**Overview**

In this hands-on lab, you'll learn the differences between a network load balancer and a HTTP load balancer, and how to set them up for your applications running on Compute Engine virtual machines.

There are several ways you can [load balance in Google Cloud](https://cloud.google.com/load-balancing/docs/load-balancing-overview#a_closer_look_at_cloud_load_balancers). This lab takes you through the setup of the following load balancers.:

* L3 [Network Load Balancer](https://cloud.google.com/compute/docs/load-balancing/network/)
* L7 [HTTP(s) Load Balancer](https://cloud.google.com/compute/docs/load-balancing/http/)

Students are encouraged to type the commands themselves, which helps in learning the core concepts. Many labs include a code block that contains the required commands. You can easily copy and paste the commands from the code block into the appropriate places during the lab.

**What you'll do**

* Setup a network load balancer.
* Setup a HTTP(s) load balancer.
* Get hands-on experience learning the differences between network load balancers and HTTP load balancers.

Prerequisites

Familiarity with standard Linux text editors such as vim, emacs, or nano is helpful.

**Setup**

**Before you click the Start Lab button**

Read these instructions. Labs are timed and you cannot pause them. The timer, which starts when you click **Start Lab**, shows how long Google Cloud resources will be made available to you.

This Qwiklabs hands-on lab lets you do the lab activities yourself in a real cloud environment, not in a simulation or demo environment. It does so by giving you new, temporary credentials that you use to sign in and access Google Cloud for the duration of the lab.

**What you need**

To complete this lab, you need:

* Access to a standard internet browser (Chrome browser recommended).
* Time to complete the lab.

**Note:** If you already have your own personal Google Cloud account or project, do not use it for this lab.

**Note:** If you are using a Pixelbook, open an Incognito window to run this lab.

**How to start your lab and sign in to the Google Cloud Console**

1. Click the **Start Lab** button. If you need to pay for the lab, a pop-up opens for you to select your payment method. On the left is a panel populated with the temporary credentials that you must use for this lab.



1. Copy the username, and then click **Open Google Console**. The lab spins up resources, and then opens another tab that shows the **Sign in** page.



***Tip:*** Open the tabs in separate windows, side-by-side.

If you see the **Choose an account** page, click **Use Another Account**. 

1. In the **Sign in** page, paste the username that you copied from the Connection Details panel. Then copy and paste the password.

***Important:*** You must use the credentials from the Connection Details panel. Do not use your Qwiklabs credentials. If you have your own Google Cloud account, do not use it for this lab (avoids incurring charges).

1. Click through the subsequent pages:
   * Accept the terms and conditions.
   * Do not add recovery options or two-factor authentication (because this is a temporary account).
   * Do not sign up for free trials.

After a few moments, the Cloud Console opens in this tab.

**Note:** You can view the menu with a list of Google Cloud Products and Services by clicking the **Navigation menu** at the top-left. 

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.

In the Cloud Console, in the top right toolbar, click the **Activate Cloud Shell** button.



Click **Continue**.



It takes a few moments to provision and connect to the environment. When you are connected, you are already authenticated, and the project is set to your *PROJECT\_ID*. For example:



gcloud is the command-line tool for Google Cloud. It comes pre-installed on Cloud Shell and supports tab-completion.

You can list the active account name with this command:

gcloud auth list

(Output)

Credentialed accounts:

- <myaccount>@<mydomain>.com (active)

(Example output)

Credentialed accounts:

- google1623327\_student@qwiklabs.net

You can list the project ID with this command:

gcloud config list project

(Output)

[core]

project = <project\_ID>

(Example output)

[core]

project = qwiklabs-gcp-44776a13dea667a6

For full documentation of gcloud see the [gcloud command-line tool overview](https://cloud.google.com/sdk/gcloud" \t "_blank).

**Set the default region and zone for all resources**

In Cloud Shell, set the default zone:

gcloud config set compute/zone us-central1-a

Set the default region:

gcloud config set compute/region us-central1

Learn more about choosing zones and regions here: [Regions & Zones documentation](https://cloud.google.com/compute/docs/zones).

**Note:**When you run gcloud on your own machine, the config settings persist across sessions. In Cloud Shell you need to set this for every new session or reconnection.

**Create multiple web server instances**

To simulate serving from a cluster of machines, create a simple cluster of Nginx web servers to serve static content using [Instance Templates](https://cloud.google.com/compute/docs/instance-templates) and [Managed Instance Groups](https://cloud.google.com/compute/docs/instance-groups/). Instance Templates define the look of every virtual machine in the cluster (disk, CPUs, memory, etc). Managed Instance Groups instantiate a number of virtual machine instances using the Instance Template.

To create the Nginx web server clusters, create the following:

* A startup script to be used by every virtual machine instance to setup Nginx server upon startup
* An instance template to use the startup script
* A target pool
* A managed instance group using the instance template

Still in Cloud Shell, create a startup script to be used by every virtual machine instance. This script sets up the Nginx server upon startup:

cat << EOF > startup.sh

#! /bin/bash

apt-get update

apt-get install -y nginx

service nginx start

sed -i -- 's/nginx/Google Cloud Platform - '"\$HOSTNAME"'/' /var/www/html/index.nginx-debian.html

EOF

Create an instance template, which uses the startup script:

gcloud compute instance-templates create nginx-template \

--metadata-from-file startup-script=startup.sh

(Output)

Created [...].

NAME MACHINE\_TYPE PREEMPTIBLE CREATION\_TIMESTAMP

nginx-template n1-standard-1 2015-11-09T08:44:59.007-08:00

Create a target pool. A target pool allows a single access point to all the instances in a group and is necessary for load balancing in the future steps.

gcloud compute target-pools create nginx-pool

(Output)

Created [...].

NAME REGION SESSION\_AFFINITY BACKUP HEALTH\_CHECKS

nginx-pool us-central1

Create a managed instance group using the instance template:

gcloud compute instance-groups managed create nginx-group \

--base-instance-name nginx \

--size 2 \

--template nginx-template \

--target-pool nginx-pool

(Output)

Created [...].

NAME LOCATION SCOPE BASE\_INSTANCE\_NAME SIZE TARGET\_SIZE INSTANCE\_TEMPLATE AUTOSCALED

nginx-group us-central1-a zone nginx 0 2 nginx-template no

This creates 2 virtual machine instances with names that are prefixed with nginx-. This may take a couple of minutes.

List the compute engine instances and you should see all of the instances created:

gcloud compute instances list

(Output)

NAME ZONE MACHINE\_TYPE PREEMPTIBLE INTERNAL\_IP EXTERNAL\_IP STATUS

nginx-7wvi us-central1-a n1-standard-1 10.240.X.X X.X.X.X RUNNING

nginx-9mwd us-central1-a n1-standard-1 10.240.X.X X.X.X.X RUNNING

Now configure a firewall so that you can connect to the machines on port 80 via the EXTERNAL\_IP addresses:

gcloud compute firewall-rules create www-firewall --allow tcp:80

You should be able to connect to each of the instances via their external IP addresses via http://EXTERNAL\_IP/ shown as the result of running the previous command.

Check your lab progress. Click **Check my progress** below to verify that you've created a group of webservers.

Create a group of webservers

Check my progress

**Create a Network Load Balancer**

Network load balancing allows you to balance the load of your systems based on incoming IP protocol data, such as address, port, and protocol type. You also get some options that are not available, with HTTP(S) load balancing. For example, you can load balance additional TCP/UDP-based protocols such as SMTP traffic. And if your application is interested in TCP-connection-related characteristics, network load balancing allows your app to inspect the packets, where HTTP(S) load balancing does not.

For more information, see [Setting Up Network Load Balancing](https://cloud.google.com/compute/docs/load-balancing/network/).

Create an L3 network load balancer targeting your instance group:

gcloud compute forwarding-rules create nginx-lb \

--region us-central1 \

--ports=80 \

--target-pool nginx-pool

(Output)

Created [https://www.googleapis.com/compute/v1/projects/...].

List all Compute Engine forwarding rules in your project.

gcloud compute forwarding-rules list

(Output)

NAME REGION IP\_ADDRESS IP\_PROTOCOL TARGET

nginx-lb us-central1 X.X.X.X TCP us-central1/targetPools/nginx-pool

You can then visit the load balancer from the browser http://IP\_ADDRESS/ where IP\_ADDRESS is the address shown as the result of running the previous command.

Check your lab progress. Click **Check my progress** below to verify that you've created an L3 Network Load Balancer that points to the webservers.

Create an L3 Network Load Balancer that points to the webservers

Check my progress

**Create a HTTP(s) Load Balancer**

HTTP(S) load balancing provides global load balancing for HTTP(S) requests destined for your instances. You can configure URL rules that route some URLs to one set of instances and route other URLs to other instances. Requests are always routed to the instance group that is closest to the user, provided that group has enough capacity and is appropriate for the request. If the closest group does not have enough capacity, the request is sent to the closest group that does have capacity.

Learn more about the [HTTP(s) Load Balancer in the documentation](https://cloud.google.com/compute/docs/load-balancing/http/).

First, create a [health check](https://cloud.google.com/compute/docs/load-balancing/health-checks). Health checks verify that the instance is responding to HTTP or HTTPS traffic:

gcloud compute http-health-checks create http-basic-check

(Output)

Created [https://www.googleapis.com/compute/v1/projects/...].

NAME HOST PORT REQUEST\_PATH

http-basic-check 80 /

Define an HTTP service and map a port name to the relevant port for the instance group. Now the load balancing service can forward traffic to the named port:

gcloud compute instance-groups managed \

set-named-ports nginx-group \

--named-ports http:80

(Output)

Updated [https://www.googleapis.com/compute/v1/projects/...].

Create a [backend service](https://cloud.google.com/compute/docs/reference/latest/backendServices):

gcloud compute backend-services create nginx-backend \

--protocol HTTP --http-health-checks http-basic-check --global

(Output)

Created [https://www.googleapis.com/compute/v1/projects/...].

NAME BACKENDS PROTOCOL

nginx-backend HTTP

Add the instance group into the backend service:

gcloud compute backend-services add-backend nginx-backend \

--instance-group nginx-group \

--instance-group-zone us-central1-a \

--global

(Output)

Updated [https://www.googleapis.com/compute/v1/projects/...].

Create a default URL map that directs all incoming requests to all your instances:

gcloud compute url-maps create web-map \

--default-service nginx-backend

(Output)

Created [https://www.googleapis.com/compute/v1/projects/...].

NAME DEFAULT\_SERVICE

Web-map nginx-backend

To direct traffic to different instances based on the URL being requested, see [content-based routing](https://cloud.google.com/compute/docs/load-balancing/http/content-based-example).

Create a target HTTP proxy to route requests to your URL map:

gcloud compute target-http-proxies create http-lb-proxy \

--url-map web-map

(Output)

Created [https://www.googleapis.com/compute/v1/projects/...].

NAME URL\_MAP

http-lb-proxy web-map

Create a global forwarding rule to handle and route incoming requests. A forwarding rule sends traffic to a specific target HTTP or HTTPS proxy depending on the IP address, IP protocol, and port specified. The global forwarding rule does not support multiple ports.

gcloud compute forwarding-rules create http-content-rule \

--global \

--target-http-proxy http-lb-proxy \

--ports 80

(Output)

Created [https://www.googleapis.com/compute/v1/projects/...].

After creating the global forwarding rule, it can take several minutes for your configuration to propagate.

gcloud compute forwarding-rules list

(Output)

NAME REGION IP\_ADDRESS IP\_PROTOCOL TARGET

http-content-rule X.X.X.X TCP http-lb-proxy

nginx-lb us-central1 X.X.X.X TCP us-central1/....

Take note of the http-content-rule IP\_ADDRESS for the forwarding rule.

From the browser, you should be able to connect to http://IP\_ADDRESS/. It may take three to five minutes. If you do not connect, wait a minute then reload the browser.

Check your lab progress. Click **Check my progress** below to verify that you've created an L7 HTTP(S) Load Balancer.

Create an L7 HTTP(S) Load Balancer

Check my progress

**Test your knowledge**

Test your knowledge about Google cloud Platform by taking our quiz. (Please select multiple correct options if necessary.)

Network Load Balancing is a regional, non-proxied load balancer.

checkTrue



False

Submit

**Congratulations!**

You built a network load balancer and a HTTP(s) load balancer. You practiced with Instance templates and Managed Instance Groups. You are well on your way to having your Google Cloud project monitored with Cloud Monitoring.

Finish Your Quest



This self-paced lab is part of the Qwiklabs [Google Cloud Essentials](https://google.qwiklabs.com/quests/23) Quest. A Quest is a series of related labs that form a learning path. Completing this Quest earns you the badge above, to recognize your achievement. You can make your badge (or badges) public and link to them in your online resume or social media account. [Enroll in this Quest](https://google.qwiklab.com/learning_paths/23/enroll) and get immediate completion credit if you've taken this lab. [See other available Qwiklabs Quests](https://google.qwiklabs.com/catalog).

Take Your Next Lab

Continue your Quest with [Hello Node Kubernetes](https://google.qwiklabs.com/catalog_lab/468), or check out these suggestions:

* [Provision Services with Google Cloud Marketplace](https://google.qwiklabs.com/catalog_lab/339)
* [Cloud Monitoring: Qwik Start](https://google.qwiklabs.com/catalog_lab/2572)