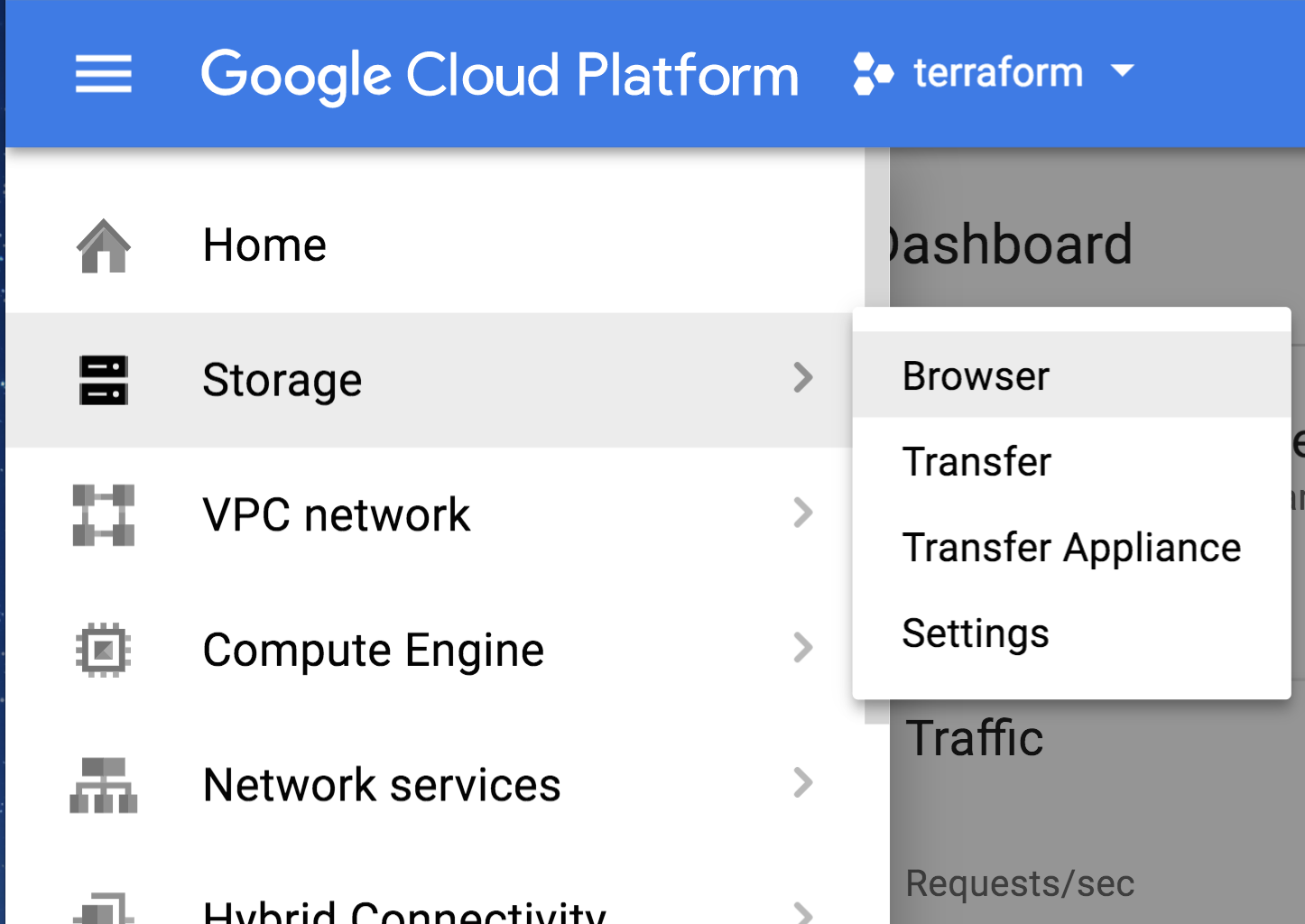


**Note: Enabling the API for the project will take a few minutes to complete.**

## Create a Bootstrap Bucket

Bootstrapping is a feature of the VM-Series firewall that allows you to load a pre-defined configuration into the firewall during boot-up. This ensures that the 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.

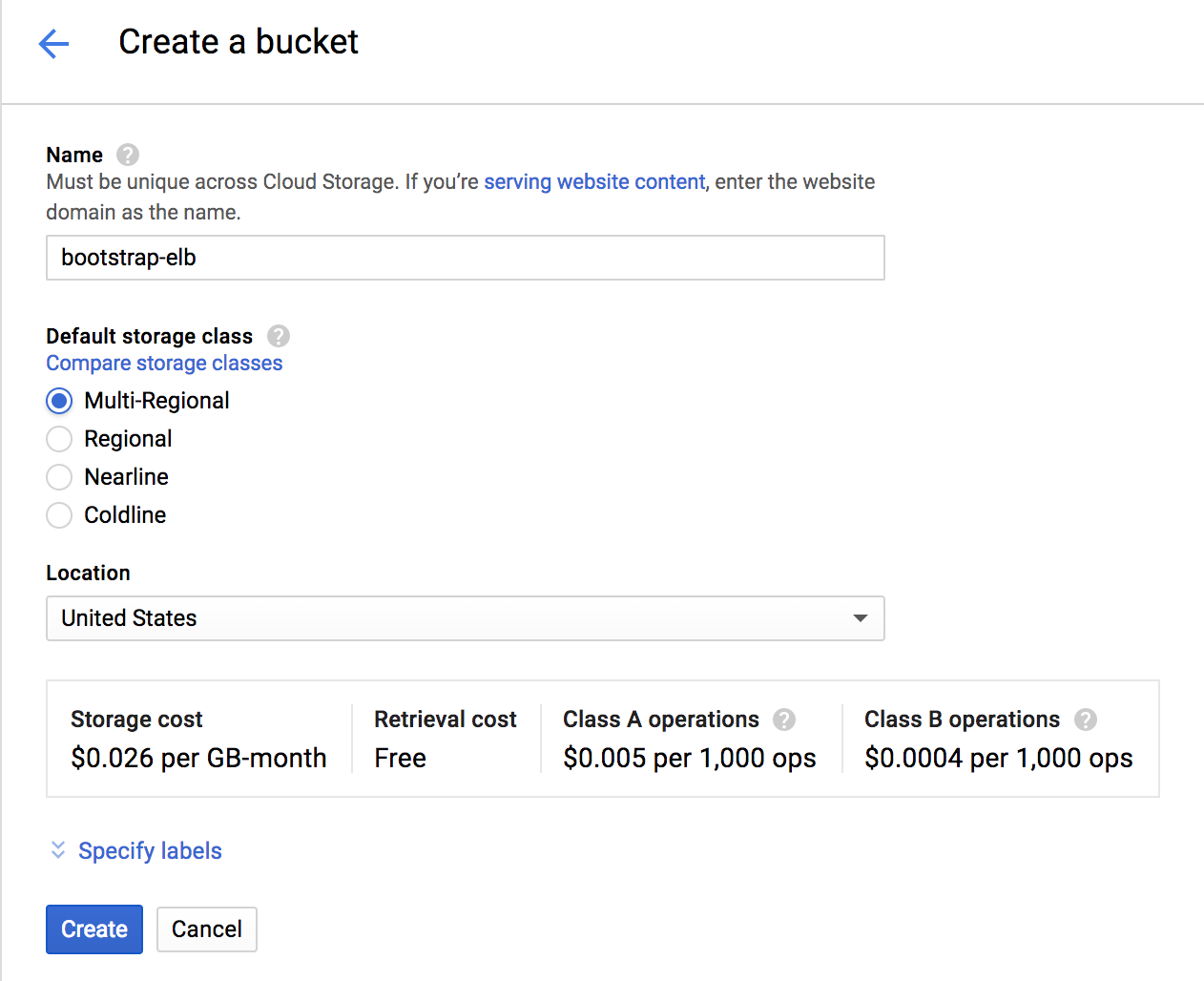
In order to create a Bootstrap bucket, navigate to **Storage > Browser**:



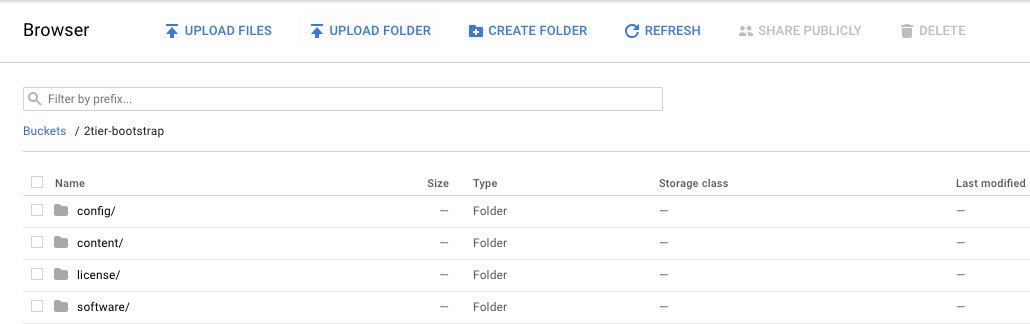
Click **Create Bucket**:



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



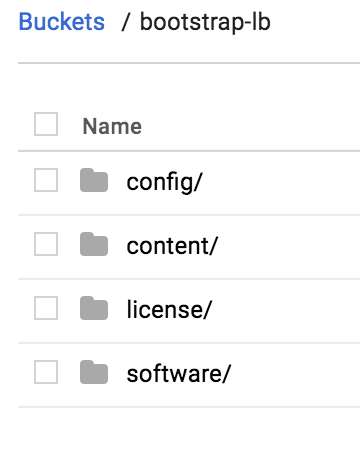
You will need to enter a globally unique bucket name. GCP will warn you if the name is not unique. Once the bucket is created, click on the newly created bucket and add four folders called **config, license, software** and **content** by clicking on **Create Folder**:

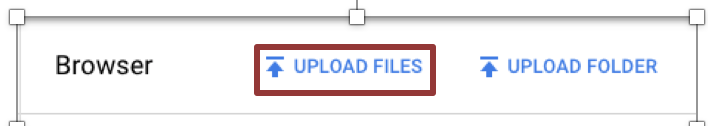


Download the following files using the links provided and save the files in a known location:

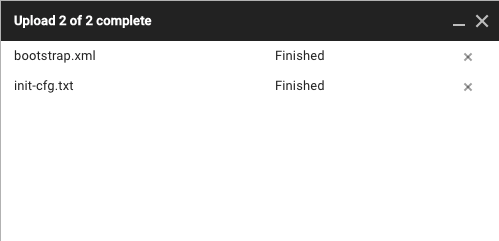
Insert git links here..

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





Select the two files (bootstrap.xml and init-cft.txt) downloaded previously and click **Open**:



NOTE: All four folders must be created for the bootstrapping process to occur. However, all folders DO NOT need to contain files.

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

## Download the Terraform Template Files

Download and save all of the template files to a known location by selecting **Clone or download**:

Insert git links here..

## 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 minimum required information is:

credentials = "${file("Your\_Project\_Credentials.json")}"

metadata\_ startup\_script= "${file("location/startup.sh")}"

project = "${Your\_Project\_ID}"

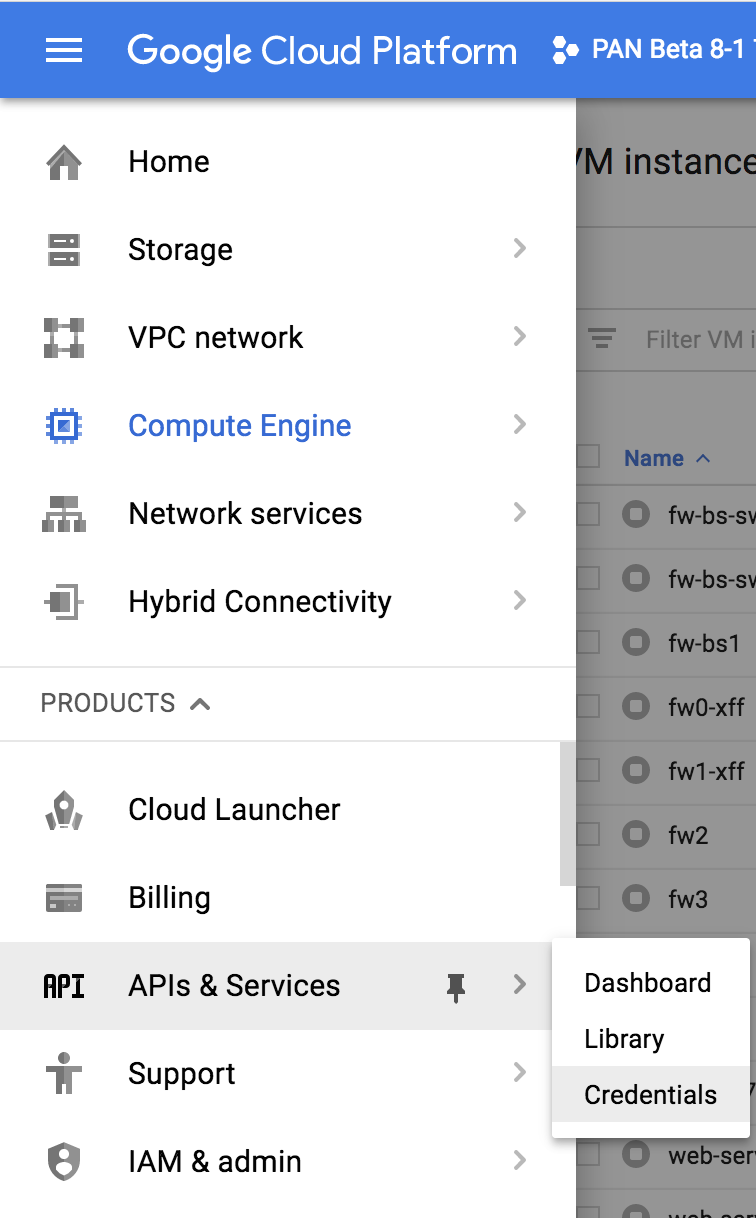
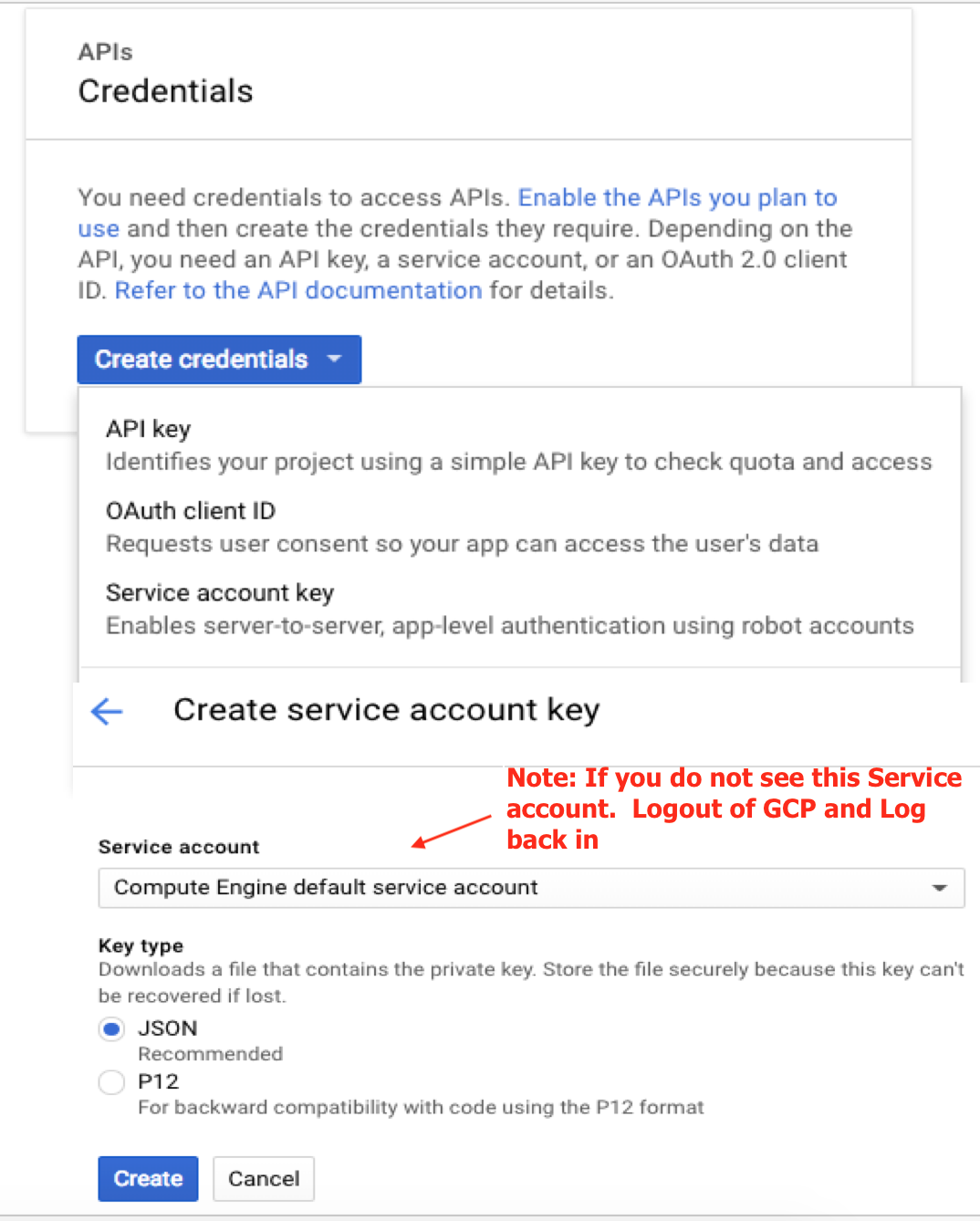
region = "${Your\_Project\_Region}"

zone = "${Your\_Project\_Zone}"

2nd zone = "${Your\_Zone2}"

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.

# Launch the 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):

Get the Project ID:

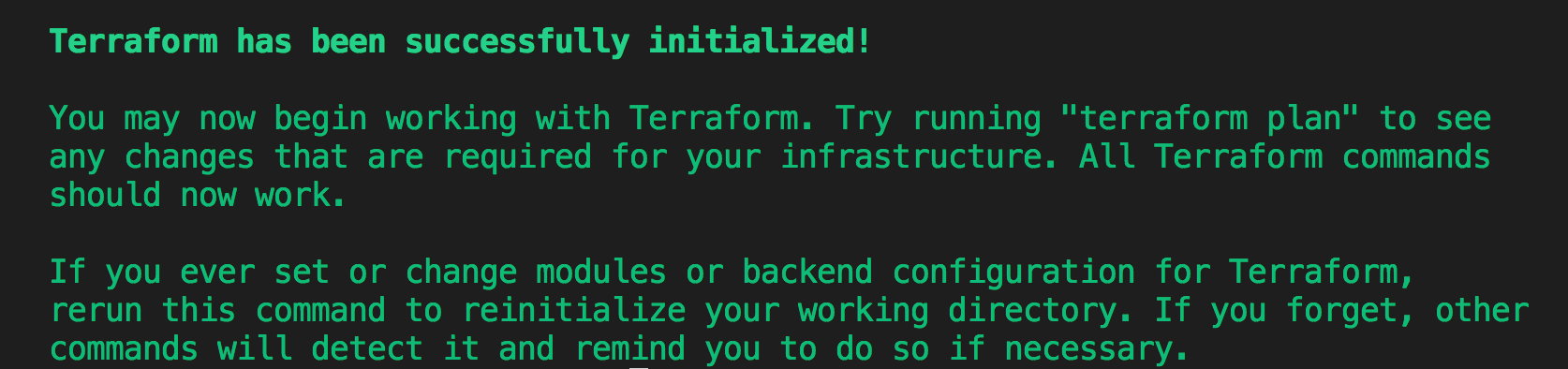


Set the target project for template deployment via command line:

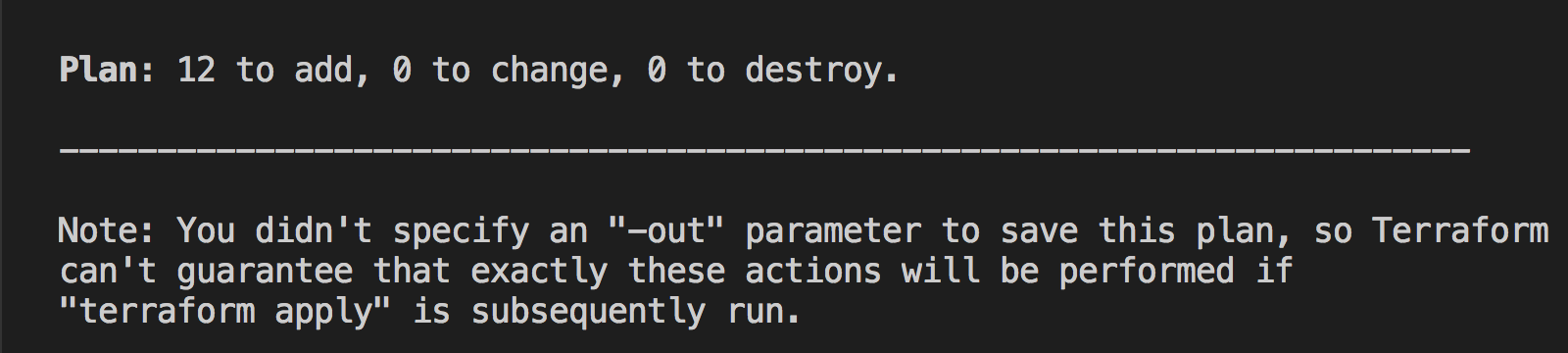
$ gcloud config set project my\_Project\_id

Run Terraform Commands:

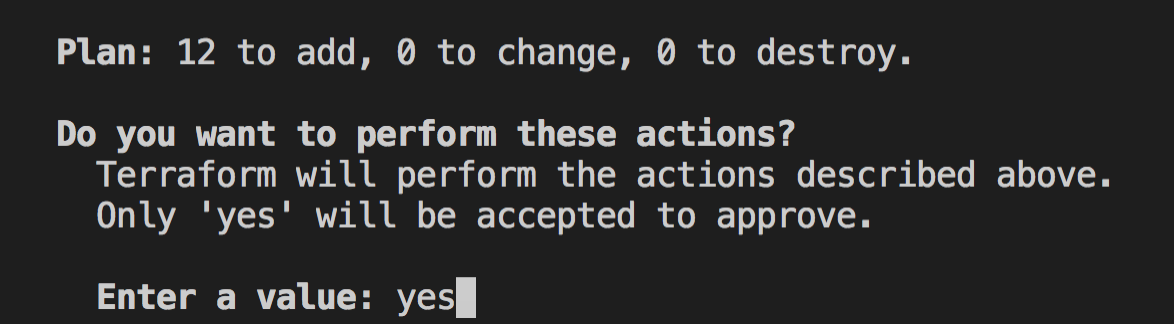
Initiate template deployment using command “terraform init”.



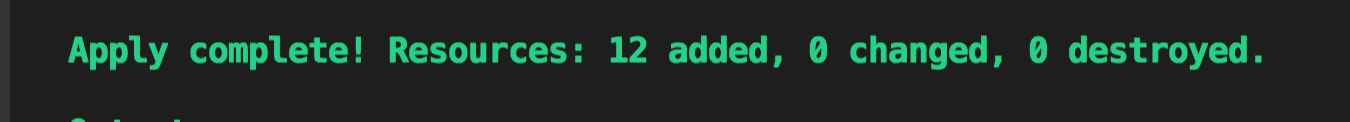
Once the terraform init has completed run the command “terraform plan”.



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.

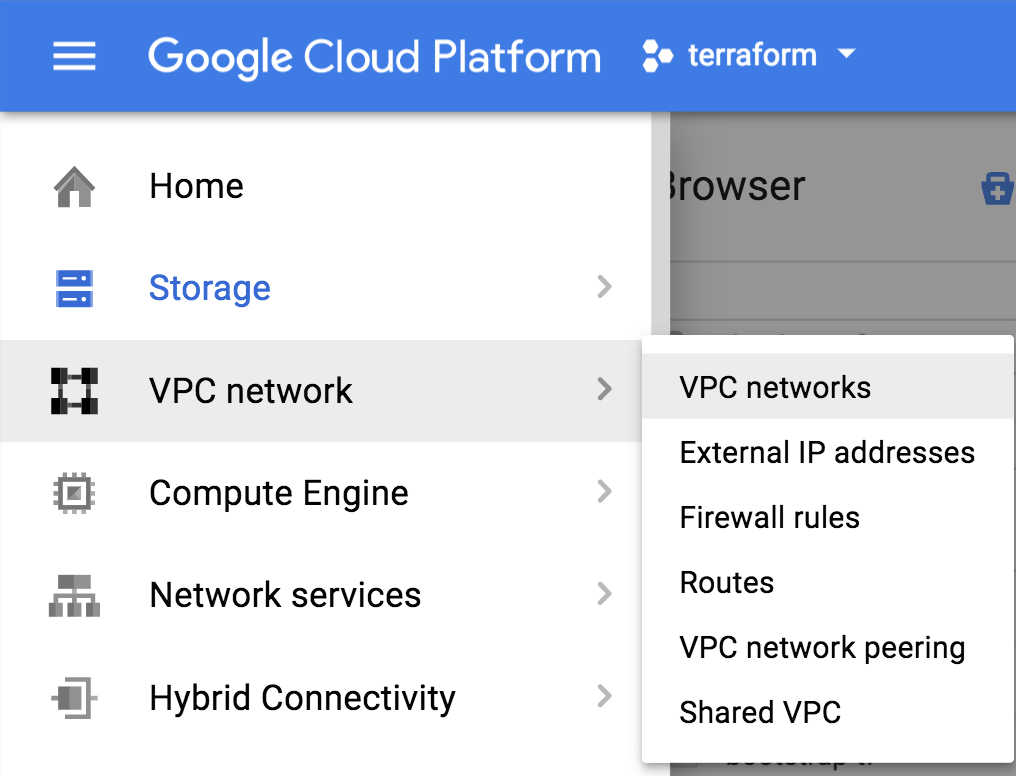


If all goes well, Terraform will report success (“Apply Complete!” and no errors):

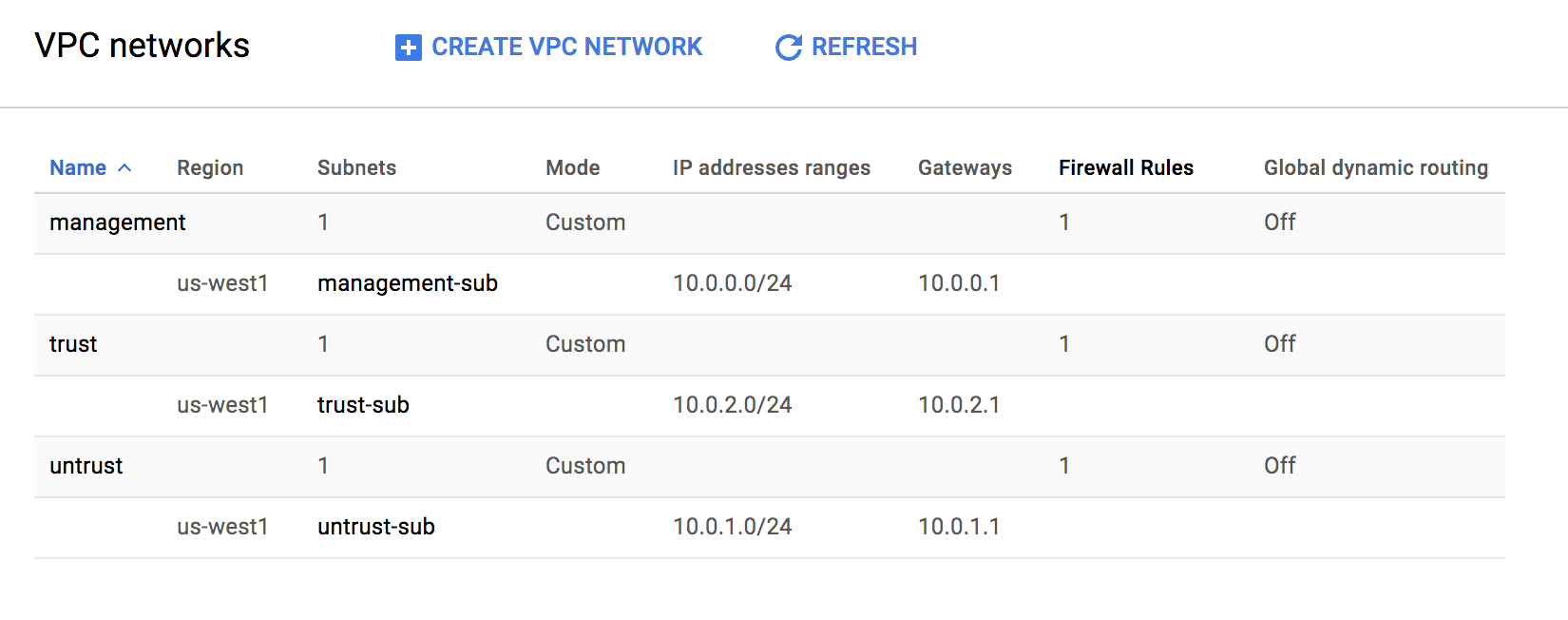


# Review what was created

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

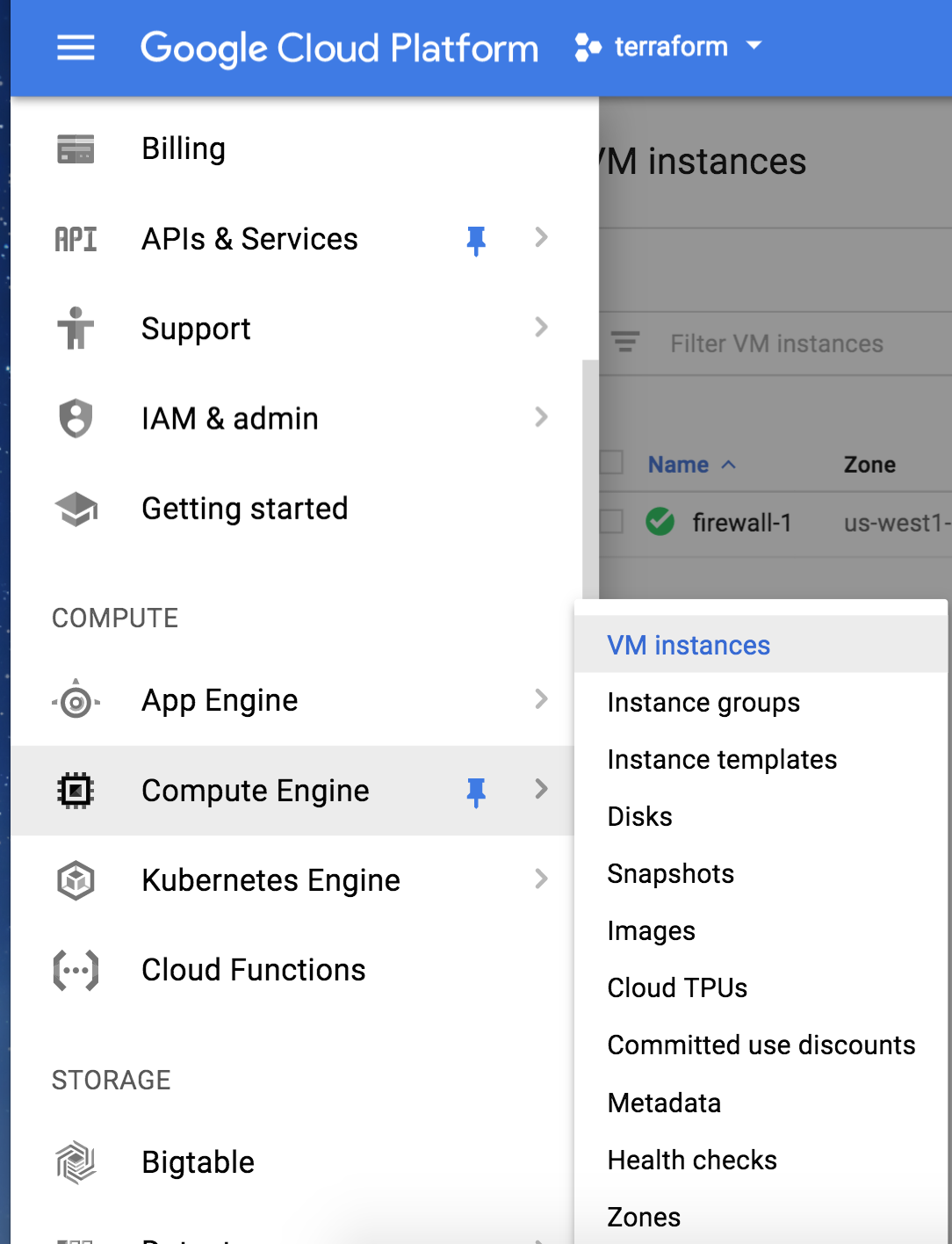


The template creates four networks: management-network, trust-network, and untrust-network.

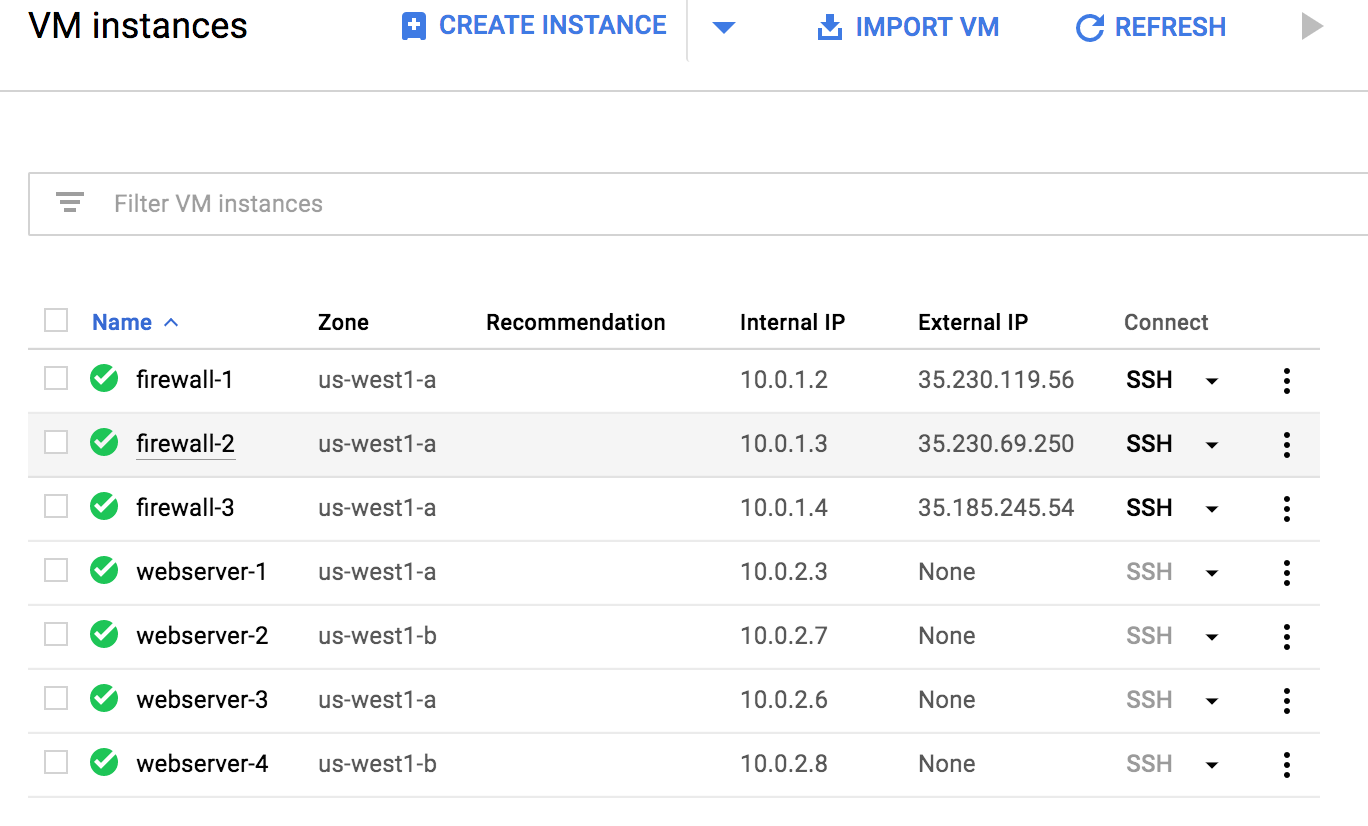


Note: A default network is automatically created when a GCP Project is instantiated. This default network can be ignored or deleted.

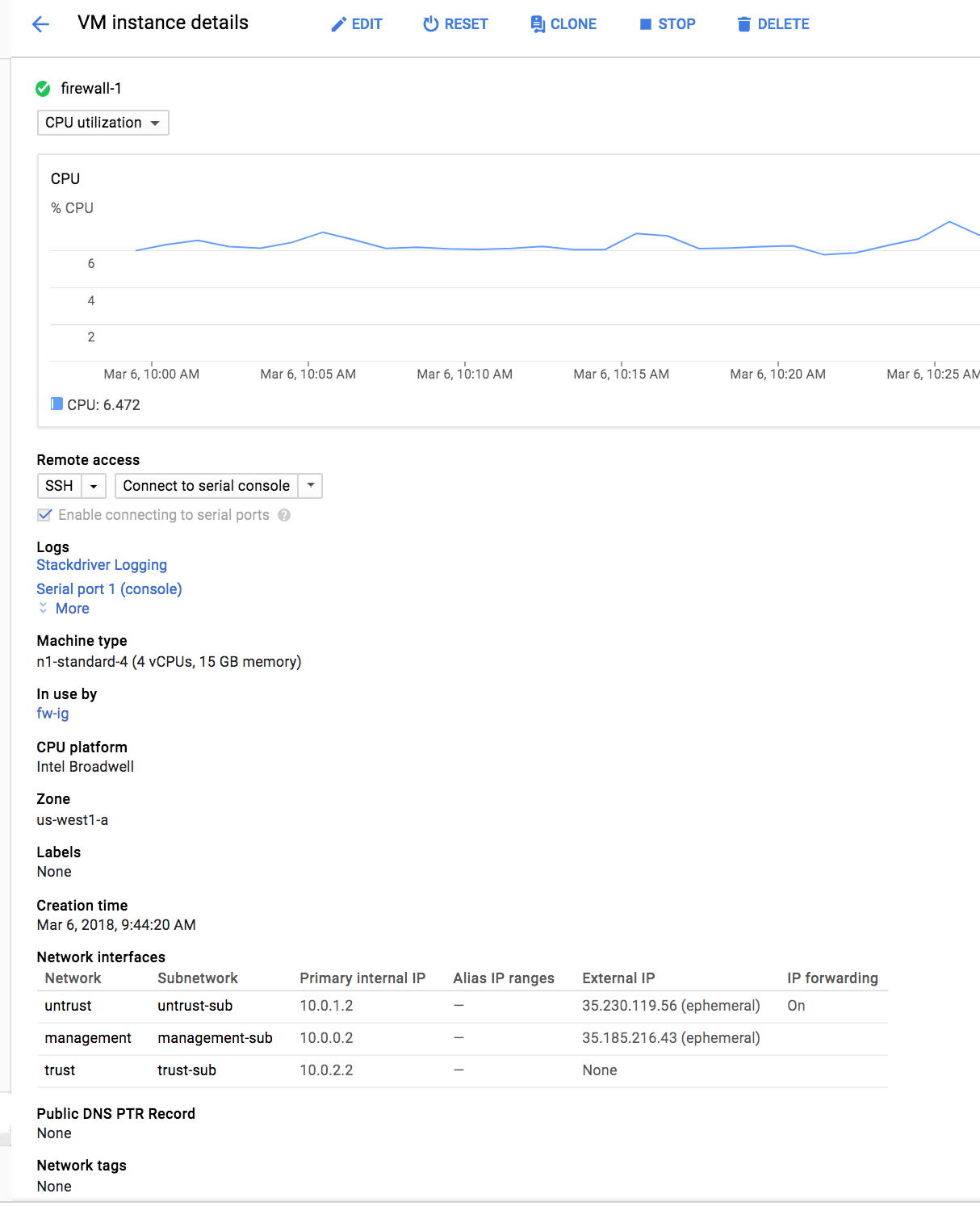
Deployed hosts can be viewed by navigating to **Compute Engine > VM Instances**:



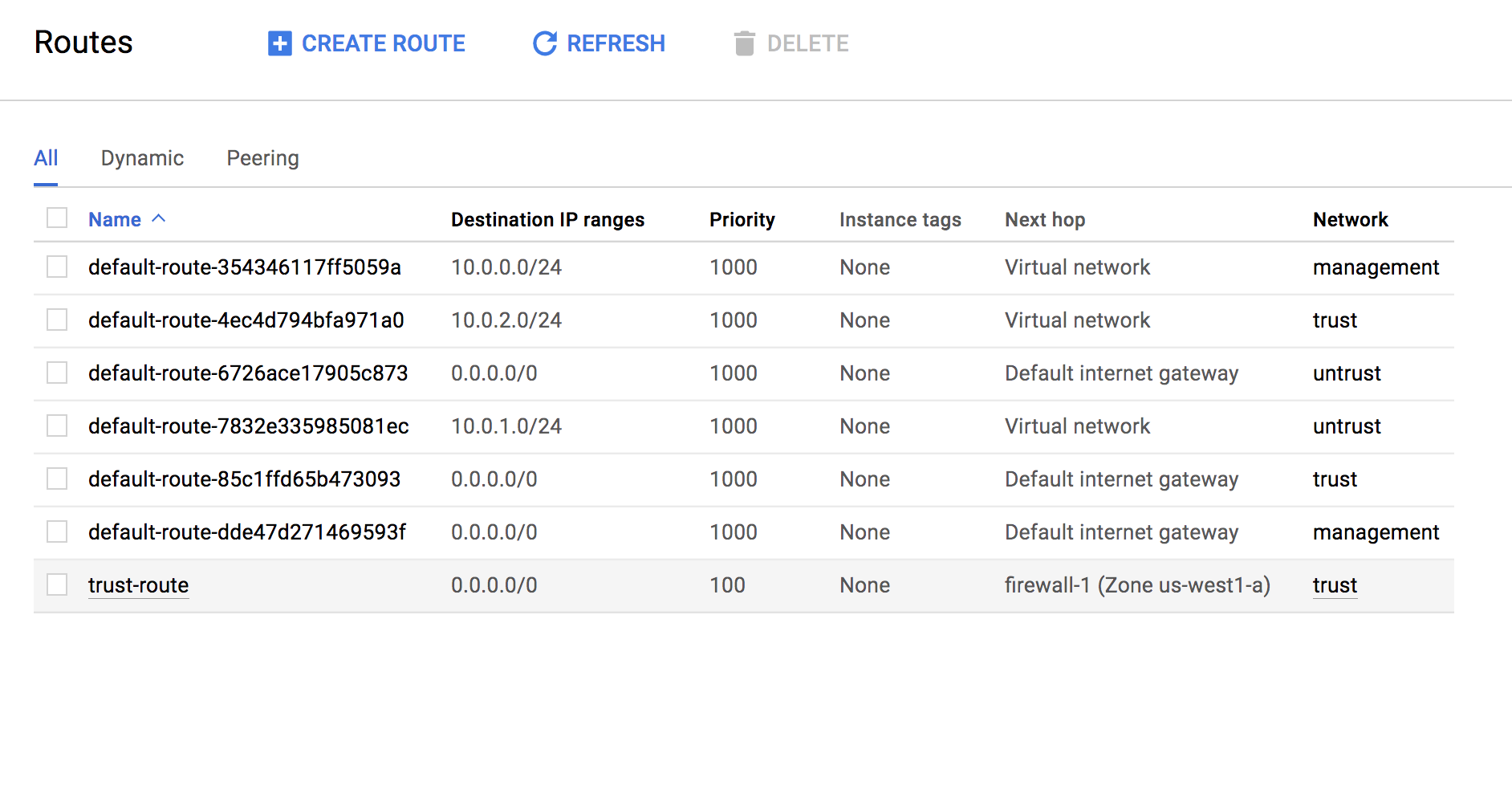
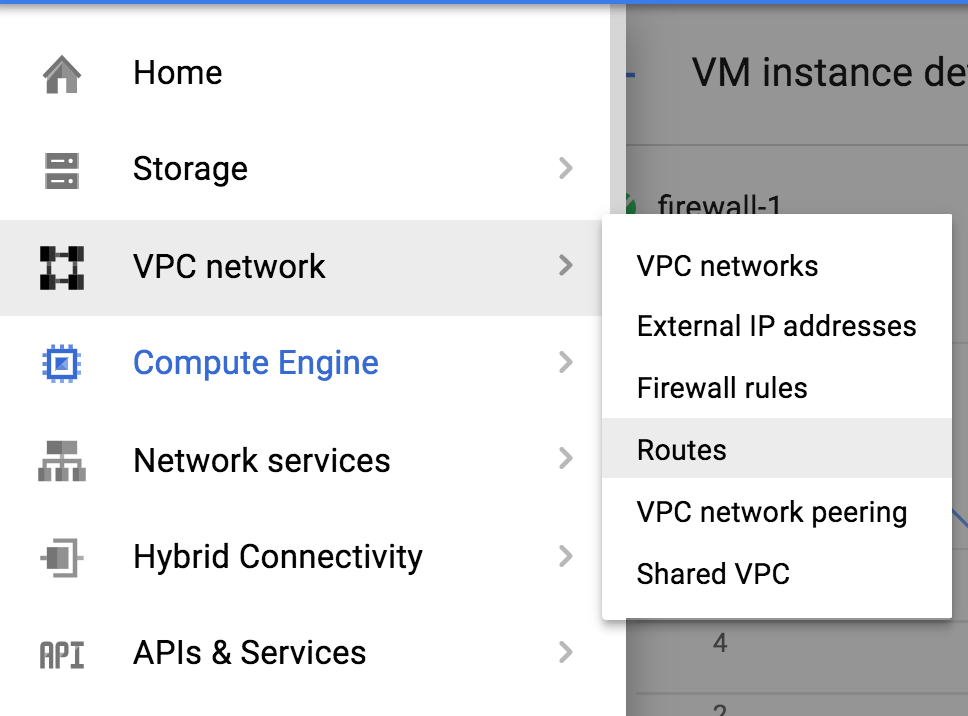
High-level information regarding the deployed instances are available with the default view:



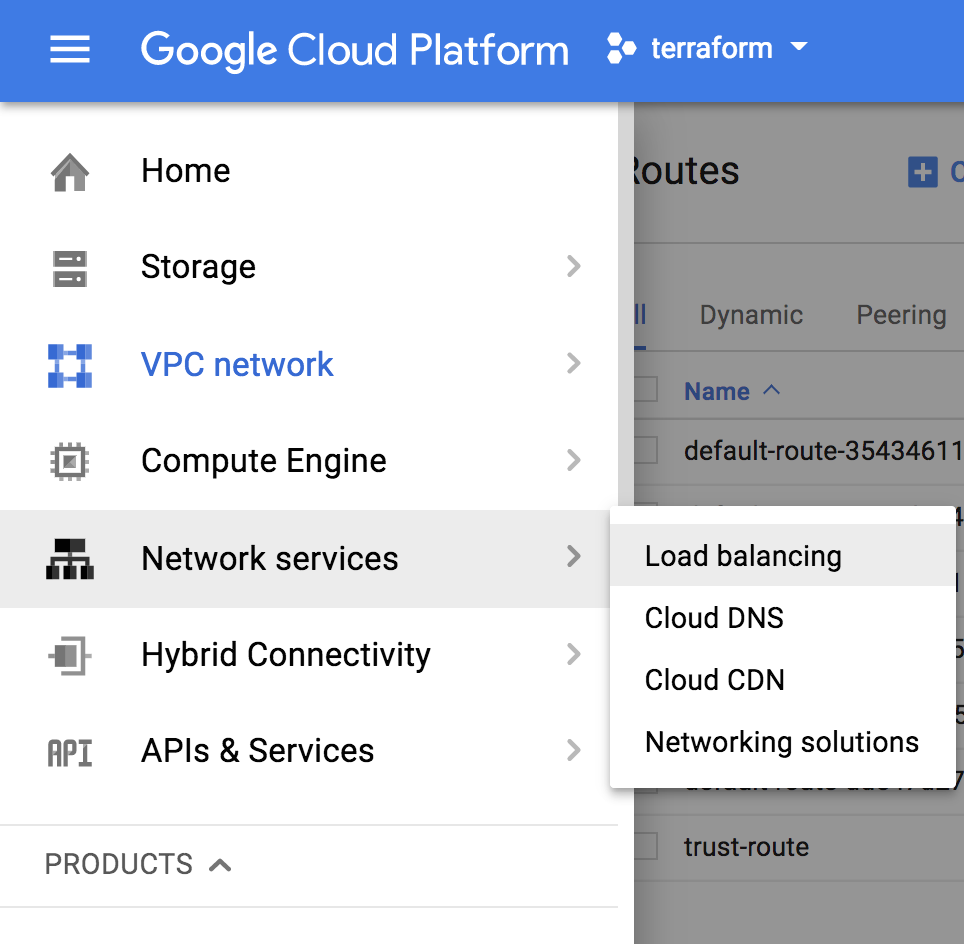
Also note the order in which the networks are attached to the firewalls. Click on firewall-1 and scroll down to see the network order.

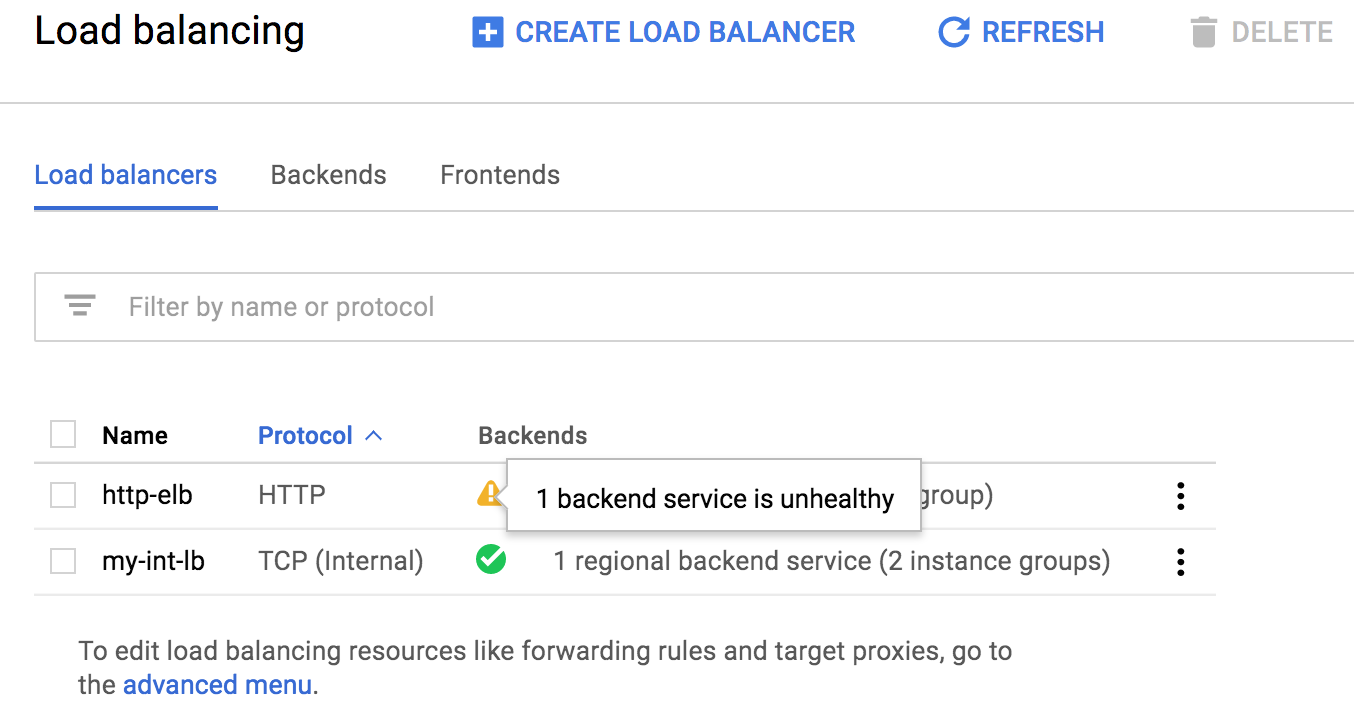


NOTE: The untrust network is first. The GCP Load Balancers only communicate with the lowest numbered interface on a VM. During the bootstrap phase of deployment the init-cfg.txt told the VM-Series firewall to perform a management interface swap. Therefore, we must have the GCP networks in this order.



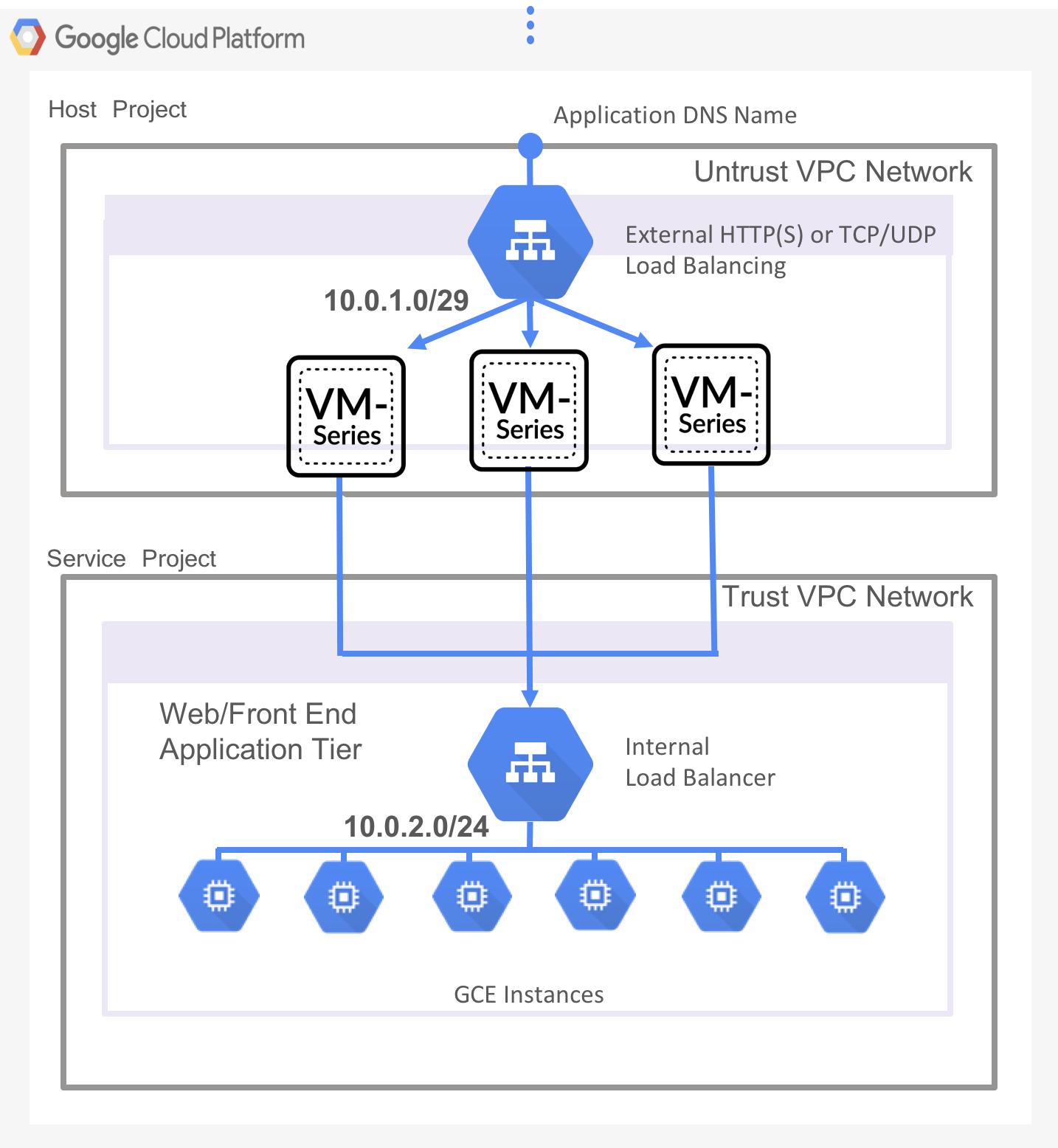
Lastly check your newly deployed Load Balancers by navigating to Network Services then select Load Balancing.





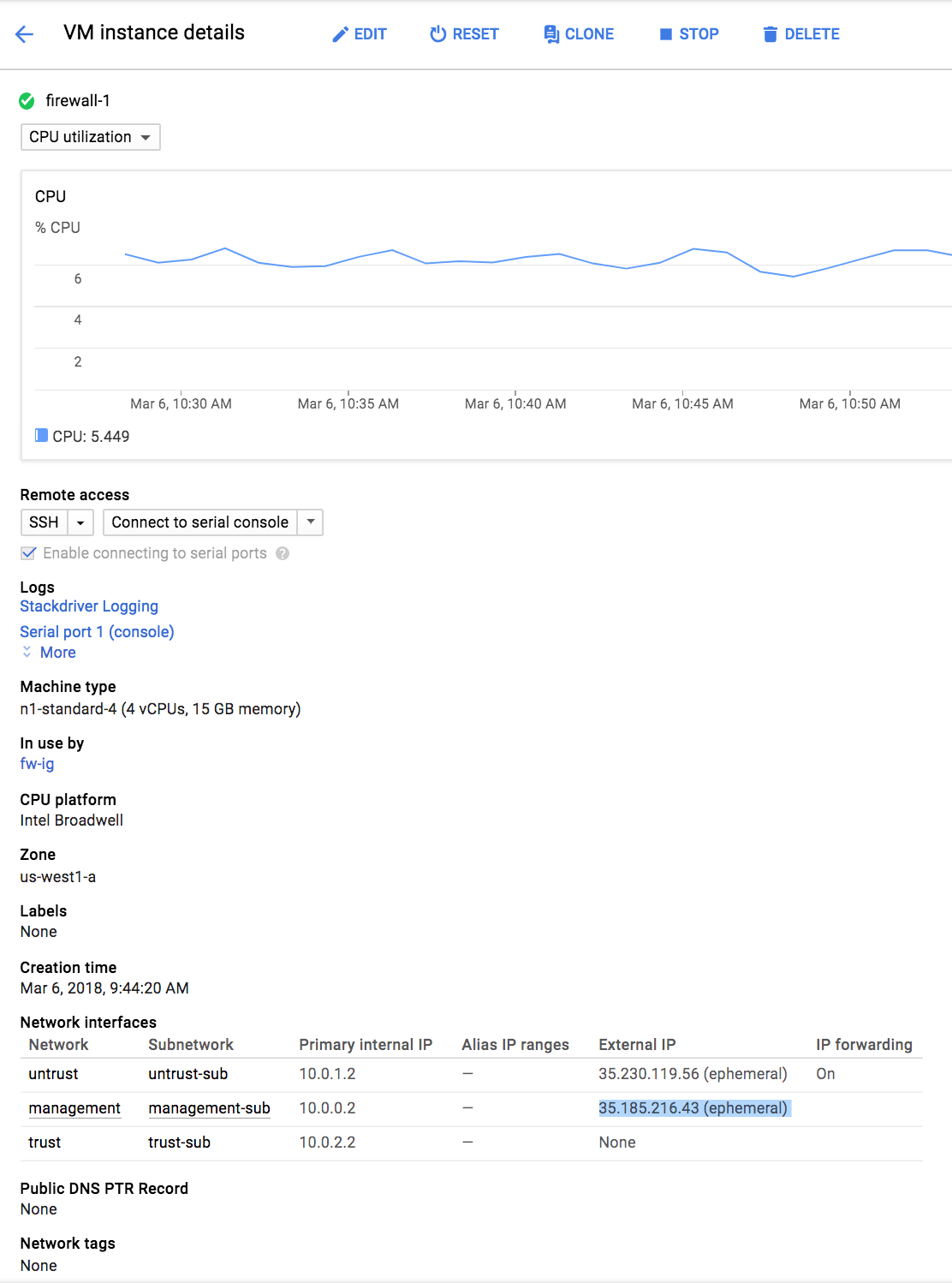
NOTE: You should see two load balancers configured. Both LBs should show all backend services as healthy.

All of this matches the topology shown previously:

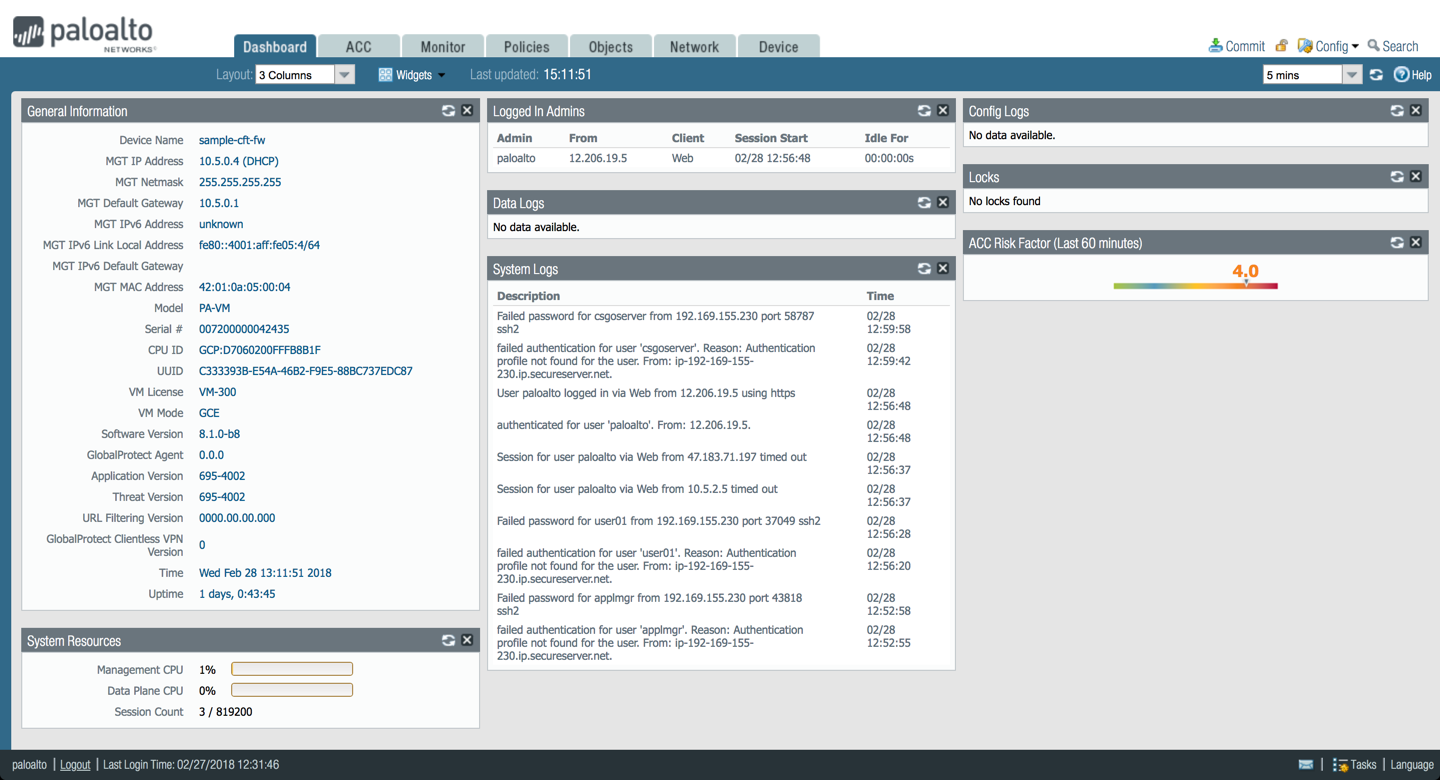


# Access the firewall

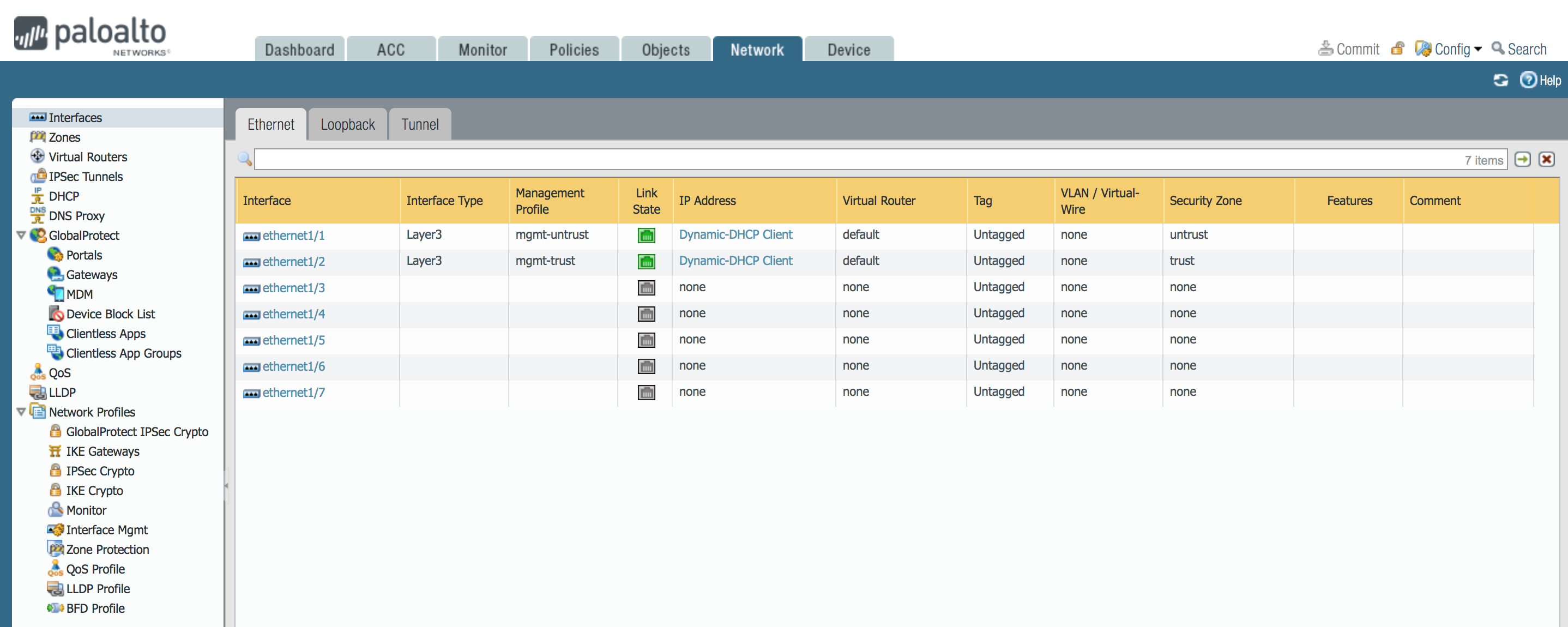
**NOTE:** Bootstrapping a VM-Series firewall takes approximately 7-9 minutes. Be patient ☺ Once the template has been deployed successfully, it may be a while before the VM-Series firewall is up and you are able to log into the VM-Series firewall by browsing to the Management public IP Address. Recall we swapped the Management interface so you will need to click on the VM Series to get the Public IP address.



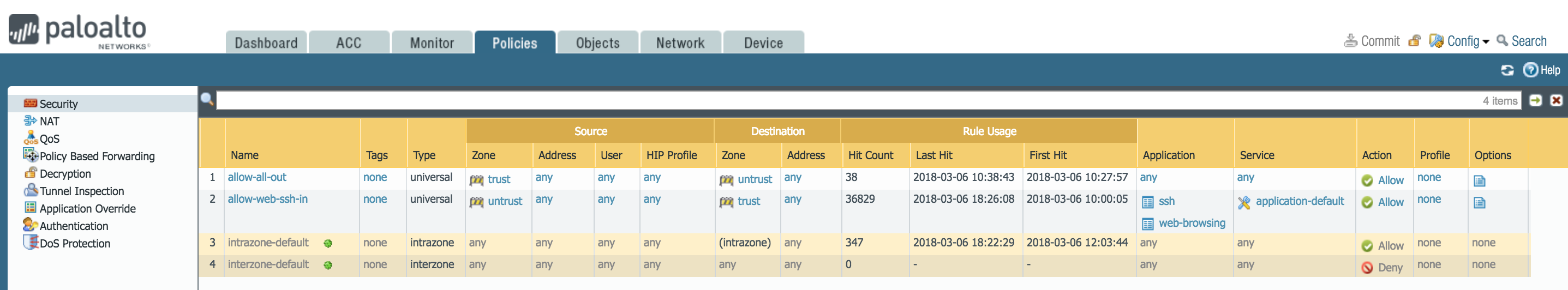
You should now be able to browse to the VM-Series firewall and login using the **username: paloalto** and password: **Pal0Alt0@123**



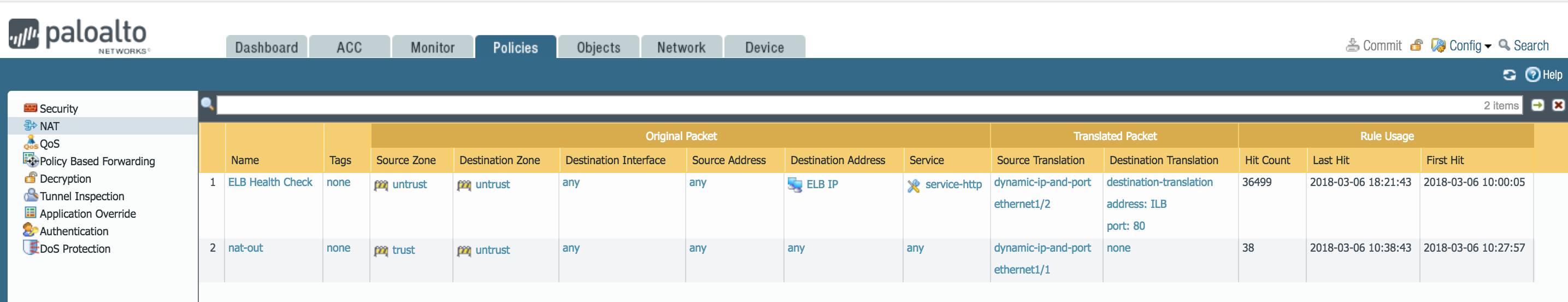
Here are the interfaces to zone mappings.

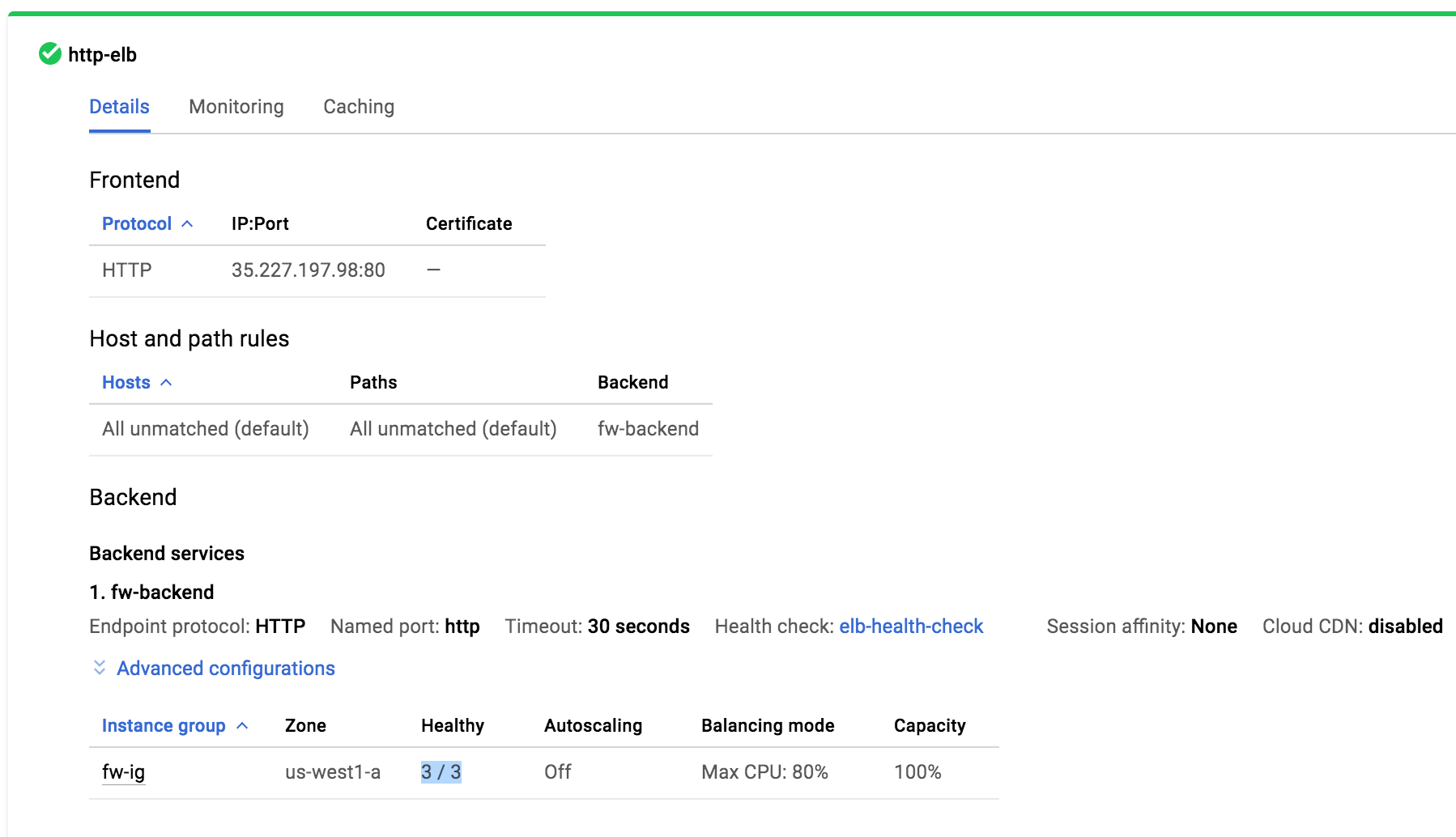


In the policies tab you can review the security policies:



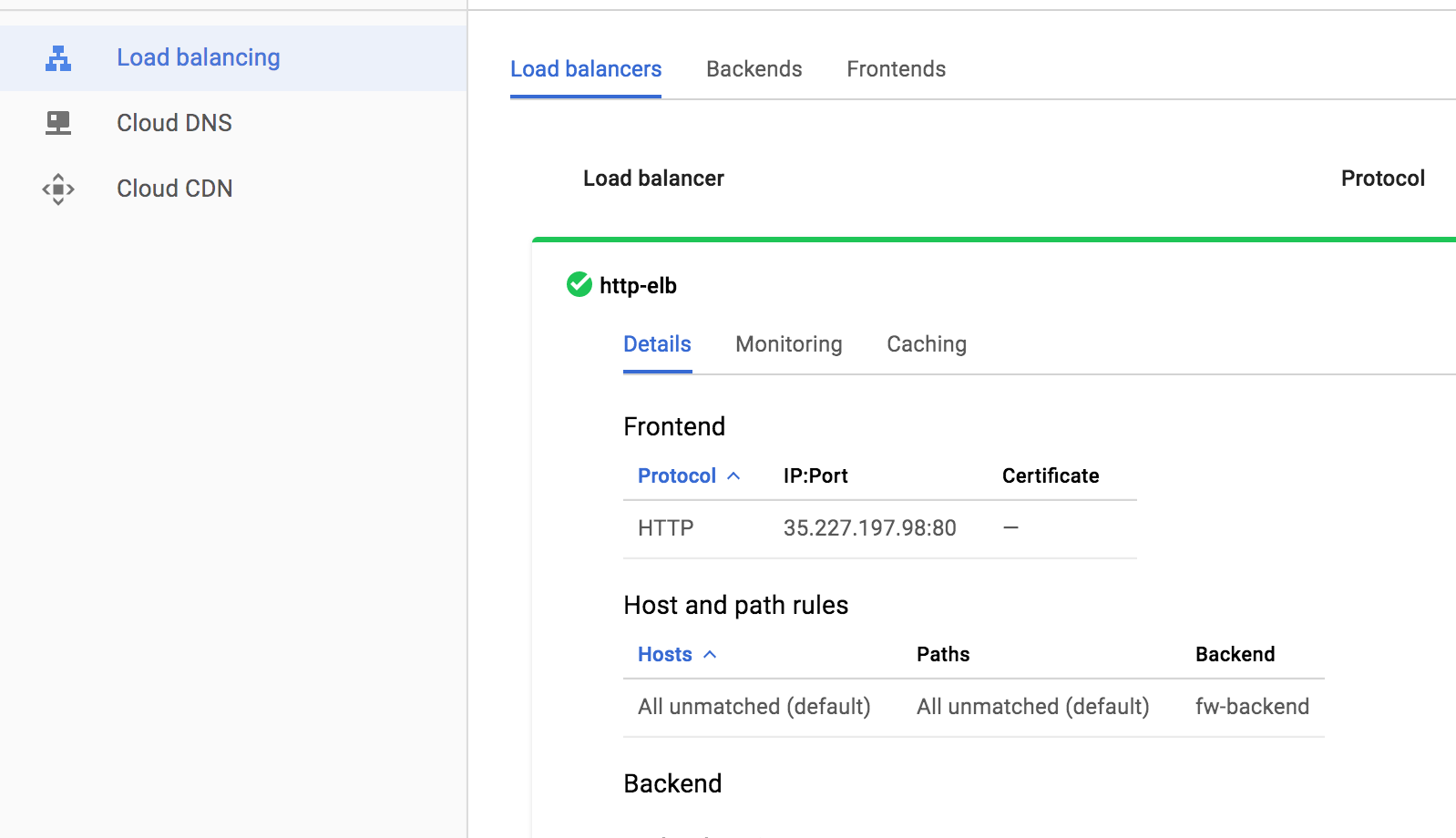
Under the polices tab select NAT on the left side. We need to have a health check from the ELB through to the backend web server of the ILB. In order for the ELB health checks to flow through the VM-Series firewall then to and through the ILB to the web servers we need the ELB Health Check NAT statements.





# Access the Webservers via ELB

Open a browser and browse to the IP address of the ELB. The IP of the ELB can be found under load balancers then expand the ELB. Each Webserver will provide their name in the browser upon reloading the browser. This lets you know the Load Balancer is working.



You have now successfully deployed a Terraform template with a VM-Series firewall for AutoScaling in GCP.

# Cleanup

## Delete the deployment

Once done, cleanup as follows:

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

# 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. There could be several reasons:

#### Corrupt configuration files

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

#### Incorrect bootstrap bucket-name

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

#### Scope is incorrect

You must have the correct scope permissions as part of your Terraform deployment.