GCP Transit VPC with Advanced Peering and East/West traffic using Load Balanced VMSeries firewalls.

Terraform Build Guide

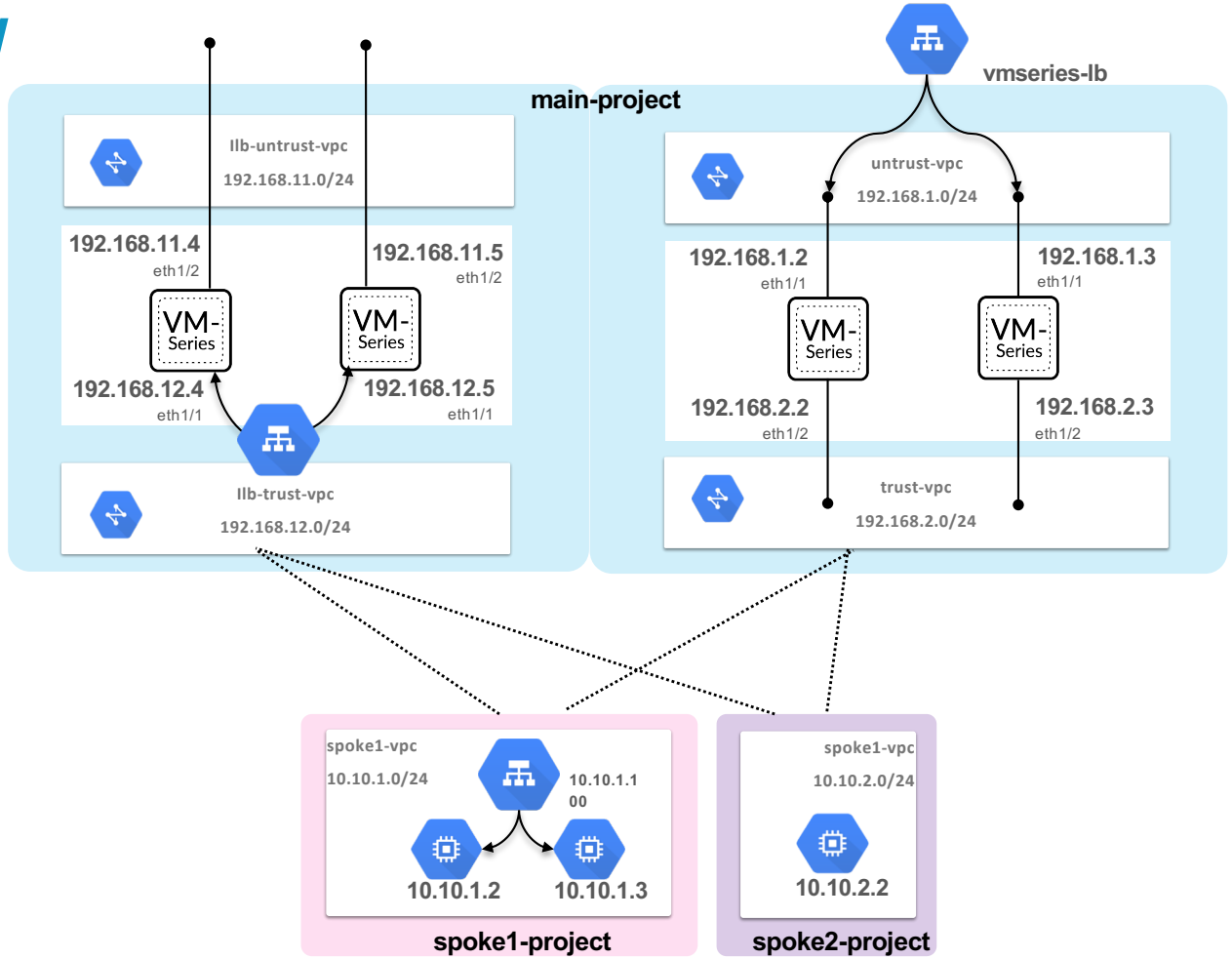
SUPPORT POLICY

This 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) or sites other than our official Downloads page on https://support.paloaltonetworks.com are provided under the best effort policy.

DEPLOYMENT OVERVIEW

- Terraform builds 4 VM-Series firewalls and two peered VPCs. 2 VM-Series provide N/S resiliency for inbound traffic while another 2 provide E/W and outbound resiliency.
- Spoke1 VPC has 1 internal load balancer and 2 backend Web servers (configured with Apache)
- Spoke2 VPC has 1 Linux host.
- spoke1-vpc & spoke2-vpc can be deployed into the same project as the VM-Series or in different projects.

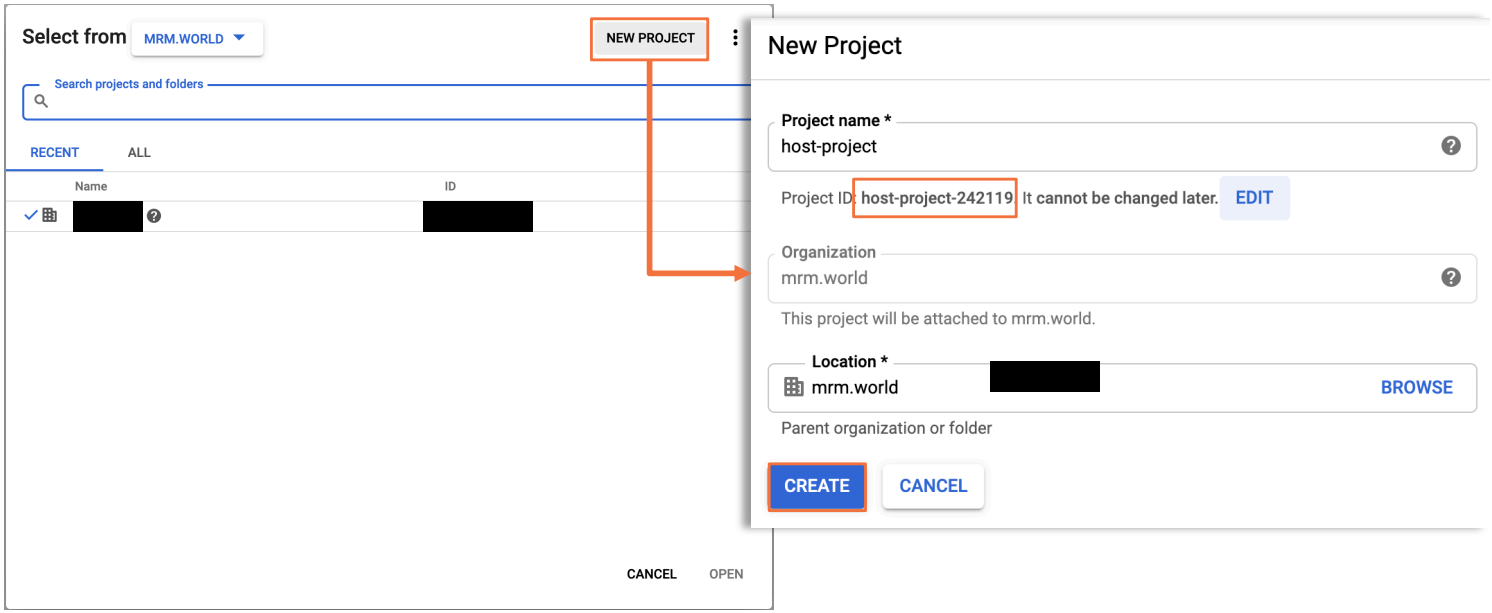
NOTE: There are a few steps that need to be completed after the Terraform deployment. This is because when this was created, the terraform provider did not yet support the functionality.



CONFIGURE GCP API & RETRIEVE API CREDENTIALS

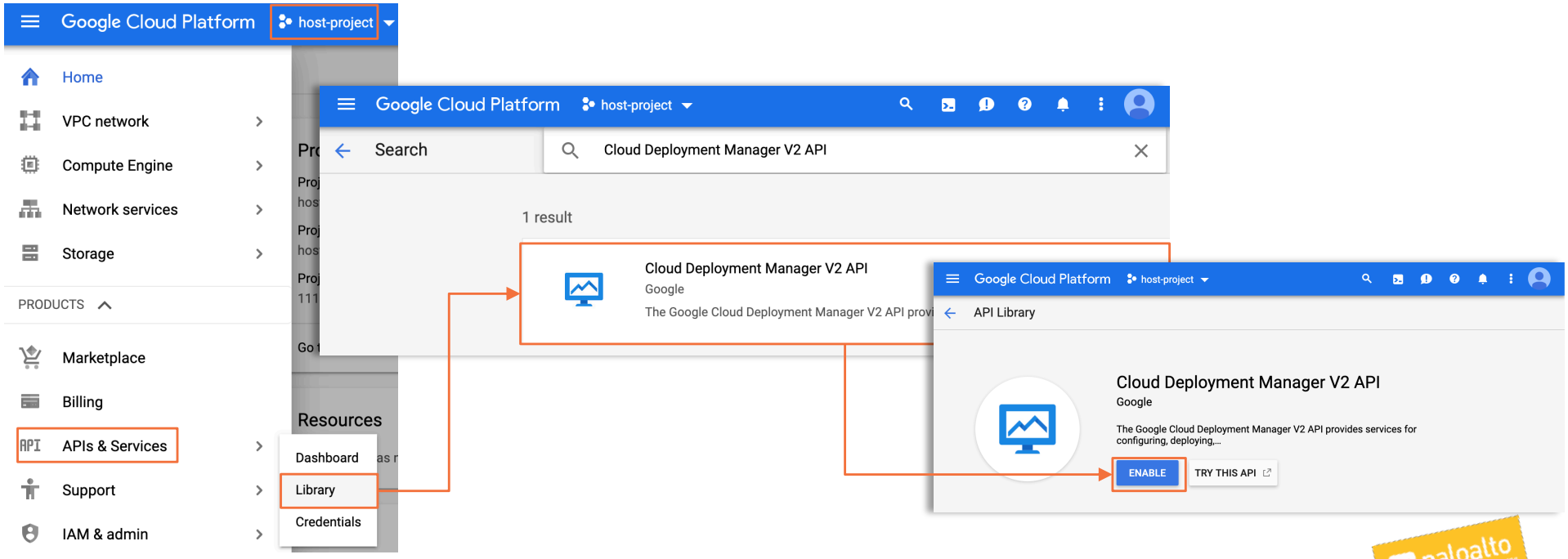
STEP 1. CREATE A PROJECT

1. Create a GCP Project
2. Record the Project ID.



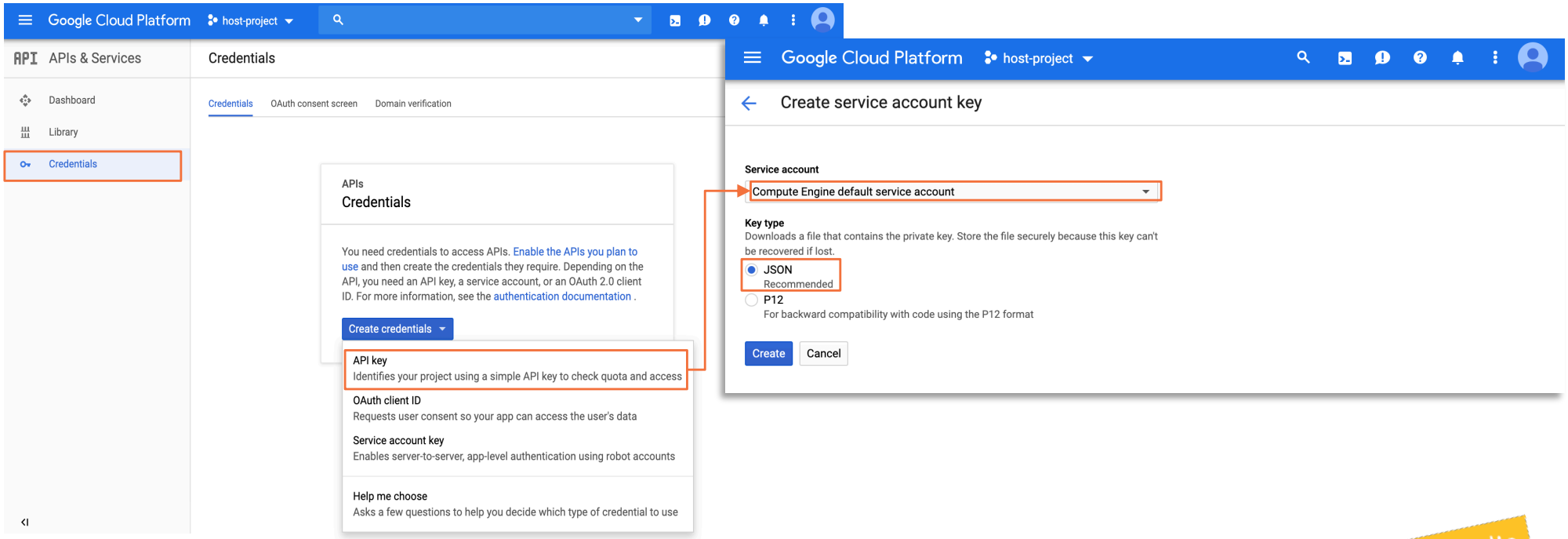
STEP 2. ENABLE GOOGLE COMPUTE API

1. Go to **API & Services → Library**
2. Search for **Cloud Deployment Manager V2 API** and click **Enable**



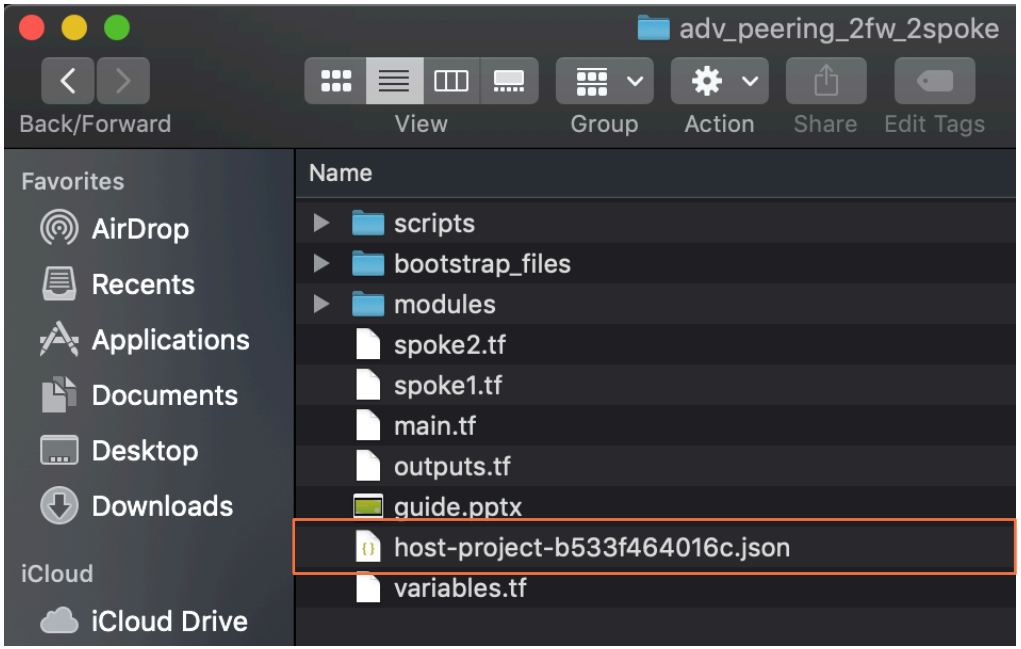
STEP 3. RETRIEVE API CREDENTIALS

1. Go to **API & Services → Credentials → Create Credentials → Service account key**
2. Select **Compute Engine default service account** and select **JSON** as the key type



STEP 4. RETRIEVE API CREDENTIALS

1. Move the downloaded key into the main directory of the Terraform build.
2. Download the Terraform build from

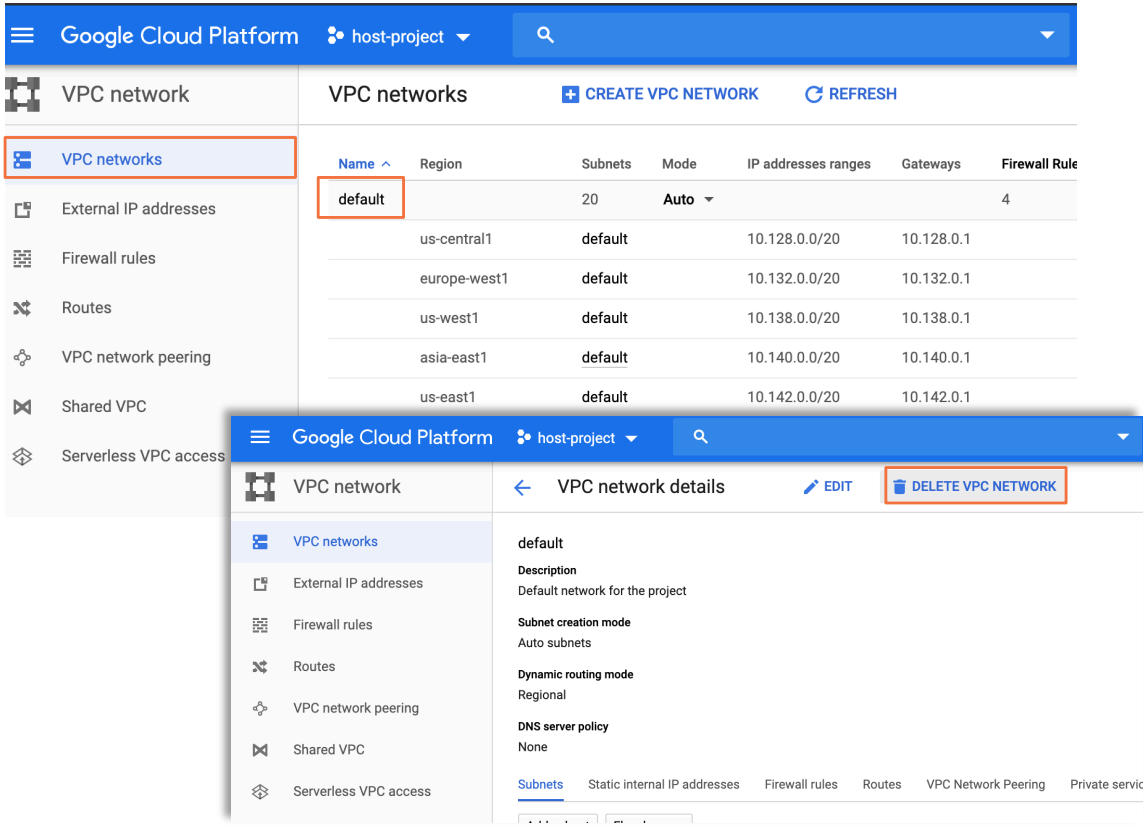


Repeat STEPS1-4 if you want Spoke1 & Spoke2 to reside in different projects than the VM-Series

DELETE DEFAULT NETWORK

Every new project has a default VPC. Each project has a soft maximum of 5 VPCs.

If you are deploying everything to the same project, make sure you either delete the default VPC in the project or ask for a quota increase.



EDIT VARIABLES.TF

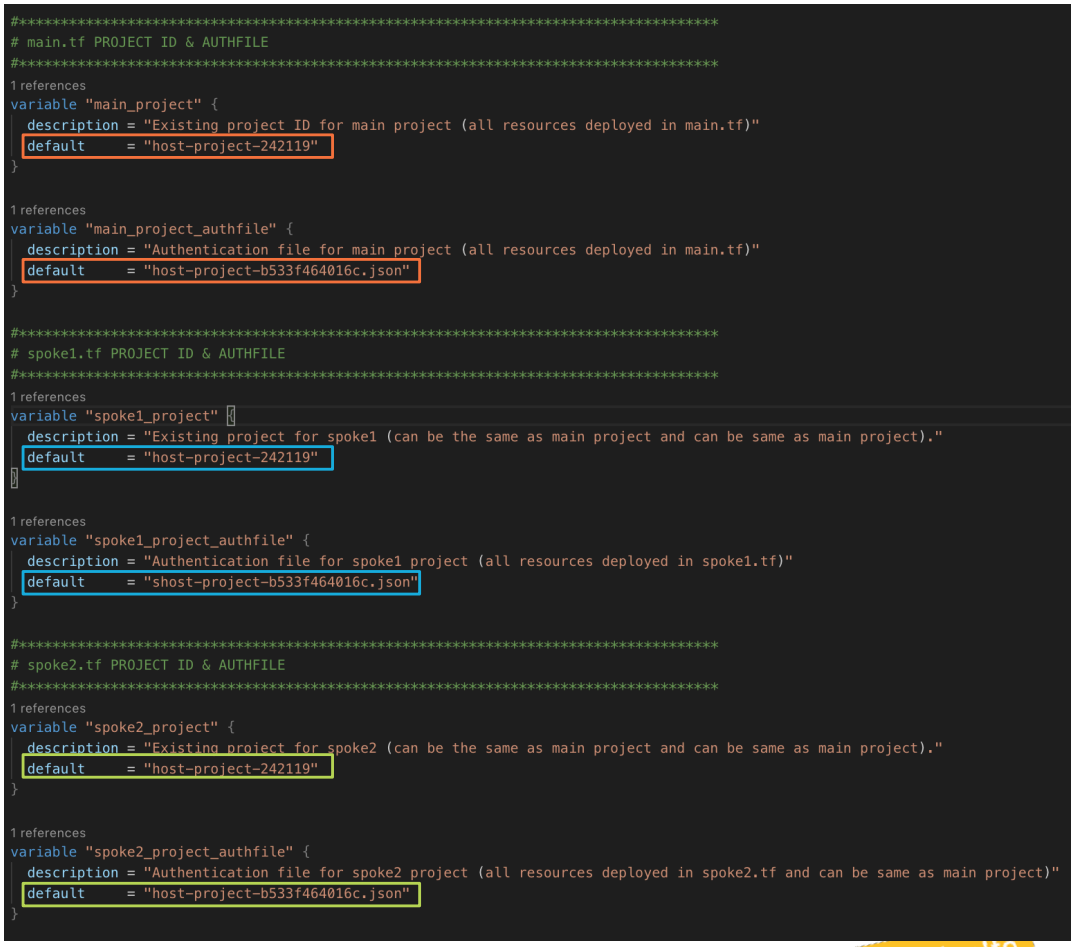
STEP 5. ADJUST VARIABLES.TF

1. Open **variables.tf** in a text editor.

2. Enter the project ID for each project in:
   - main_project
   - spoke1_project
   - spoke2_project

3. Enter the corresponding key file for the projects in:
   1. main_project_authfile
   2. spoke1_project_authfile
   3. spoke2_project_authfile

4. In this example, we are deploying everything to the same project (host-project-242119), so the project ID and authfile value will be the same for main, spoke1, and spoke2 environments.



```
#*******************************************************************************
# main.tf PROJECT ID & AUTHFILE
#*******************************************************************************
1 references
variable "main_project" {
  description = "Existing project ID for main project (all resources deployed in main.tf)"
  default     = "host-project-242119"
}

1 references
variable "main_project_authfile" {
  description = "Authentication file for main project (all resources deployed in main.tf)"
  default     = "host-project-b533f464016c.json"
}

#*******************************************************************************
# spoke1.tf PROJECT ID & AUTHFILE
#*******************************************************************************
1 references
variable "spoke1_project" {
  description = "Existing project for spoke1 (can be the same as main project and can be same as main project)."
  default     = "host-project-242119"
}

1 references
variable "spoke1_project_authfile" {
  description = "Authentication file for spoke1 project (all resources deployed in spoke1.tf)"
  default     = "shost-project-b533f464016c.json"
}

#*******************************************************************************
# spoke2.tf PROJECT ID & AUTHFILE
#*******************************************************************************
1 references
variable "spoke2_project" {
  description = "Existing project for spoke2 (can be the same as main project and can be same as main project)."
  default     = "host-project-242119"
}

1 references
variable "spoke2_project_authfile" {
  description = "Authentication file for spoke2 project (all resources deployed in spoke2.tf and can be same as main project)"
  default     = "host-project-b533f464016c.json"
}
```

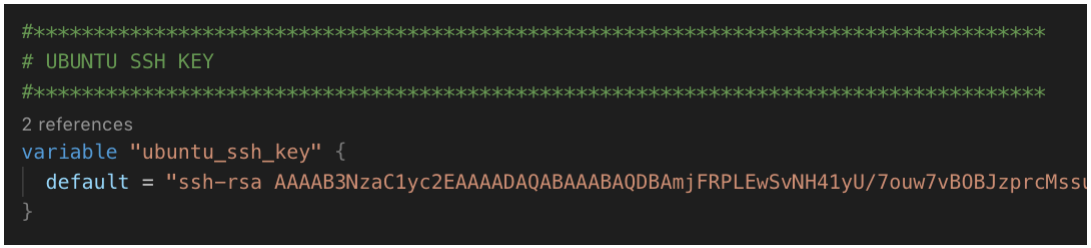
STEP 6. SSH KEY FOR UBUNTU VM & VM-SERIES LICENSE TYPE

1. Create an SSH key for instances in the Spoke VPCs.

```
$ ssh-keygen -t rsa -f ~/.ssh/ubuntukey -C ubuntu
```

<enter passphrase x 2>

```
$ chmod 600 ~/.ssh/ubuntukey
$ cat ~/.ssh/ubuntukey.pub
```



2. Copy CAT output and paste it as the default value for **ubuntu_ssh_key** inside **variables.tf**

3. Uncomment the **vmseries_image** to the license SKU that you want.

SAVE VARIABLES.TF

RUN TERRAFORM

STEP 7. RUN TERRAFORM

1. terraform init

```
adv_peering_2fw_2spoke mmclimans$ 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.
```

2. terraform apply

```
adv_peering_2fw_2spoke mmclimans$ terraform apply
…
…
…
Plan: 49 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
```

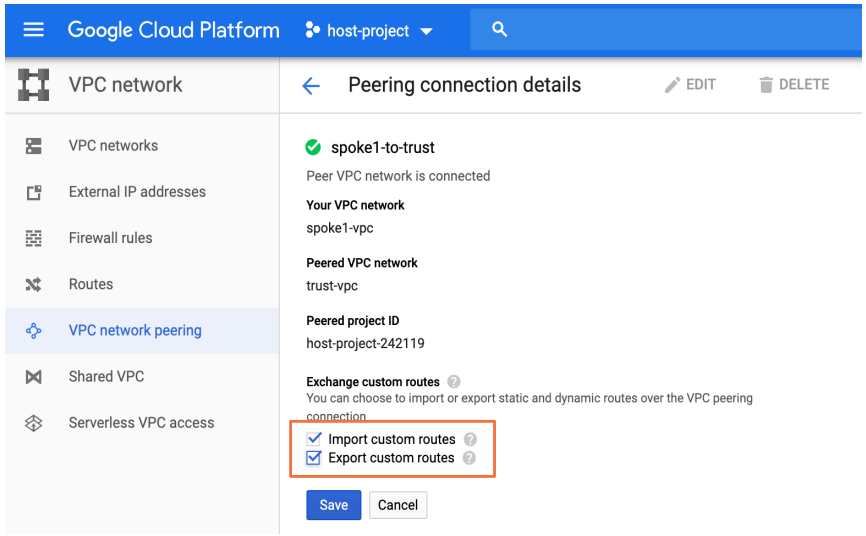
You will receive this output once the deployment has completed.

```
Apply complete! Resources: 74 added, 0 changed, 0 destroyed.

Outputs:

                              IMPORTANT!! PLEASE READ!!                                    = [
    ================================================================================,
    Before proceeding, you must enable import/export custom routes on all peering links,
    and remove the default (0.0.0.0/0) route from TRUST, SPOKE1, and SPOKE2 VPCs,
    There is also a need to create a default route in the ilb-trust-subnet pointing to,
    the internal load balancer,
    ================================================================================
]
GLB-ADDRESS     = http://34.102.167.223
MGMT-URL-FW1    = https://35.239.219.110
MGMT-URL-FW2    = https://35.232.77.73
MGMT-URL-ILB-FW3  = https://34.67.53.20
MGMT-URL-ILB-FW4    = https://34.68.219.215
SSH-SPOKE1-VM1-FW1 = ssh ubuntu@35.224.112.224 -p 221 -i <INSERT KEY>
SSH-SPOKE1-VM1-FW2 = ssh ubuntu@34.68.152.104 -p 221 -i <INSERT KEY>
SSH-SPOKE1-VM2-FW1 = ssh ubuntu@35.224.112.224 -p 222 -i <INSERT KEY>
SSH-SPOKE1-VM2-FW2 = ssh ubuntu@34.68.152.104 -p 222 -i <INSERT KEY>
SSH-SPOKE2-FW1 = ssh ubuntu@35.224.112.224 -p 223 -i <INSERT KEY>
SSH-SPOKE2-FW2 = ssh ubuntu@34.68.152.104 -p 223 -i <INSERT KEY>
```

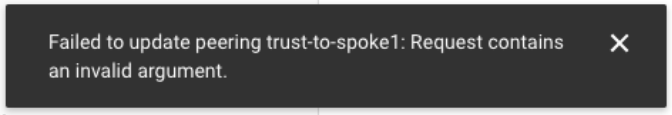
STEP 8. ENABLE IMPORT/EXPORT CUSTOM ROUTES

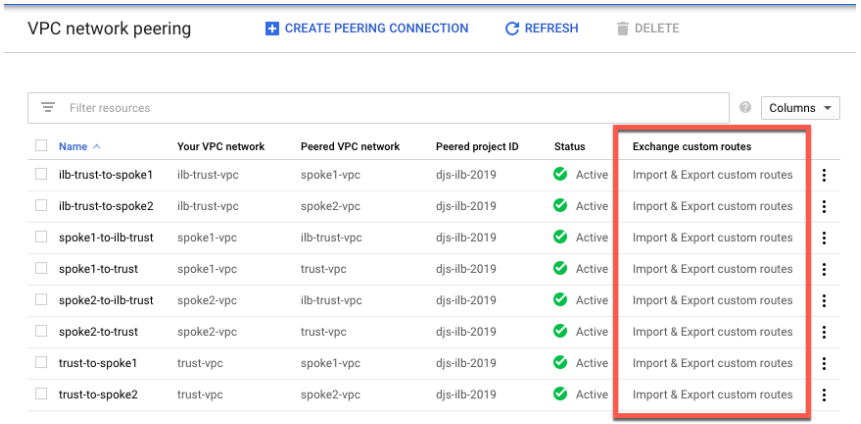
Go to: **VPC Network → VPC network peering**

- For **EACH PEER**, enable **Import custom routes** & **Export custom routes**



NOTE: If you see the following error, you are clicking too fast. **SLOW DOWN.** Wait a few seconds and click Save again.



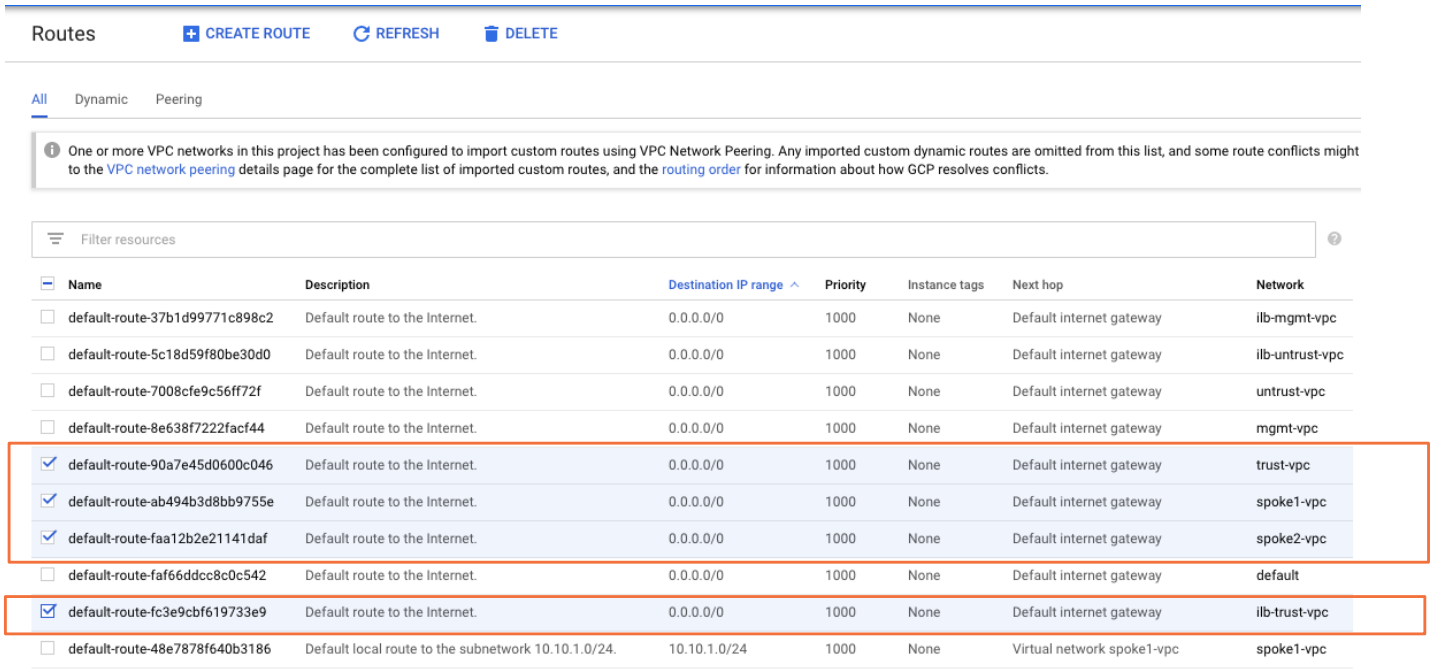


When done, your peering connections should look like this.

STEP 9. DELETE TRUST & SPOKE DEFAULT INTERNET ROUTES

Go to: **VPC Network → Routes**

- Delete **spoke1-vpc**, **spoke2-vpc**, & **trust-vpc** default route to the internet.



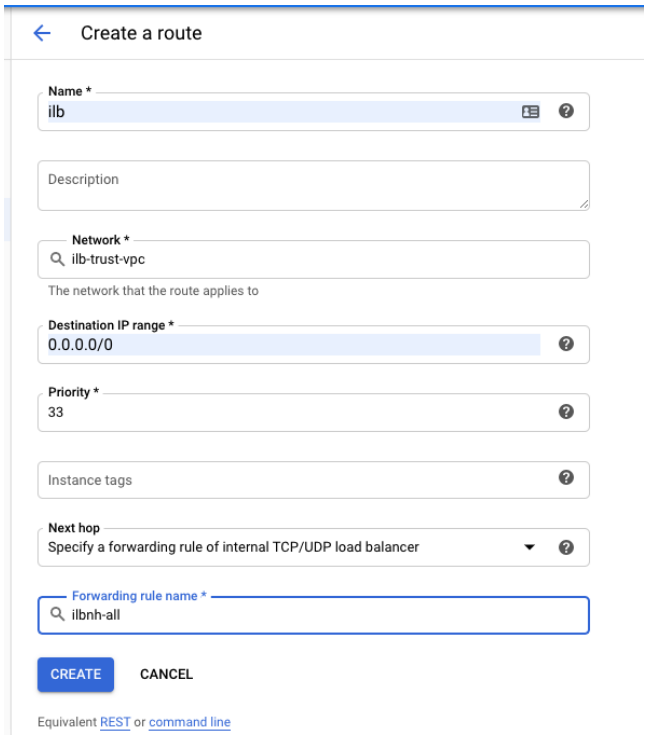
Name	Description	Destination IP range	Priority	Instance tags	Next hop	Network
default-route-37b1d99771c898c2	Default route to the Internet.	0.0.0.0/0	1000	None	Default internet gateway	ilb-mgmt-vpc
default-route-5c18d59f80be30d0	Default route to the Internet.	0.0.0.0/0	1000	None	Default internet gateway	ilb-untrust-vpc
default-route-7008cfe9c56ff72f	Default route to the Internet.	0.0.0.0/0	1000	None	Default internet gateway	untrust-vpc
default-route-8e638f7222facf44	Default route to the Internet.	0.0.0.0/0	1000	None	Default internet gateway	mgmt-vpc
default-route-90a7e45d0600c046	Default route to the Internet.	0.0.0.0/0	1000	None	Default internet gateway	trust-vpc
default-route-ab494b3d8bb9755e	Default route to the Internet.	0.0.0.0/0	1000	None	Default internet gateway	spoke1-vpc
default-route-faa12b2e21141daf	Default route to the Internet.	0.0.0.0/0	1000	None	Default internet gateway	spoke2-vpc
default-route-faf66ddcc8c0c542	Default route to the Internet.	0.0.0.0/0	1000	None	Default internet gateway	default
default-route-fc3e9cbf619733e9	Default route to the Internet.	0.0.0.0/0	1000	None	Default internet gateway	ilb-trust-vpc
default-route-48e7878f640b3186	Default local route to the subnetwork 10.10.1.0/24.	10.10.1.0/24	1000	None	Virtual network spoke1-vpc	spoke1-vpc

NOTE: This is required because GCP Peering has a restriction that when a dynamic route overlaps with a subnet route in a peer network. For dynamic routes, the destination ranges that overlap with a subnet route from the peer network are silently dropped. GCP uses the subnet route. https://cloud.google.com/vpc/docs/vpc-peering

STEP 10. CREATE A DEFAULT ROUTE IN THE ILB-TRUST-VPC SUBNET

Go to: **VPC Network → Routes**

- Click Create Route and enter the following info to create a default route that will be propagated to the spoke subnets:



- Click Create

TEST TRAFFIC FLOWS

POST DEPLOYMENT VALIDATION

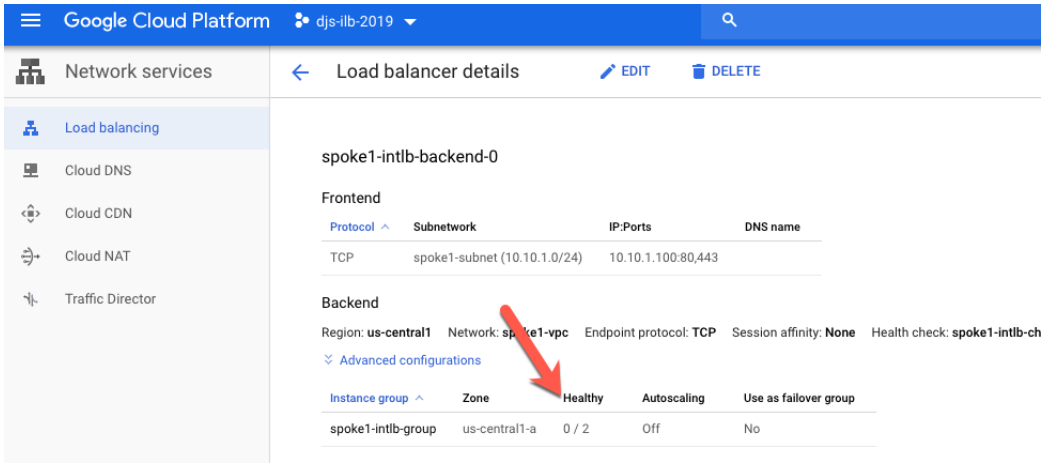
1. After the deployment is complete navigate to Network services > Load balancing and click the spoke1-intlb-backend-0

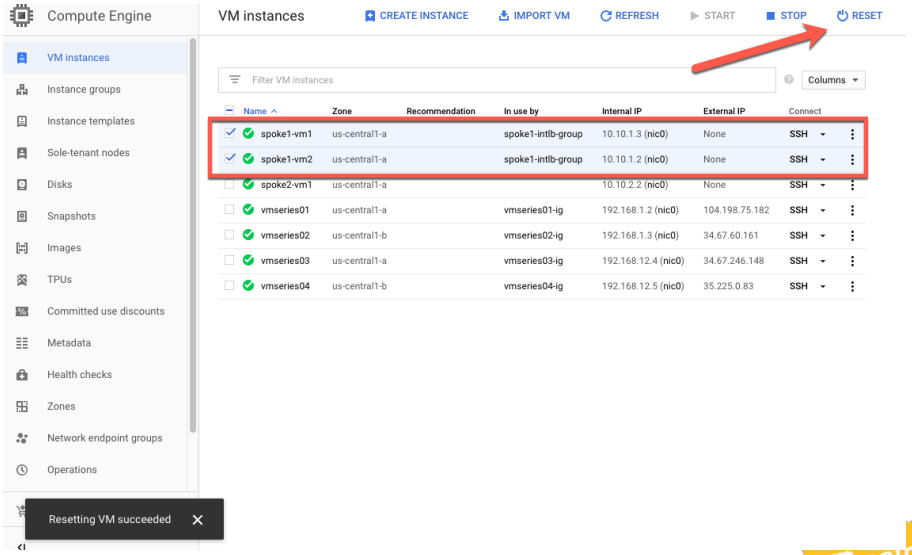
2. If the targets are showing unhealthy as displayed in the diagram on the right, Navigate to Compute Engine > VM Instances and reset the Spoke 1 VMs.

3. Otherwise, it I possible to login to the VMs using the login information from the deployment and execute the following:

```
sudo apt-get update
sudo apt-get install -y php
sudo apt-get install -y apache2
sudo apt-get install -y php7
sudo apt-get install -y libapache2-mod-php7
sudo wget -O /var/www/html/index.php
https://raw.githubusercontent.com/wwce/terraform/master/gcp/adv_peering_2fw_2spoke/scripts/showheaders.php
sudo systemctl restart apache2
```

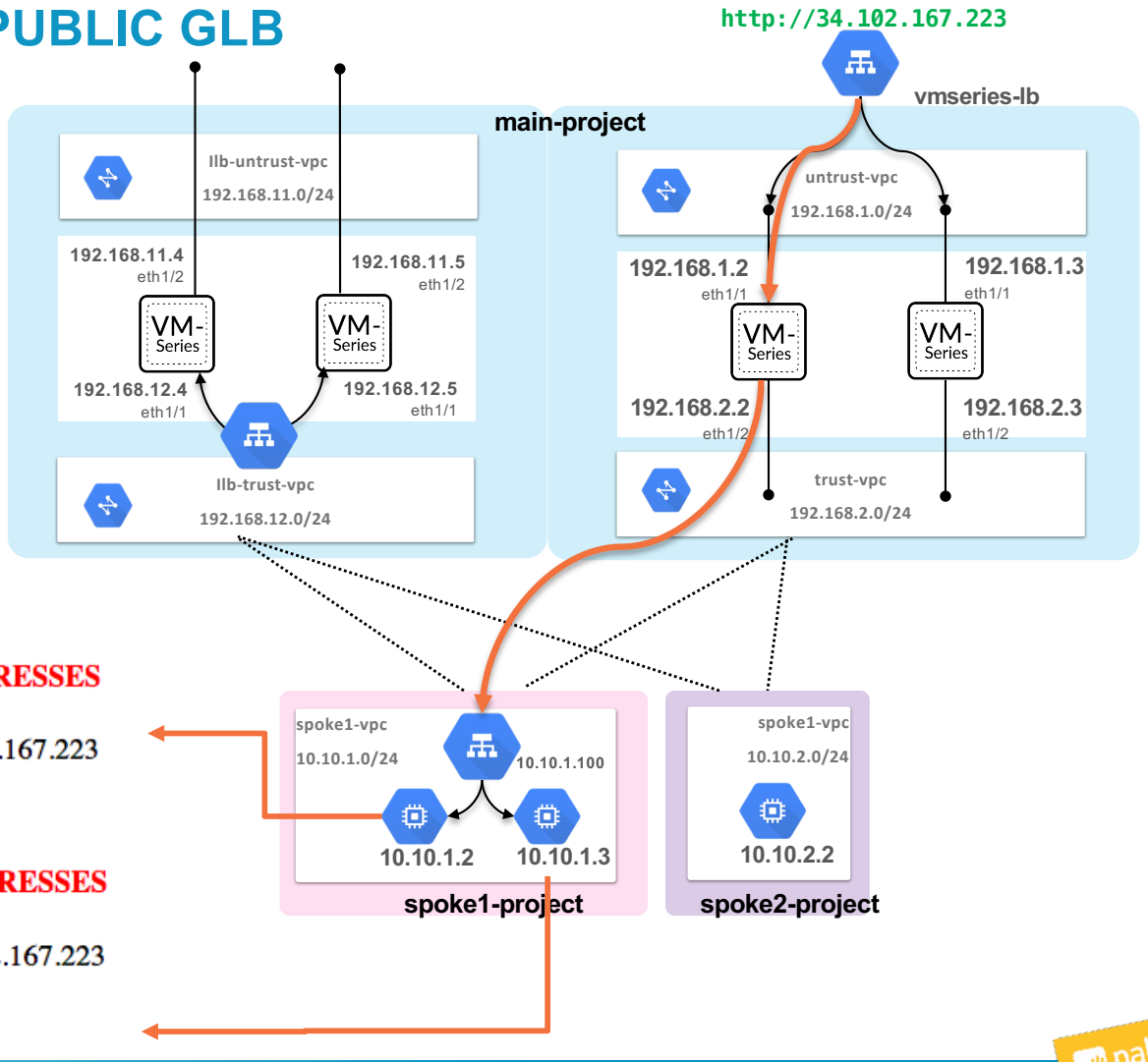
```
Apply complete! Resources: 74 added, 0 changed, 0 destroyed.

Outputs:

                        IMPORTANT!! PLEASE READ!!                                = [
    ================================================================================,
    Before proceeding, you must enable import/export custom routes on all peering links,
    and remove the default (0.0.0.0/0) route from TRUST, SPOKE1, and SPOKE2 VPCs,
    There is also a need to create a default route in the ilb-trust-subnet pointing to,
    the internal load balancer,
    ================================================================================
]
GLB-ADDRESS    = http://34.102.167.223
MGMT-URL-FW1   = https://35.239.219.110
MGMT-URL-FW2   = https://35.226.119.108
MGMT-URL-ILB-FW3  = https://34.68.212.223
MGMT-URL-ILB-FW4   = https://34.69.41.227
SSH-SPOKE1-VM1-FW2 = ssh ubuntu@35.232.77.73 -p 221 -i <INSERT KEY>
SSH-SPOKE1-VM2-FW1 = ssh ubuntu@34.68.152.104 -p 222 -i <INSERT KEY>
SSH-SPOKE2-FW1 = ssh ubuntu@34.68.152.104 -p 223 -i <INSERT KEY>
SSH-SPOKE2-FW2 = ssh ubuntu@35.232.77.73 -p 223 -i <INSERT KEY>
```





TEST INBOUND THROUGH PUBLIC GLB

1. From the Terraform output, copy `GLB-ADDRESS = http://35.244.207.26` into a web browser.
2. Once the page resolves, on each refresh you should receive varying local IPs. This indicates that ingress load balancing is functioning as expected.
3. View the firewall logs on firewall VMs 1 and 2 to view load balancing functionality.



SOURCE & DESTINATION ADDRESSES
INTERVAL: 0.0001981258392334
SOURCE IP: [redacted], 34.102.167.223
LOCAL IP: 10.10.1.3
VM NAME: spoke1-vm2

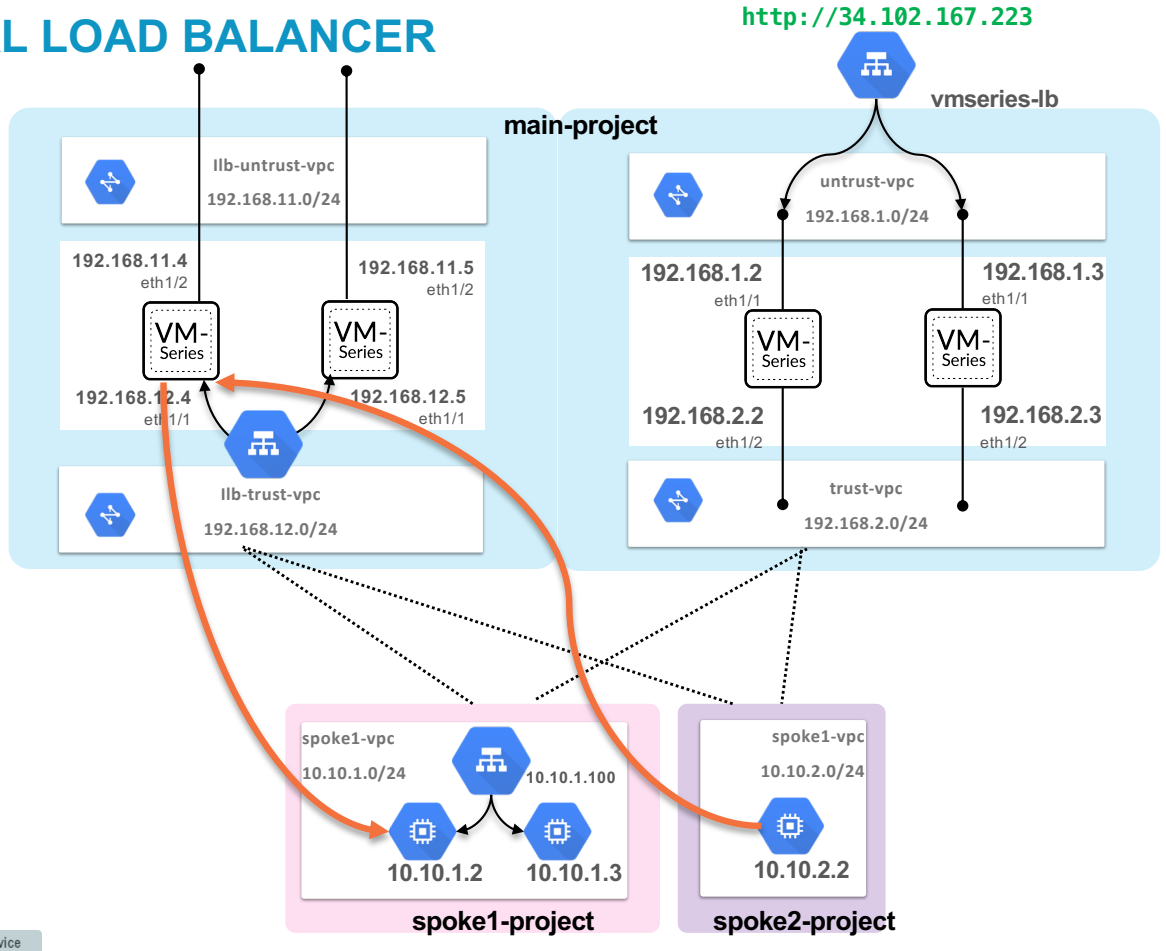
SOURCE & DESTINATION ADDRESSES
INTERVAL: 0.00021600723266602
SOURCE IP: [redacted], 34.102.167.223
LOCAL IP: 10.10.1.2
VM NAME: spoke1-vm1

TEST EAST/WEST THROUGH INTERNAL LOAD BALANCER

1. From the Terraform output, copy `SSH-SPOKE2-FW1 = ssh ubuntu@35.224.112.224 -p 223 -i <INSERT KEY>` command prompt and insert the path to the SSH key that was used in the variables.tf file.

2. Once logged into the VM perform a curl to one of the web servers in spoke1 and check the firewall logs on the firewall VMs 3 and 4.

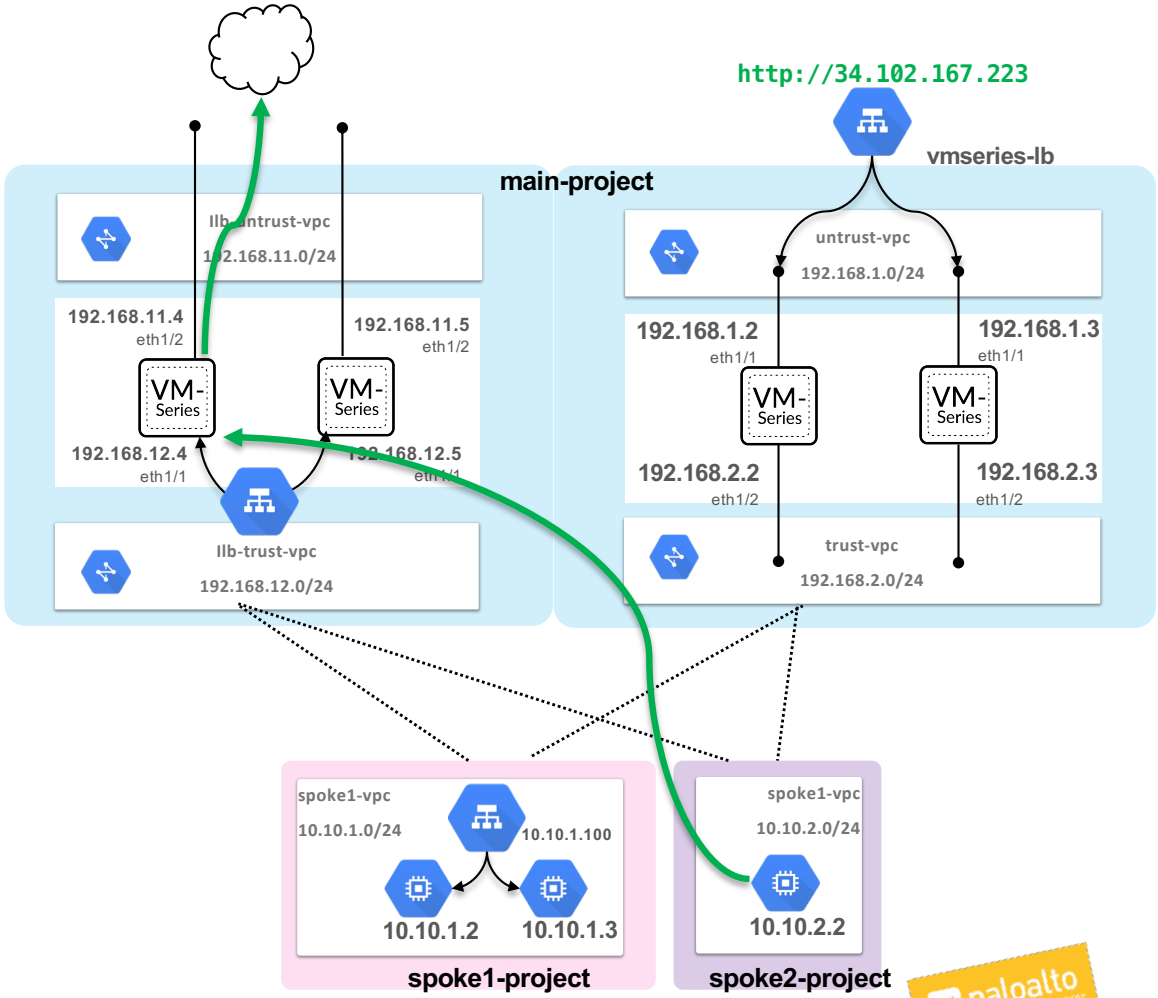
```
SJCMACC0N5JHD4:~ dspears$ ssh ubuntu@35.224.112.224 -p 223 -i ~/.ssh/davejspears
The authenticity of host '[35.224.112.224]:223 ([35.224.112.224]:223)' can't be established.
ECDSA key fingerprint is SHA256:glc9LoNL8EBZ4camESwHZcI0efW9gRvRXPCtJTBn67w.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '[35.224.112.224]:223' (ECDSA) to the list of known hosts.
Welcome to Ubuntu 16.04.6 LTS (GNU/Linux 4.15.0-1042-gcp x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

0 packages can be updated.
0 updates are security updates.



The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@spoke2-vm1:~$ curl 10.10.1.2 | grep Title
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   463  100   463    0     0  69718      0 --:--:-- --:--:-- --:--:-- 77166
ubuntu@spoke2-vm1:~$
```



TEST OUTBOUND THROUGH INTERNAL LOAD BALANCER

1. From the Terraform output, copy `SSH-SPOKE2-FW1 = ssh ubuntu@35.224.112.224 -p 223 -i <INSERT KEY>` command prompt and insert the path to the SSH key that was used in the variables.tf file.

2. Once logged into the VM perform a curl to one of the web servers in spoke1 and check the firewall logs on the firewall VMs 3 and 4.



TEST OUTBOUND

1. Open the command prompt from the sever in spoke1 or get the ssh information from the terraform output and reconnect.

2. Test egress connectivity (i.e. `sudo apt-get update`).

3. View the firewall logs. The egress request should flow through both firewalls since we are leveraging ECMP.

```
Apply complete! Resources: 74 added, 0 changed, 0 destroyed.

Outputs:

                        IMPORTANT!! PLEASE READ!!                                  = [
    ================================================================================,
    Before proceeding, you must enable import/export custom routes on all peering links,
    and remove the default (0.0.0.0/0) route from TRUST, SPOKE1, and SPOKE2 VPCs,
    There is also a need to create a default route in the ilb-trust-subnet pointing to,
    the internal load balancer,
    ================================================================================
]
GLB_ADDRESS     = http://34.102.167.223
MGMT-URL-FW1    = https://35.239.210.110
MGMT-URL-FW2    = https://35.232.77.73
MGMT-URL-ILB-FW3  = https://34.67.53.20
MGMT-URL-ILB-FW4    = https://34.68.219.215
SSH-SPOKE1-VM1-FW1 = ssh ubuntu@35.224.112.224 -p 221 -i <INSERT KEY>
SSH-SPOKE1-VM1-FW2 = ssh ubuntu@34.68.152.104 -p 221 -i <INSERT KEY>
SSH-SPOKE1-VM2-FW1 = ssh ubuntu@35.224.112.224 -p 222 -i <INSERT KEY>
SSH-SPOKE1-VM2-FW2 = ssh ubuntu@34.68.152.104 -p 222 -i <INSERT KEY>
SSH-SPOKE2-FW1   ssh ubuntu@35.224.112.224 -p 223 -i <INSERT KEY>
SSH-SPOKE2-FW2 = ssh ubuntu@34.68.152.104 -p 223 -i <INSERT KEY>
```

FW3 Egress Traffic

Dashboard | ACC | **Monitor** | Policies | Objects | Network | Device

(rule eq 'allow-all-out-1')

	Receive Time	Type	From Zone	To Zone	Source	Destination	To Port	Application	Action	Rule	Session End Reason	Bytes
	10/02 10:48:08	end	trust	untrust	10.10.2.2	35.239.123.202	80	apt-get	allow	allow-all-out-1	tcp-fin	15.7M
	10/02 10:36:56	end	trust	untrust	10.10.2.2	91.189.95.15	80	web-browsing	allow	allow-all-out-1	tcp-fin	5.7k
	10/02 10:26:02	end	trust	untrust	10.10.1.3	151.101.0.133	443	ssl	allow	allow-all-out-1	tcp-rst-from-client	8.6k
	10/02 10:25:53	end	trust	untrust	10.10.1.3	35.184.34.241	80	apt-get	allow	allow-all-out-1	tcp-fin	1.3M
	10/02 10:25:53	end	trust	untrust	10.10.1.2	35.239.123.202	80	apt-get	allow	allow-all-out-1	tcp-fin	1.3M
	10/02 10:25:45	end	trust	untrust	10.10.1.3	35.184.34.241	80	apt-get	allow	allow-all-out-1	tcp-fin	1.6M

FW4 Egress Traffic

Dashboard | ACC | **Monitor** | Policies | Objects | Network | Device

(rule eq 'allow-all-out-1')

	Receive Time	Type	From Zone	To Zone	Source	Source User	Destination	To Port	Application	Action	Rule	Session End Reason	Bytes
	10/02 10:48:08	end	trust	untrust	10.10.2.2		91.189.91.24	80	apt-get	allow	allow-all-out-1	tcp-fin	4.1k
	10/02 10:48:08	end	trust	untrust	10.10.2.2		91.189.92.191	80	apt-get	allow	allow-all-out-1	tcp-fin	19.3k
	10/02 10:48:06	end	trust	untrust	10.10.2.2		91.189.91.24	80	apt-get	allow	allow-all-out-1	tcp-fin	808.6k
	10/02 10:48:06	end	trust	untrust	10.10.2.2		91.189.91.24	80	apt-get	allow	allow-all-out-1	tcp-fin	204.5k
	10/02 10:48:06	end	trust	untrust	10.10.2.2		91.189.91.24	80	apt-get	allow	allow-all-out-1	tcp-fin	7.2k
	10/02 10:48:06	end	trust	untrust	10.10.2.2		91.189.91.24	80	apt-get	allow	allow-all-out-1	tcp-fin	494.2k