Network Principles in Practice: Cloud Networking

Module: Virtual Private Cloud (VPC)

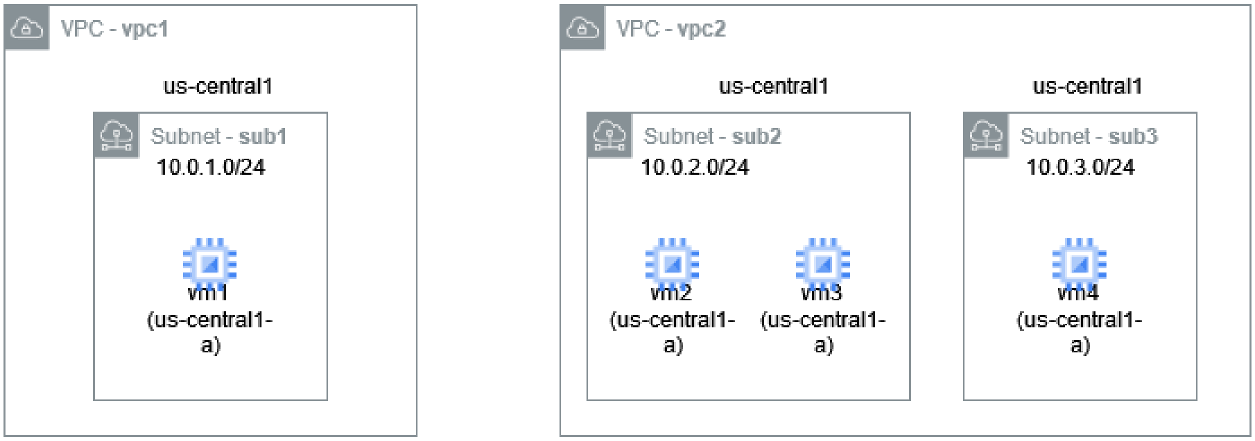
Assignment

Overview

In this lab, you will build the following infrastructure in terraform.

Not shown is that tf-mod2-lab1-vpc1 and tf-mod2-lab1-vpc2 should each have a firewall rule, named tf-mod2-lab1-fwrule1 or tf-mod2-lab1-fwrule2 respective. Allowed traffic will be icmp, and tcp traffic to ports 22 and 1234. Traffic should be allowed from any source (i.e., 0.0.0.0/0)

Note: In the diagram (and in the text description below), we use shortened resource names (vpc1, sub1, vm1, etc.). You should prefix all resource names with "tf-mod2-lab1-". For example, vpc1 would be tf-mod2-lab1-vpc1.



For a higher quality diagram, you can open the file mod2-setup.xml in https://app.diagrams.net/

For metadata for vm1 and vm2, setup the startup-script to have it install netcat-traditional and ncat. You can do so as follows:

```
 metadata = {
   startup-script = "sudo apt update; sudo apt install -y netcat-traditional ncat;"
  }
```

Preparing to Collect Output

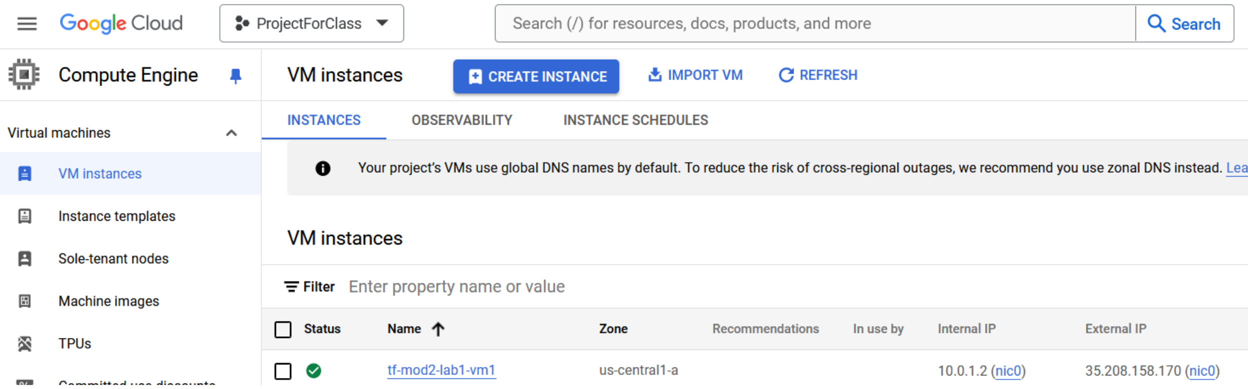
The lab will consist of questions on the coursera site about the lab. This will be done with some steps or queries you will perform, and capturing the output. This is the preparation for that.

Once you have run terraform apply, take the following steps:

1) Make sure jq is installed. https://jqlang.github.io/jq/

2) Run terraform show --json > terraform_show_output.json
(note: that's two dashes before json)

3) Find the internal and external IP addresses of each of the VMs.
   a. You can do that in the Google Cloud console



Your project's VMs use global DNS names by default. To reduce the risk of cross-regional outages, we recommend you use zonal DNS instead. Lea

VM instances

□	Status	Name ↑	Zone	Recommendations	In use by	Internal IP	External IP
□	✓	tf-mod2-lab1-vm1	us-central1-a			10.0.1.2 (nic0)	35.208.158.170 (nic0)

   b. Or from the terraform show output by running these queries with jq:
   For vm1 external (for vm2 – replace vm1 with vm2):

```
jq '.values.root_module.resources[] | select(.address == "google_compute_instance.tf-mod2-lab1-vm1") | .values.network_interface[] | select(.name == "nic0") | .access_config[].nat_ip' terraform_show_out.json
```

   For vm1 internal (for others, just replace vm1 with vm2 or vm3 or vm4):

```
.values.root_module.resources[] | select(.address == "google_compute_instance.tf-mod2-lab1-vm1").values.network_interface[0].network_ip
```

   If in either case, the query is empty (prints nothing), you have an error in the terraform – whether the naming or structure.

Collecting Output

Test 1: Manually check connectivity with ping.

a. On vm1 run ping <external IP address of vm2> (note output)
b. On vm1 run ping <internal IP address of vm2> (note output)
c. On vm2 run ping <internal IP address of vm4> (note output)

Test 2: Manually check connectivity from vm1 to vm2 with netcat (on port 1234). Note output.

a. On vm2 run “/usr/bin/nc -l -p 1234”
b. On vm1 run “/usr/bin/nc -zv -w2 <external IP of vm2> 1234”
(note: <external IP of vm2> means replace that with the external IP address of vm2 in your setup, which you can find in the Google Cloud console.)

Test 3: Manually check connectivity from vm1 to vm2 with netcat (on port 5555).

a. On vm2 run “/usr/bin/nc -l -p 5555”
b. On vm1 run “/usr/bin/nc -zv -w2 <external IP of vm2> 5555”
(note: <external IP of vm2> means replace that with the external IP address of vm2 in your setup, which you can find in the Google Cloud console.)

Queries to Run with jq

Run these jq queries and note the output of each. You’d run jq ‘<query>’ terraform_show_output.json (replacing <query> with one of the strings below.

Query 1:

```
.values.root_module.resources[] | select(.type == "google_compute_firewall" and .name =="tf-mod2-lab1-fwrule1").values.allow[][] | select(. != "").ports[]
```

Query 2:

```
.values.root_module.resources[] | select(.type == "google_compute_network" and .name == "tf-mod2-lab1-vpc1") | .address
```

Query 3:

```
.values.root_module.resources | map(select(.address == "google_compute_instance.tf-mod2-lab1-vm1")) | .name
```

Query 4:

```
.values.root_module.resources[] | select(.type == "google_compute_subnetwork" and .name == "tf-mod2-lab1-sub1") | .values.ip_cidr_range
```

With the output for each of these, go and answer some questions on the coursera site.

In Development

There is a prototype of some python code to help perform checks if you want to try it out. The README will have more information, but it’ll check the structure and print out whether it matches what’s expected, and it’ll preform the netcat checks (by ssh into vm1 and vm2 and running the commands)

https://github.com/eric-keller/coursera-npp-cloud/tree/main/cloud-extract