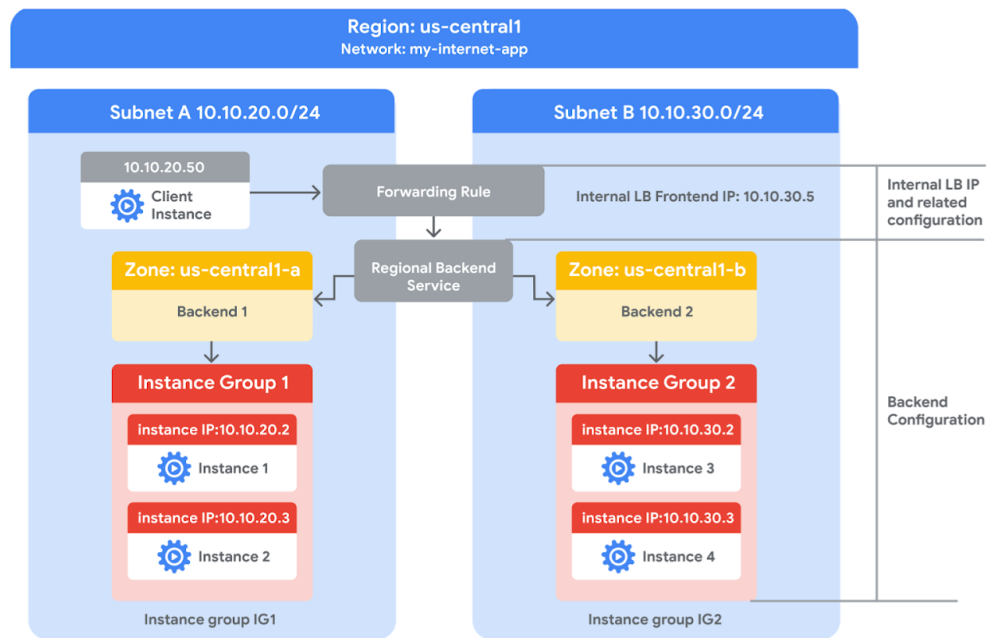


Configuring an Internal Load Balancer

Overview

Google Cloud offers Internal Load Balancing for your TCP/UDP-based traffic. Internal Load Balancing enables you to run and scale your services behind a private load balancing IP address that is accessible only to your internal virtual machine instances.

In this lab, you create two managed instance groups in the same region. Then you configure and test an internal load balancer with the instances groups as the backends, as shown in this network diagram:



Objectives

In this lab, you will learn how to perform the following tasks:

- Create internal traffic and health check firewall rules.
- Create a NAT configuration using Cloud Router.
- Configure two instance templates.
- Create two managed instance groups.
- Configure and test an internal load balancer.

Task 1. Configure internal traffic and health check firewall rules

Configure firewall rules to allow internal traffic connectivity from sources in the 10.10.0.0/16 range. This rule allows incoming traffic from any client located in the subnet.

Health checks determine which instances of a load balancer can receive new connections. For HTTP load balancing, the health check probes to your load-balanced instances come from addresses in the ranges **130.211.0.0/22** and **35.191.0.0/16**. Your firewall rules must allow these connections.

Explore the my-internal-app network

The network **my-internal-app** with **subnet-a** and **subnet-b** and firewall rules for **RDP**, **SSH**, and **ICMP** traffic have been configured for you.


- In the Cloud Console, on the **Navigation menu** (≡), click **VPC network > VPC networks**.

Notice the **my-internal-app** network with its subnets: **subnet-a** and **subnet-b**. Each Google Cloud project starts with the **default** network. In addition, the **my-internal-app** network has been created for you as part of your network diagram.

You will create the managed instance groups in **subnet-a** and **subnet-b**. Both subnets are in the **us-central1** region because an internal load balancer is a regional service. The managed instance groups will be in different zones, making your service immune to zonal failures.

Create the firewall rule to allow traffic from any sources in the 10.10.0.0/16 range

Create a firewall rule to allow traffic in the 10.10.0.0/16 subnet.

1. On the **Navigation menu** () , click **VPC network > Firewall**.
Notice the **app-allow-icmp** and **app-allow-ssh-rdp** firewall rules.
These firewall rules have been created for you.
2. Click **Create Firewall Rule**.
3. Specify the following, and leave the remaining settings as their defaults.

Property	Value (type value or select option as specified)
Name	fw-allow-lb-access
Network	my-internal-app
Targets	Specified target tags
Target tags	backend-service


Source filter	IPv4 ranges
Source IPv4 ranges	10.10.0.0/16
Protocols and ports	Allow all

Note: Make sure to include the **/16** in the **Source IPv4 ranges**.

4. Click **Create**.

Create the health check rule

Create a firewall rule to allow health checks.

1. On the **Navigation menu** () , click **VPC network > Firewall**.
2. Click **Create Firewall Rule**.
3. Specify the following, and leave the remaining settings as their defaults.

Property	Value (type value or select option as specified)
Name	fw-allow-health-checks

Network	my-internal-app
Targets	Specified target tags
Target tags	backend-service
Source filter	IPv4 Ranges
Source IPv4 ranges	130.211.0.0/22 and 35.191.0.0/16
Protocols and ports	Specified protocols and ports

Note: Make sure to include the **/22** and **/16** in the **Source IPv4 ranges**.

4. For **tcp**, check the checkbox and specify port **80**.
5. Click **Create**.

Task 2. Create a NAT configuration using Cloud Router

The Google Cloud VM backend instances that you setup in Task 3 will not be configured with external IP addresses.

Instead, you will setup the Cloud NAT service to allow these VM instances to send outbound traffic only through the Cloud NAT, and receive inbound traffic through the load balancer.

Create the Cloud Router instance

1. In the Cloud Console, on the **Navigation menu** (≡), click **Network services > Cloud NAT**.
2. Click **Get started**.
3. Specify the following, and leave the remaining settings as their defaults.

Property	Value (type value or select option as specified)
Gateway name	nat-config
Network	my-internal-app

Region	us-central1 (Iowa)
--------	--------------------

4.

Click **Cloud Router**, and select **Create new router**.

5. For **Name**, type `nat-router-us-central1`.

6. Click **Create**.

7. In Create Cloud NAT gateway, click **Create**.

Note: Wait until the NAT Gateway Status changes to Running before moving onto the next task.

Task 3. Configure instance templates and create instance groups

A managed instance group uses an instance template to create a group of identical instances. Use these to create the backends of the internal load balancer.

This task has been performed for you at the start of this lab. You will need to SSH into each instance group VM and run the following command to setup the environment.

1. On the **Navigation menu**, click **Compute Engine > VM instances**.

Notice two instances that start with **instance-group-1** and **instance-group-2**.

2. Select the SSH button next to **instance-group-1** to SSH into the VM.

3. Run the following command to re-run the instance's startup script:

```
sudo google_metadata_script_runner startup
```

4. Repeat the previous steps for **instance-group-2**.
5. Wait for both startup scripts to finish executing, then close the SSH terminal to each VM. The output of the startup script should state the following:

```
Finished running startup scripts.
```

Verify the backends

Verify that VM instances are being created in both subnets and create a utility VM to access the backends' HTTP sites.

1. On the **Navigation menu**, click **Compute Engine > VM instances**.

Notice two instances that start with **instance-group-1** and **instance-group-2**.

These instances are in separate zones, and their internal IP addresses are part of the **subnet-a** and **subnet-b** CIDR blocks.

2. Click **Create Instance**.

3. Specify the following, and leave the remaining settings as their defaults.

Property	Value (type value or select option as specified)
Name	utility-vm

Region	us-central1
Zone	us-central1-f
Series	N1
Machine type	n1-standard-1 (1 vCPU, 3.75 GB memory)
Boot disk	Debian GNU/Linux 10 (buster)

4.

In **Advanced options**, click **Networking, disks, security, management, sole tenancy**.

5. Click **Networking**.

6. For **Network interfaces**, click the dropdown to edit.

7. Specify the following, and leave the remaining settings as their defaults.

Property	Value (type value or select option as specified)
Network	my-internal-app

Subnetwork	subnet-a
Primary internal IP	Ephemeral (Custom)
Custom ephemeral IP address	10.10.20.50
External IPv4 address	None

8.

Click **Done**.

9. Click **Create**.

10. Note that the internal IP addresses for the backends are **10.10.20.2** and **10.10.30.2**.

Note: If these IP addresses are different, replace them in the two **curl** commands below.

11. For **utility-vm**, click **SSH** to launch a terminal and connect. If you see the *Connection via Cloud Identity-Aware Proxy Failed* popup, click **Retry**.

12. To verify the welcome page for **instance-group-1-xxxx**, run the following command:

```
curl 10.10.20.2
```

The output should look like this.

Output:

```
<h1>Internal Load Balancing Lab</h1><h2>Client IP</h2>Your IP address :  
10.10.20.50<h2>Hostname</h2>Server Hostname :  
instance-group-1-1zn8<h2>Server Location</h2>Region and Zone :  
us-central1-a
```

13. To verify the welcome page for **instance-group-2-xxxx**, run the following command:

```
curl 10.10.30.2
```

The output should look like this.

Output:

```
<h1>Internal Load Balancing Lab</h1><h2>Client IP</h2>Your IP address :  
10.10.20.50<h2>Hostname</h2>Server Hostname :  
instance-group-2-q5wp<h2>Server Location</h2>Region and Zone :  
us-central1-b
```

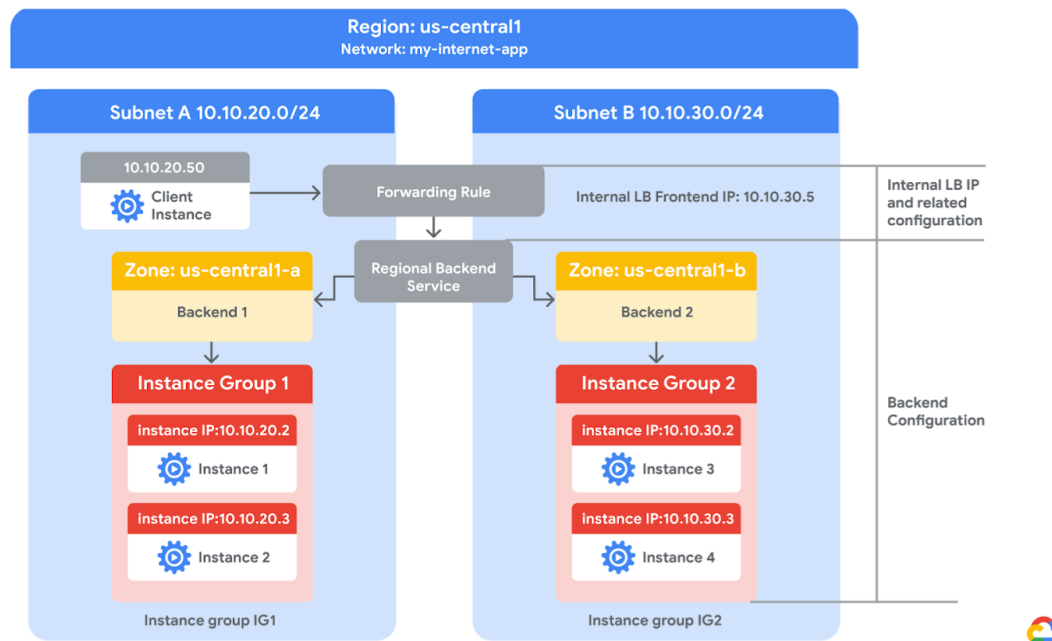
Note: This will be useful when verifying that the internal load balancer sends traffic to both backends.

14. Close the SSH terminal to **utility-vm**:

```
exit
```

Task 4. Configure the internal load balancer

Configure the internal load balancer to balance traffic between the two backends (**instance-group-1** in us-central1-a and **instance-group-2** in us-central1-b), as illustrated in the network diagram.



Start the configuration

1. In the Cloud Console, on the **Navigation menu** (**≡**), click **Network Services > Load balancing**.
2. Click **Create load balancer**.
3. Under **TCP Load Balancing**, click **Start configuration**.
4. For **Internet facing or internal only**, select **Only between my VMs**.

Note: Choosing **Only between my VMs** makes this load balancer internal. This choice requires the backends to be in a single region (us-central1) and does not allow offloading TCP processing to the load balancer.

5. Click **Continue**.
6. For **Name**, type `my-ilb`.
7. For **Region**, type `us-central1`.
8. For **Network**, type `my-internal-app`.

Configure the regional backend service

The backend service monitors instance groups and prevents them from exceeding configured usage.

1. Click **Backend configuration**.
2. Specify the following, and leave the remaining settings as their defaults.

Property	Value (select option as specified)
Instance group	instance-group-1 (us-central1-a)

3.
Click **Done**.
4. Click **Add backend**.
5. For **Instance group**, select **instance-group-2 (us-central1-b)**.
6. Click **Done**.
7. For **Health Check**, select **Create a health check**.
8. Specify the following, and leave the remaining settings as their defaults.

Property	Value (select option as specified)
Name	my-ilb-health-check
Protocol	TCP
Port	80
Check interval	10 sec
Timeout	5 sec
Healthy threshold	2
Unhealthy threshold	3

Note: Health checks determine which instances can receive new connections. This HTTP health check polls instances every 10 seconds, waits up to 5 seconds for a response, and treats 2 successful or 3 failed attempts as healthy threshold or unhealthy threshold, respectively.

9. Click **Save**.

10. Verify that there is a blue checkmark next to **Backend configuration** in the Cloud Console. If there isn't, double-check that you have completed all the steps above.

Configure the frontend

The frontend forwards traffic to the backend.

1. Click **Frontend configuration**.
2. Specify the following, and leave the remaining settings as their defaults.

Property	Value (type value or select option as specified)
Subnetwork	subnet-b
Internal IP > IP address	Create IP address

3. Specify the following, and leave the remaining settings as their defaults.

Property	Value (type value or select option as specified)
Name	my-ilb-ip

Static IP address	Let me choose
Custom IP address	10.10.30.5

4.
Click **Reserve**.
5. Under **Ports**, for **Port number**, type **80**.
6. Click **Done**.

Review and create the internal load balancer

1. Click **Review and finalize**.
2. Review the **Backend** and **Frontend**.
3. Click **Create**.

Wait for the load balancer to be created before moving to the next task.

Task 5. Test the internal load balancer

Verify that the **my-ilb** IP address forwards traffic to **instance-group-1** in us-central1-a and **instance-group-2** in us-central1-b.

Access the internal load balancer

1. On the **Navigation** menu, click **Compute Engine > VM instances**.
2. For **utility-vm**, click **SSH** to launch a terminal and connect.
3. To verify that the internal load balancer forwards traffic, run the following command:

```
curl 10.10.30.5
```

The output should look like this.

Output:

```
<h1>Internal Load Balancing Lab</h1><h2>Client IP</h2>Your IP address :
10.10.20.50<h2>Hostname</h2>Server Hostname:
instance-group-2-1zn8<h2>Server Location</h2>Region and Zone:
us-central1-b
```

Note: As expected, traffic is forwarded from the internal load balancer (10.10.30.5) to the backend.

4. Run the same command a couple of times:

[illegible]

You should be able to see responses from **instance-group-1** in us-central1-a and **instance-group-2** in us-central1-b. If not, run the command again.