


Multiple VPC Networks

Activate Cloud Shell

Cloud Shell is a virtual machine that is loaded with development tools. It offers a persistent 5GB home directory and runs on the Google Cloud. Cloud Shell provides command-line access to your Google Cloud resources.

1. Click **Activate Cloud Shell**  at the top of the Google Cloud console.
2. Click through the following windows:
 - Continue through the Cloud Shell information window.
 - Authorize Cloud Shell to use your credentials to make Google Cloud API calls.

When you are connected, you are already authenticated, and the project is set to your **Project_ID**, `PROJECT_ID`. The output contains a line that declares the **Project_ID** for this session:

```
Your Cloud Platform project in this session is set to "PROJECT ID"
gcloud is the command-line tool for Google Cloud. It comes pre-installed on Cloud Shell
and supports tab-completion.
```

3. (Optional) You can list the active account name with this command:
`gcloud auth list`

4. Click **Authorize**.

Output:

```
ACTIVE: *
ACCOUNT: "ACCOUNT"

To set the active account, run:
$ gcloud config set account `ACCOUNT`
```

5. (Optional) You can list the project ID with this command:
`gcloud config list project`

Output:

```
[core]  
project = "PROJECT_ID"
```


Note: For full documentation of `gcloud`, in Google Cloud, refer to [the gcloud CLI overview guide](#).

Task 1. Create custom mode VPC networks with firewall rules

Create two custom networks **managementnet** and **privatenet**, along with firewall rules to allow **SSH**, **ICMP**, and **RDP** ingress traffic.

Create the managementnet network

Create the **managementnet** network using the Cloud console.

1. In the Cloud console, navigate to **Navigation menu** () > **VPC network** > **VPC networks**.
2. Notice the **default** and **mynetwork** networks with their subnets.

Each Google Cloud project starts with the **default** network. In addition, the **mynetwork** network has been premade as part of your network diagram.

3. Click **Create VPC Network**.
4. Set the **Name** to `managementnet`.
5. For **Subnet creation mode**, click **Custom**.
6. Set the following values, leave all other values at their defaults:

Property	Value (type value or select option as specified)
Name	managementsubnet-1
Region	<Region_1>
IPv4 range	10.130.0.0/20

- Click **Done**.
- Click **EQUIVALENT COMMAND LINE**.

These commands illustrate that networks and subnets can be created using the Cloud Shell command line. You will create the **privatenet** network using these commands with similar parameters.

- Click **Close**.
- Click **Create**.

Create the privatenet network

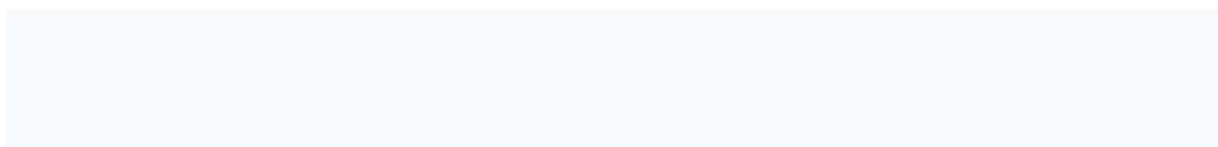
Create the **privatenet** network using the Cloud Shell command line.

- Run the following command to create the **privatenet** network:

```
gcloud compute networks create privatenet --subnet-mode=custom
```
- Run the following command to create the **privatesubnet-1** subnet:

```
gcloud compute networks subnets create privatesubnet-1 --network=privatenet --region=Region_1 --range=172.16.0.0/24
```
- Run the following command to create the **privatesubnet-2** subnet:

```
gcloud compute networks subnets create privatesubnet-2 --network=privatenet --region=Region_2 --range=172.20.0.0/20
```



Create the privatenet network

4. Run the following command to list the available VPC networks:

```
gcloud compute networks list
```

The output should look like this:

```
NAME: default
SUBNET_MODE: AUTO
BGP_ROUTING_MODE: REGIONAL
IPV4_RANGE:
GATEWAY_IPV4:

NAME: managementnet
SUBNET_MODE: CUSTOM
BGP_ROUTING_MODE: REGIONAL
IPV4_RANGE:
GATEWAY_IPV4:

...
```

Note: **default** and **mynetwork** are auto mode networks, whereas, **managementnet** and **privatenet** are custom mode networks. Auto mode networks create subnets in each region automatically, while custom mode networks start with no subnets, giving you full control over subnet creation

5. Run the following command to list the available VPC subnets (sorted by VPC network):

```
gcloud compute networks subnets list --sort-by=NETWORK
```

The output should look like this:

```
NAME: default
REGION: Region_1
NETWORK: default
RANGE: 10.128.0.0/20
STACK_TYPE: IPV4_ONLY
IPV6_ACCESS_TYPE:
INTERNAL_IPV6_PREFIX:
EXTERNAL_IPV6_PREFIX:

...
```

Note: As expected, the **default** and **mynetwork** networks have subnets in [each region \(zones/regions may changes as per lab's requirements\)](#) as they are auto mode networks. The **managementnet** and **privatenet** networks only have the subnets that you created as they are custom mode networks .

6. In the Cloud console, navigate to **Navigation menu > VPC network > VPC networks**.
7. You see that the same networks and subnets are listed in the Cloud console.

Create the firewall rules for managementnet

Create firewall rules to allow **SSH**, **ICMP**, and **RDP** ingress traffic to VM instances on the **managementnet** network.

1. In the Cloud console, navigate to **Navigation menu** (☰) > **VPC network** > **Firewall**.
2. Click + **Create Firewall Rule**.
3. Set the following values, leave all other values at their defaults:

Property	Value (type value or select option as specified)
Name	managementnet-allow-icmp-ssh-rdp
Network	managementnet
Targets	All instances in the network
Source filter	IPv4 Ranges
Source IPv4 ranges	0.0.0.0/0
Protocols and ports	Specified protocols and ports, and then <i>check tcp, type: 22, 3389;</i> and <i>check Other protocols, type: icmp.</i>

Note: Make sure to include the **/0** in the **Source IPv4 ranges** to specify all networks.

4. Click **EQUIVALENT COMMAND LINE**.

These commands illustrate that firewall rules can also be created using the Cloud Shell command line. You will create the **privatenet**'s firewall rules using these commands with similar parameters.

5. Click **Close**.

6. Click **Create**.

Test Completed Task

Click **Check my progress** to verify your performed task. If you have successfully created firewall rules for managementnet network, you will see an assessment score.

Create the firewall rules for managementnet

Check my progress

Create the firewall rules for privatenet

Create the firewall rules for **privatenet** network using the Cloud Shell command line.

1. In Cloud Shell, run the following command to create the **privatenet-allow-icmp-ssh-rdp** firewall rule:

```
gcloud compute firewall-rules create privatenet-allow-icmp-ssh-rdp --
direction=INGRESS --priority=1000 --network=privatenet --action=ALLOW -
-rules=icmp,tcp:22,tcp:3389 --source-ranges=0.0.0.0/0
```

The output should look like this:

```
Creating firewall...done.
NAME: privatenet-allow-icmp-ssh-rdp
NETWORK: privatenet
DIRECTION: INGRESS
PRIORITY: 1000
ALLOW: icmp,tcp:22,tcp:3389
DENY:
DISABLED: False
```

Create the firewall rules for privatenet

Check my progress

2. Run the following command to list all the firewall rules (sorted by VPC network):

```
gcloud compute firewall-rules list --sort-by=NETWORK
```

The output should look like this:

```
NAME: default-allow-icmp
NETWORK: default
DIRECTION: INGRESS
PRIORITY: 65534
ALLOW: icmp
DENY:
DISABLED: False
```

```
NAME: default-allow-internal
NETWORK: default
DIRECTION: INGRESS
PRIORITY: 65534
ALLOW: tcp:0-65535,udp:0-65535,icmp
DENY:
DISABLED: False
...
```

The firewall rules for **mynetwork** network have been created for you. You can define multiple protocols and ports in one firewall rule (**privatenet** and **managementnet**), or spread them across multiple rules (**default** and **mynetwork**).

3. In the Cloud console, navigate to **Navigation menu > VPC network > Firewall**.
4. You see that the same firewall rules are listed in the Cloud console.

Task 2. Create VM instances

Create two VM instances:

- **managementnet-vm-1** in **managementsubnet-1**
- **privatenet-vm-1** in **privatesubnet-1**

Create the managementnet-vm-1 instance

Create the **managementnet-vm-1** instance using the Cloud console.

1. In the Cloud console, navigate to **Navigation menu > Compute Engine > VM instances**.

The **mynet-vm-2** and **mynet-vm-1** has been created for you, as part of your network diagram.

2. Click **Create Instance**.
3. In the **Machine configuration**:

Set the following values, leave all other values at their defaults:

Property	Value (type value or select option as specified)
Name	managementnet-vm-1
Region	US_Region
Zone	US_Zone
Series	E2
Machine Type	e2-micro

4. Click **Networking**.

For **Network interfaces**, click the dropdown to edit. Set the following values, leave all other values at their defaults:

Property	Value (type value or select option as specified)
Network	managementnet
Subnetwork	managementsubnet-1

5. Click **Done**.
6. Click **EQUIVALENT CODE**.

This illustrate that VM instances can also be created using the Cloud Shell command line. You will create the **privatenet-vm-1** instance using these commands with similar parameters.

7. Click **Create**.

Create the privatenet-vm-1 instance

Create the **privatenet-vm-1** instance using the Cloud Shell command line.

1. In Cloud Shell, run the following command to create the **privatenet-vm-1** instance:
`gcloud compute instances create privatenet-vm-1 --zone= --machine-type=e2-micro --subnet=privatesubnet-1`

The output should look like this:

```
Created [https://www.googleapis.com/compute/v1/projects/qwiklabs-gcp-04-972c7275ce91/zones/"/instances/privatenet-vm-1].
NAME: privatenet-vm-1
ZONE:
MACHINE_TYPE: e2-micro
PREEMPTIBLE:
INTERNAL_IP: 172.16.0.2
EXTERNAL_IP: 34.135.195.199
STATUS: RUNNING
```

Create the privatenet-vm-1 instance

Check my progress

2. Run the following command to list all the VM instances (sorted by zone):
`gcloud compute instances list --sort-by=ZONE`

The output should look like this:

```
NAME: mynet-vm-2
ZONE:
MACHINE_TYPE: e2-micro
PREEMPTIBLE:
INTERNAL_IP: 10.164.0.2
EXTERNAL_IP: 34.147.23.235
STATUS: RUNNING

NAME: mynet-vm-1
ZONE:
MACHINE_TYPE: e2-micro
PREEMPTIBLE:
INTERNAL_IP: 10.128.0.2
EXTERNAL_IP: 35.232.221.58
STATUS: RUNNING
...
```

3. In the Cloud console, navigate to **Navigation menu** (☰) > **Compute Engine** > **VM instances**.

4. You see that the VM instances are listed in the Cloud console.
5. Click on **Column display options**, then select **Network**. Click **Ok**.

There are three instances in **Region_1** and one instance in **Region_2**. However, these instances are spread across three VPC networks (**managementnet**, **mynetwork** and **privatenet**), with no instance in the same zone and network as another. In the next section, you explore the effect this has on internal connectivity.

Task 3. Explore the connectivity between VM instances

Explore the connectivity between the VM instances. Specifically, determine the effect of having VM instances in the same zone versus having instances in the same VPC network.

Ping the external IP addresses

Ping the external IP addresses of the VM instances to determine if you can reach the instances from the public internet.

1. In the Cloud console, navigate to **Navigation menu > Compute Engine > VM instances**.
2. Note the external IP addresses for **mynet-vm-2**, **managementnet-vm-1**, and **privatenet-vm-1**.
3. For **mynet-vm-1**, click **SSH** to launch a terminal and connect.
4. To test connectivity to **mynet-vm-2**'s external IP, run the following command, replacing **mynet-vm-2**'s external IP:

```
ping -c 3 'Enter mynet-vm-2 external IP here'
```

This should work!

5. To test connectivity to **managementnet-vm-1**'s external IP, run the following command, replacing **managementnet-vm-1**'s external IP:

```
ping -c 3 'Enter managementnet-vm-1 external IP here'
```

This should work!

6. To test connectivity to **privatenet-vm-1**'s external IP, run the following command, replacing **privatenet-vm-1**'s external IP:

```
ping -c 3 'Enter privatenet-vm-1 external IP here'
```

This should work!

Note: You are able to ping the external IP address of all VM instances, even though they are either in a different zone or VPC network. This confirms public access to those instances is only controlled by the **ICMP** firewall rules that you established earlier.

Ping the internal IP addresses

Ping the internal IP addresses of the VM instances to determine if you can reach the instances from within a VPC network.

1. In the Cloud console, navigate to **Navigation menu > Compute Engine > VM instances**.
2. Note the internal IP addresses for **mynet-vm-2**, **managementnet-vm-1**, and **privatenet-vm-1**.
3. Return to the **SSH** terminal for **mynet-vm-1**.
4. To test connectivity to **mynet-vm-2**'s internal IP, run the following command, replacing **mynet-vm-2**'s internal IP:

```
ping -c 3 'Enter mynet-vm-2 internal IP here'
```

Note: You are able to ping the internal IP address of **mynet-vm-2** because it is on the same VPC network as the source of the ping (**mynet-vm-1**), even though both VM instances are in separate zones, regions and continents!

5. To test connectivity to **managementnet-vm-1**'s internal IP, run the following command, replacing **managementnet-vm-1**'s internal IP:

```
ping -c 3 'Enter managementnet-vm-1 internal IP here'
```

Note: This should not work as indicated by a 100% packet loss!

6. To test connectivity to **privatenet-vm-1**'s internal IP, run the following command, replacing **privatenet-vm-1**'s internal IP:

```
ping -c 3 'Enter privatenet-vm-1 internal IP here'
```

Note: This should not work either as indicated by a 100% packet loss! You are unable to ping the internal IP address of **managementnet-vm-1** and **privatenet-vm-1** because they are in separate VPC networks from the source of the ping (**mynet-vm-1**), even though they are all in the same region **Region_1**.

VPC networks are by default isolated private networking domains. However, no internal IP address communication is allowed between networks, unless you set up mechanisms such as VPC peering or VPN.

Note: For the below task consider **region_1 = Region_1** and **region_2 = Region_2**

Task 4. Create a VM instance with multiple network interfaces

Every instance in a VPC network has a default network interface. You can create additional network interfaces attached to your VMs. Multiple network interfaces enable you to create configurations in which an instance connects directly to several VPC networks (up to 8 interfaces, depending on the instance's type).

Create the VM instance with multiple network interfaces

Create the **vm-appliance** instance with network interfaces in **privatesubnet-1**, **managementsubnet-1** and **mynetwork**. The CIDR ranges of these subnets do not overlap, which is a requirement for creating a VM with multiple network interface controllers (NICs).

1. In the Cloud console, navigate to **Navigation menu > Compute Engine > VM instances**.
2. Click **Create Instance**.
3. In the **Machine configuration**:

Set the following values, leave all other values at their defaults:

Property	Value (type value or select option as specified)
Name	vm-appliance
Region	US_Region
Zone	US_Zone
Series	E2
Machine Type	e2-standard-4

Note: The number of interfaces allowed in an instance is dependent on the instance's machine type and the number of vCPUs. The e2-standard-4 allows up to 4 network interfaces. Refer to [the Maximum number of network interfaces section of the Google Cloud Guide](#) for more information.

4. Click **Networking**.

For **Network interfaces**, click the dropdown to edit. Set the following values, leave all other values at their defaults:

Property	Value (type value or select option as specified)
Network	privatenet
Subnetwork	privatesubnet-1

Click **Done**.

Click **Add a network interface**.

Set the following values, leave all other values at their defaults:

Property	Value (type value or select option as specified)
Network	managementnet
Subnetwork	managementsubnet-1

Click **Done**.

Click **Add a network interface**.

Set the following values, leave all other values at their defaults:


Property	Value (type value or select option as specified)
Network	mynetwork
Subnetwork	mynetwork

5. Click **Done**.

6. Click **Create**.

Explore the network interface details

Explore the network interface details of **vm-appliance** within the Cloud console and within the VM's terminal.

1. In the Cloud console, navigate to **Navigation menu** () > **Compute Engine** > **VM instances**.
2. Click **nic0** within the **Internal IP** address of **vm-appliance** to open the **Network interface details** page.
3. Verify that **nic0** is attached to **privatesubnet-1**, is assigned an internal IP address within that subnet (172.16.0.0/24), and has applicable firewall rules.
4. Click **nic0** and select **nic1**.
5. Verify that **nic1** is attached to **managementsubnet-1**, is assigned an internal IP address within that subnet (10.130.0.0/20), and has applicable firewall rules.

6. Click **nic1** and select **nic2**.
7. Verify that **nic2** is attached to **mynetwork**, is assigned an internal IP address within that subnet (10.128.0.0/20), and has applicable firewall rules.

Note: Each network interface has its own internal IP address so that the VM instance can communicate with those networks.

8. In the Cloud console, navigate to **Navigation menu > Compute Engine > VM instances**.
9. For **vm-appliance**, click **SSH** to launch a terminal and connect.
10. Run the following, to list the network interfaces within the VM instance:
`sudo ifconfig`

The output should look like this:

```
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1460
    inet 172.16.0.3 netmask 255.255.255.255 broadcast 172.16.0.3
    inet6 fe80::4001:acff:fe10:3 prefixlen 64 scopeid 0x20<link>
    ether 42:01:ac:10:00:03 txqueuelen 1000 (Ethernet)
    RX packets 626 bytes 171556 (167.5 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 568 bytes 62294 (60.8 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1460
    inet 10.130.0.3 netmask 255.255.255.255 broadcast 10.130.0.3
    inet6 fe80::4001:aff:fe82:3 prefixlen 64 scopeid 0x20<link>
    ether 42:01:0a:82:00:03 txqueuelen 1000 (Ethernet)
    RX packets 7 bytes 1222 (1.1 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 17 bytes 1842 (1.7 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
eth2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1460
    inet 10.128.0.3 netmask 255.255.255.255 broadcast 10.128.0.3
    inet6 fe80::4001:aff:fe80:3 prefixlen 64 scopeid 0x20<link>
    ether 42:01:0a:80:00:03 txqueuelen 1000 (Ethernet)
    RX packets 17 bytes 2014 (1.9 KiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 17 bytes 1862 (1.8 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Note: The **sudo ifconfig** command lists a Linux VM's network interfaces along with the internal IP addresses for each interface.

Explore the network interface connectivity

Demonstrate that the **vm-appliance** instance is connected to **privatesubnet-1**, **managementsubnet-1** and **mynetwork** by pinging VM instances on those subnets.

1. In the Cloud console, navigate to **Navigation menu > Compute Engine > VM instances**.
2. Note the internal IP addresses for **privatenet-vm-1**, **managementnet-vm-1**, **myinet-vm-1**, and **myinet-vm-2**.
3. Return to the **SSH** terminal for **vm-appliance**.

4. To test connectivity to **privatenet-vm-1**'s internal IP, run the following command, replacing **privatenet-vm-1**'s internal IP:

```
ping -c 3 'Enter privatenet-vm-1's internal IP here'
```

This works!

5. Repeat the same test by running the following:

```
ping -c 3 privatenet-vm-1
```

Note: You are able to ping **privatenet-vm-1** by its name because VPC networks have an internal DNS service that allows you to address instances by their DNS names rather than their internal IP addresses. When an internal DNS query is made with the instance hostname, it resolves to the primary interface (nic0) of the instance. Therefore, this only works for **privatenet-vm-1** in this case.

6. To test connectivity to **managementnet-vm-1**'s internal IP, run the following command, replacing **managementnet-vm-1**'s internal IP:

```
ping -c 3 'Enter managementnet-vm-1's internal IP here'
```

This works!

7. To test connectivity to **mynet-vm-1**'s internal IP, run the following command, replacing **mynet-vm-1**'s internal IP:

```
ping -c 3 'Enter mynet-vm-1's internal IP here'
```

This works!

8. To test connectivity to **mynet-vm-2**'s internal IP, run the following command, replacing **mynet-vm-2**'s internal IP:

```
ping -c 3 'Enter mynet-vm-2's internal IP here'
```

Note: This does not work! In a multiple interface instance, every interface gets a route for the subnet that it is in. In addition, the instance gets a single default route that is associated with the primary interface eth0. Unless manually configured otherwise, any traffic leaving an instance for any destination other than a directly connected subnet will leave the instance via the default route on eth0.

9. To list the routes for **vm-appliance** instance, run the following command:

```
ip route
```

The output should look like this:

```
default via 172.16.0.1 dev eth0
10.128.0.0/20 via 10.128.0.1 dev eth2
10.128.0.1 dev eth2 scope link
10.130.0.0/20 via 10.130.0.1 dev eth1
```



```
10.130.0.1 dev eth1 scope link
172.16.0.0/24 via 172.16.0.1 dev eth0
172.16.0.1 dev eth0 scope link
```

Note: The primary interface eth0 gets the default route (default via 172.16.0.1 dev eth0), and all three interfaces eth0, eth1 and eth2 get routes for their respective subnets. Since, the subnet of **mynet-vm-2 (10.132.0.0/20)** is not included in this routing table, the ping to that instance leaves **vm-appliance** on eth0 (which is on a different VPC network). You could change this behavior by configuring policy routing as documented in the [Configuring policy routing section of the Google Cloud Guide](#).