Module 1: Labs and Demos

Lab 1: Console and Cloud Shell Console and Cloud Shell

In this lab you will become familiar with the GCP web-based interface including Console, the GUI (graphical user interface) environment, and Cloud Shell, the CLI (command line interface) environment.

Task 1: Create a bucket using the GCP Console

In this task, you create a bucket. However, the text also helps you become familiar with how actions are presented in the lab instructions in this class and teaches you about the GCP Console interface.

Navigate to the Storage service and create the bucket

- In the GCP Console, on the **Products & services** (☰) menu, click **Storage** > **Browser**.
- Click **Create bucket**.
- For **Name**, type a globally unique bucket name, and leave all other values as their defaults.
- Click **Create**.

Explore features in the GCP Console

The Google Cloud Platform menu contains a Notifications (🔔) icon. Sometimes, feedback from the underlying commands is provided there. If you are not sure what is happening, check Notifications for additional information and history.

Task 2: Access Cloud Shell

In this section, you explore Cloud Shell and some of its features.
You can use the Cloud Shell to manage projects and resources via command line, without having to install the Cloud SDK and other tools on your computer.
Cloud shell provides the following:

- Temporary Compute Engine VM
- Command-line access to the instance via a browser
- 5 GB of persistent disk storage ($HOME dir)
- Pre-installed Cloud SDK and other tools
- gcloud: for working with Google Compute Engine and many GCP services
- gsutil: for working with Cloud Storage
- kubectl: for working with Google Container Engine and Kubernetes
- bq: for working with BigQuery

- Language support for Java, Go, Python, Node.js, PHP, and Ruby
- Web preview functionality
- Built-in authorization for access to resources and instances

After 1 hour of inactivity, the Cloud Shell instance is recycled. Only the /home directory persists. Any changes made to the system configuration, including environment variables, are lost between sessions.

Open Cloud Shell and explore features

- In the Google Cloud Platform menu, click **Activate Google Cloud Shell (>_)**. If prompted, click **Start Cloud Shell**. Cloud Shell opens at the bottom of the GCP Console window.

There are three icons to the far right of the Cloud Shell toolbar:

- **Hide/Restore:** The first one hides and restores the window, giving you full access to the GCP Console without closing Cloud Shell.
- **Open in new window:** Having Cloud Shell at the bottom of the GCP Console is useful when you are issuing individual commands. However, sometimes you will be editing files or want to see the full output of a command. For these uses, this icon pops the Cloud Shell out into a full-sized terminal window.
- **Close all tabs:** This icon closes Cloud Shell. Every time you close Cloud Shell, the virtual machine is recycled and all machine context is lost.
- Close Cloud Shell now.

Task 3: Create a bucket using Cloud Shell

Create a second bucket and verify in the GCP Console

- Open Cloud Shell again.
- Use the gsutil command to create another bucket. Replace <BUCKET_NAME> with a globally unique name (you can append a 2 to the globally unique bucket name you used previously):

```
gsutil mb gs://<BUCKET_NAME>
```

- In the GCP Console, on the **Products & services** (☰) menu, click **Storage** > **Browser**, or click **Refresh** if you are already in the Storage Browser. The second bucket should be displayed in the **Buckets** list.

You have performed equivalent actions using the GCP Console and Cloud Shell.
You created a bucket using the GCP Console and another bucket using Cloud Shell.

Task 4: Explore more Cloud Shell features

Upload a file

- Open Cloud Shell.
- Click the three dots (⋮) icon in the Cloud Shell toolbar to display further options.
- Click **Upload file**. Upload a file from your local machine to the Cloud Shell VM. This file will be referred to as [MY_FILE].
- In Cloud Shell, type ls to confirm that the file was uploaded.
- Copy the file into one of the buckets you created earlier in the lab. Replace

[MY_FILE] with the file you uploaded and [BUCKET_NAME] with one of your bucket names:

```
gsutil cp [MY_FILE] gs://[BUCKET_NAME]
```

If your filename has whitespaces, ensure to place single quotes around the filename. For example, gsutil cp ‘my file.txt' gs://[BUCKET_NAME]

You have uploaded a file to the Cloud Shell VM and copied it to a bucket.

- In the GCP Console, on the **Products & services** (☰) menu, click **Storage** > **Browser**, select the buckets you created, and delete them.
- Confirm the bucket deletion by clicking DELETE.
- Explore the options available in Cloud Shell by clicking on different icons in the Cloud Shell toolbar.
- Close all the Cloud Shell sessions.

Task 5: Create a persistent state in Cloud Shell

In this section you will learn a best practice for using Cloud Shell. The gcloud command often requires specifying values such as a **Region** or **Zone** or **Project ID**. Entering them repeatedly increases the chances of making typing errors. If you use Cloud Shell a lot, you may want to set common values in environment variables and use them instead of typing the actual values.

Identify available regions

- Open Cloud Shell from the GCP Console. Note that this allocates a new VM for you.
- To list available regions, execute the following command:

```
gcloud compute regions list
```

- Select a region from the list and note the value in any text editor. This region will now be referred to as [YOUR_REGION] in the remainder of the lab.

Create and verify an environment variable

- Create an environment variable and replace [YOUR_REGION] with the region you selected in the previous step:

```
INFRACLASS_REGION=[YOUR_REGION]
```

- Verify it with echo:

```
echo $INFRACLASS_REGION
```

You can use environment variables like this in gcloud commands to reduce the opportunities for typos, and so that you won't have to remember a lot of detailed information.

Every time you close Cloud Shell and reopen it, a new VM is allocated, and the environment variable you just set disappears. In the next steps, you create a file to set the value so that you won't have to enter the command each time Cloud Shell is cycled.

Append the environment variable to a file

- Create a subdirectory for materials used in this class:

```
mkdir infraclass
```

- Create a file called config in the infraclass directory:

```
touch infraclass/config
```

- Append the value of your Region environment variable to the config file:

```
echo INFRACLASS_REGION=$INFRACLASS_REGION >> ~/infraclass/config
```

- Create a second environment variable for your Project ID, replacing [YOUR_PROJECT_ID] with your Project ID. You can find the project ID on the GCP Console Home page.

```
INFRACLASS_PROJECT_ID=[YOUR_PROJECT_ID]
```

- Append the value of your Project ID environment variable to the config file:

```
echo INFRACLASS_PROJECT_ID=$INFRACLASS_PROJECT_ID >>
~/infraclass/config
```

- Use the source command to set the environment variables, and use the echo command to verify that the project variable was set:

```
source infraclass/config
echo $INFRACLASS_PROJECT_ID
```

This gives you a method to create environment variables and to easily recreate them if the Cloud Shell is cycled. However, you will still need to remember to issue the source command each time Cloud Shell is opened.

In the next step you will modify the .profile file so that the source command is issued automatically any time a terminal to Cloud Shell is opened.

- Close and re-open Cloud Shell. Then issue the echo command again:

```
echo $INFRACLASS_PROJECT_ID
```

There will be no output because the environment variable no longer exists.

Modify the bash profile and create persistence

- Edit the shell profile with the following command:

```
nano .profile
```

- Add the following line to the end of the file:

```
source infraclass/config
```

- Press **Ctrl+O**, **ENTER** to save the file, and then press **Ctrl+X** to exit nano.
- Close and then re-open Cloud Shell to cycle the VM.
- Use the echo command to verify that the variable is still set:

```
echo $INFRACLASS_PROJECT_ID
```

You should now see the expected value that you set in the config file.

If you ever find your Cloud Shell environment corrupted, you can find instructions on resetting it here:

Resetting Cloud Shell

This will cause everything in your Cloud Shell environment to be set back to its original default state.

Task 6: Review the GCP interface

Cloud Shell is an excellent interactive environment for exploring GCP using Google Cloud SDK commands like gcloud and gsutil.

You can install the Google Cloud SDK on a computer or on a VM instance in GCP.

The gcloud and gsutil commands can be automated using a scripting language like bash (Linux) or Powershell (Windows). You can also explore using the command-line tools in Cloud Shell, and then use the parameters as a guide for re-implementing in the SDK using one of the supported languages.

The GCP interface consists of two parts: the GCP Console and Cloud Shell.

The Console:

- Provides a fast way to get things done
- Presents options to you, instead of requiring you to know them

- Performs behind-the-scenes validation before submitting the commands

Cloud Shell provides:

- Detailed control
- Complete range of options and features
- A path to automation through scripting

Cleanup

- In the **Cloud Platform Console**, sign out of the Google account.
- Close the browser tab.

Last Updated: 2018-04-30

End your lab

Lab 2: Infrastructure Preview

Infrastructure Preview

In this lab you will use Cloud Launcher to examine some of the powerful infrastructure features available in GCP. Many of the services that are used automatically in this lab will be explored in detail later in the class.

Task 1: Use Cloud Launcher to build a deployment

Navigate to Cloud Launcher

- In the GCP Console, on the **Products & Services** menu (☰)**Cloud Launcher**.
- Locate the Jenkins deployment by searching for **Jenkins Certified by Bitnami**.
- Click on the deployment and read about the service provided by the software.

Jenkins is an open-source continuous integration environment. You can define jobs in Jenkins that can perform tasks such as running a scheduled build of software and backing up data. Notice the software that is installed as part of Jenkins shown in the left side of the description.

The service you are using, Cloud Launcher, is part of Google Cloud Platform. The Jenkins template is developed and maintained by an ecosystem partner named Bitnami. Notice on the left side a field that says "Last updated." How recently was this template updated?

The template system is part of another GCP service called Deployment Manager. Later in this class you learn how templates such as this one can be built. That service is available to you. You can create templates like the one you are about to use.

In a class that was previously offered, students would set up a Jenkins environment similar to the one you are about to launch. It took about two days of labs to build the infrastructure that you will achieve in the next few minutes.

Launch Jenkins

- Click **Launch on Compute Engine**.
- Verify the deployment, and click **Deploy**.
- Click **Close** on the Welcome to Deployment Manager window.

It will take a minute or two for Deployment Manager to set up the deployment. You can watch the status as tasks are being performed. Deployment Manager is acquiring a virtual machine instance and installing and configuring software for you. You will see **jenkins-1 has been deployed** when the process is complete.

Deployment Manager is a GCP service that uses templates written in a combination of YAML, python, and Jinja2 to automate the allocation of GCP resources and perform setup tasks. Behind the scenes a virtual machine has been created. A startup script was used to install and configure software, and network Firewall Rules were created to allow traffic to the service.

Task 2: Examine the deployment

In this section, you examine what was built in GCP.

View installed software and login to Jenkins

- In the right pane, click **More about the software** to view additional software details. Look at all the software that was installed.
- Copy the **Admin user** and **Admin password** values to a text editor.
- Click **Visit the site** to view the site in another browser tab. If you get an error, you might have to reload the page a couple of times.
- Log in with the **Admin user** and **Admin password** values.
- After logging in, you will be asked to Customize Jenkins. Click **Install suggested plugins**, and then click **Restart** after the installation is complete. The restart will take a couple of minutes.

Explore Jenkins

- In the Jenkins interface, in the left pane, click **Manage Jenkins**. Look at all of the actions available. You are now prepared to manage Jenkins. The focus of this lab is GCP infrastructure, not Jenkins management, so seeing that this menu is available is the purpose of this step.
- Leave the browser window open to the Jenkins service. You will use it in the next task.

Now you have seen that the software is installed and working properly. In the next task you will open an SSH terminal session to the VM where the service is hosted, and verify that you have administrative control over the service.

Task 3: Administer the service

View the deployment and SSH to the VM

- In the GCP Console, on the **Products & Services** menu (☰), click **Deployment Manager**.
- Click **jenkins-1**.
- Click **SSH** to connect to the Jenkins server.

The Console interface is performing several tasks for you transparently. For example, it has transferred keys to the virtual machine that is hosting the Jenkins software so that you can connect securely to the machine using SSH.

Shut down and restart the services

- In the SSH window, enter the following command to shut down all the running services:

```
sudo /opt/bitnami/ctlscript.sh stop
```

- Refresh the browser window for the Jenkins UI. You will no longer see the Jenkins interface because the service was shut down.
- In the SSH window, enter the following command to restart the services:

```
sudo /opt/bitnami/ctlscript.sh restart
```

- Return to the browser window for the Jenkins UI and refresh it. You may have to do it a couple of times before the service is reachable.
- In the SSH window, type `exit` to close the SSH terminal session.

Task 4: Review

In a few minutes you were able to launch a complete Continuous Integration solution. You demonstrated that you had user access through the Jenkins UI, and you demonstrated that you had administrative control over Jenkins by using SSH to connect to the VM where the service is hosted and by stopping and then restarting the services.

Cleanup

- In the **Cloud Platform Console**, sign out of the Google account.
- Close the browser tab.

Last Updated: 2018-02-15

End your lab

Module 2: Labs and Demos

Overview

In this lab you build and explore a complex GCP network structure. In most labs you choose the regions and zones where objects are located; however, this lab is prescriptive about the network layout. The lab systematically highlights the differences between placing instances in a variety of network locations and depending on the instances relative location, how you establish communications between virtual machines.

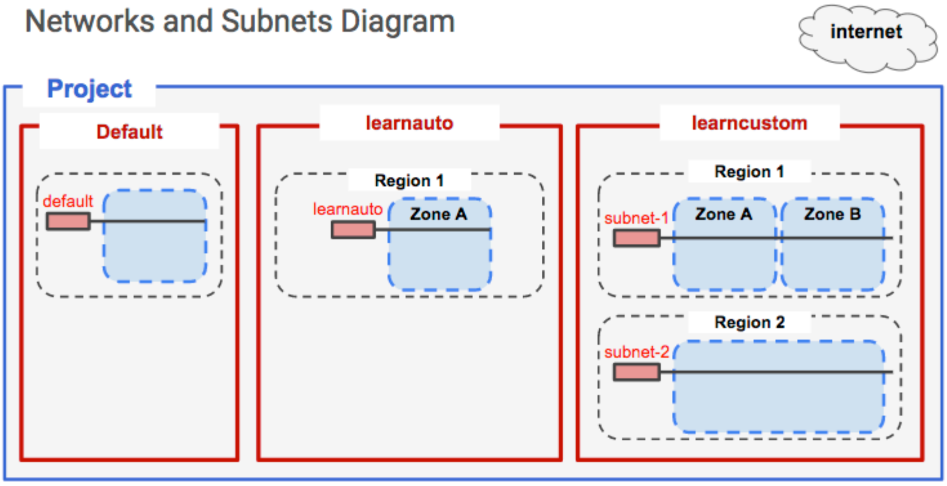
Objectives

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

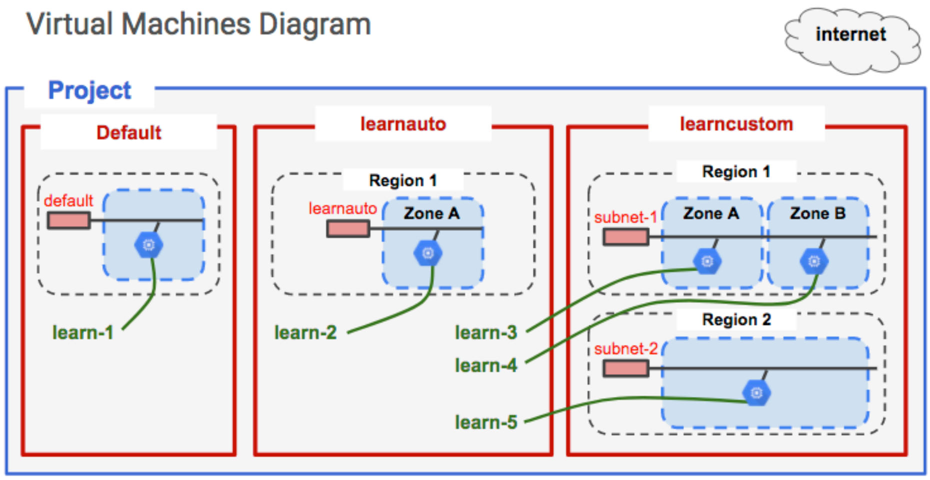
- Create an auto-mode network, a custom-mode network, and associated subnetworks
- Compare connectivity in the various types of networks
- Create routes and firewall rules using IP addresses and tags to enable connectivity
- Convert an auto-mode network to a custom-mode network
- Create, expand, and delete subnetworks

Here is a preview of the lab activities and the networks you will create:

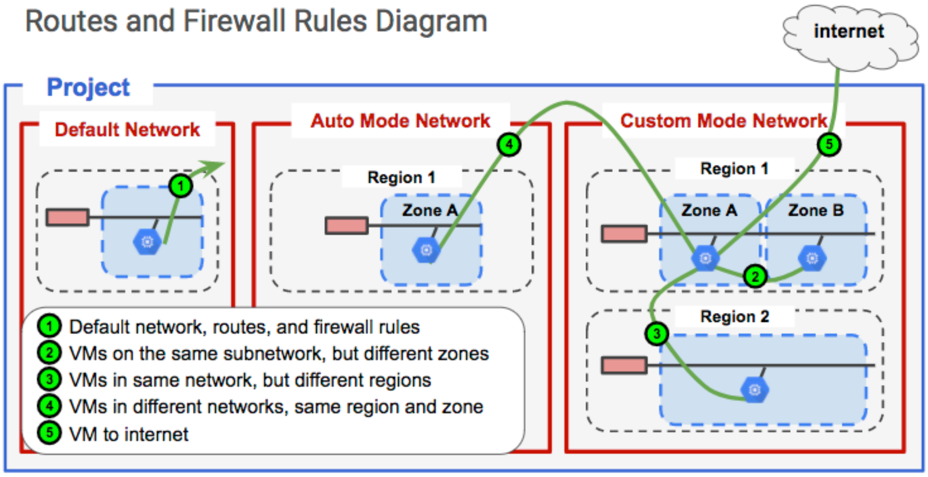
Task 1: Create the network topology



Task 2: Create the VM instances



Task 3: Work with routes and firewall rules


Routes and Firewall Rules Diagram
internet
Project
Default Network
Auto Mode Network
Region 1
Zone A
Custom Mode Network
Region 1
Zone A
Zone B
Region 2
1
2
3
4
5
1 Default network, routes, and firewall rules
2 VMs on the same subnetwork, but different zones
3 VMs in same network, but different regions
4 VMs in different networks, same region and zone
5 VM to internet

The scoping and connectivity relationships between zones, regions, networks, and subnets are different from networking in other public clouds.

You have been provided with a project in Qwiklabs. The project ID is a unique name across all Google Cloud projects. It is referred to later in this lab as PROJECT_ID.

Task 1: Create the network topology

Explore the default network

The default network is created automatically for you with each new project. The default network layout is not ideal for managing resources. The main benefit is that it is a fast way to get a project set up and running. The default network is great for prototyping solutions and for training purposes.

- In the Google Cloud Platform (GCP) Console, on the **Products & services** menu (☰), click **VPC network** > **VPC networks**.

Notice the default network. It was created automatically for you with a subnetwork in each region.

Example:

asia-east1 l default l 10.140.0.0/20 l 10.140.0.1

For more information, see:

IP Addresses: https://cloud.google.com/compute/docs/ip-addresses/

Subnets and CIDR ranges: https://cloud.google.com/compute/docs/aliasip/#subnets_and_cidr_ranges

- In the left pane, click **Routes**.

Notice that a route was created for each subnetwork, and one global route was created to enable traffic to the internet.

Create an auto-mode network and subnets

- In the left pane, click **VPC networks**.
- Click **Create VPC network**.
- Specify the following:

Property	**Value** (type value or select option as specified)
Name	**learnauto**
Description	**Learn about auto-mode networks**
Subnet creation mode	**Automatic**

When you click **Automatic**, the list of subnetworks to be created is automatically displayed.

- For **Firewall rules**, select all listed firewall rules.
- At the bottom of the page are two links labeled **Equivalent REST** or **command line**. Click **REST** to see POST commands for API programming automation of this process.
- Click **Close**.
- Click **command line** to see commands you could use for automation of this process. You could use these commands to create the network by clicking RUN IN CLOUD SHELL—but don't do it.

Note: These commands tend to include options that are not required. They may not work in a bash script without being altered. Don't rely on them. You should consider these more of a suggestion. If you need to automate with scripts, plan to craft your own commands from examples in the documentation.

- Click **Close**.
- Click **Create**.
- Click **REFRESH** occasionally until the networks are created and appear in the list.

Explore the auto-mode network

- In the left pane, click **Routes**.

Notice that a route has been created for each subnetwork, and one route was created to enable traffic from anywhere, including the internet. Traffic is delivered via the most specific matching route: traffic intended for any of the listed subnets gets delivered via virtual network to the host. These routes take precedence over the route that matches all traffic.

- Click **Destination IP ranges** to sort the list of routes.

Notice that there is an identical subnetwork and route in the learnauto network as there is in the default network. It is possible to have VMs with duplicate Internal IP addresses in the two networks.

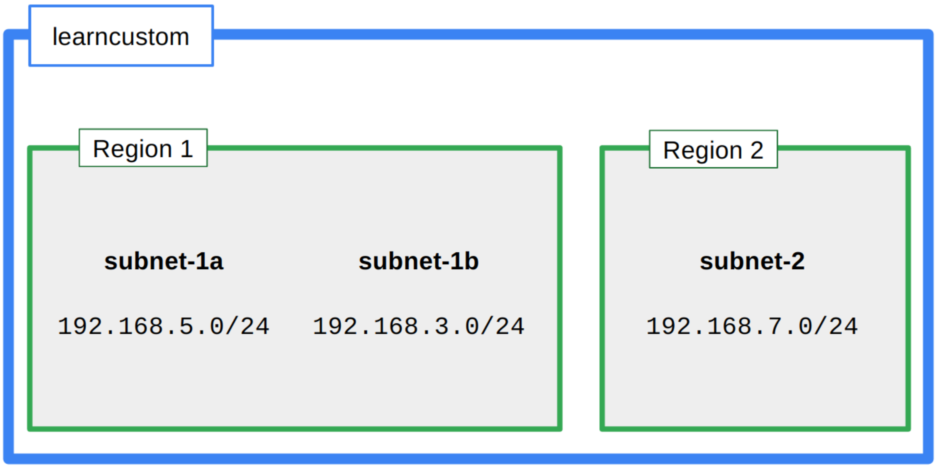
- In the left pane, click **Firewall rules**.

Verify that firewall rules were created for the learnauto network and its subnetworks. If you delete your default network, you can always recreate it as an auto network using the name "Default."

Create a custom-mode network

In this subtask, you create a custom-mode network named **learncustom** with three subnetworks:

- (**subnet-1a**) 192.168.5.0/24
- (**subnet-1b**) 192.168.3.0/24, in the same region
- (**subnet-2**) 192.168.7.0/24 in a different region



- In the left pane, click **VPC networks**.
- Click **Create VPC network**.
- Specify the following:

Property	**Value** (type value or select option as specified)
Name	**learncustom**
Description	**Learn about custom networks**
Subnet creation mode	**Custom**

Use the dialog to add three subnets as follows.

- For the first subnet, specify the following:

Property	**Value** (type value or select option as specified)
Name	**subnet-1a**
Region	**us-east1**
IP address range	**192.168.5.0/24**

- Click **Add subnet**.
- For the second subnet, specify the following:

Property	**Value** (type value or select option as specified)
Name	**subnet-1b**
Region	**us-east1**
IP address range	**192.168.3.0/24**

- Click **Add subnet**.
- For the third subnet, specify the following:

Property	**Value** (type value or select option as specified)
Name	**subnet-2**
Region	**us-west1**
IP address range	**192.168.7.0/24**

- Click **Create**.

Explore the routes and firewall rules

Did creating the custom network automatically create routes?

- In the left pane, click **Routes**.
- Click **Network** in the table header to sort by network name. Routes should be displayed for each subnetwork.

Did creating the custom network automatically create firewall rules?

- In the left pane, click **Firewall rules**.
- Click **Network** in the table header to sort by network name. No default firewall rules were created for the custom network. You will have to manually add default rules in the next step.

Create firewall rules for the learncustom network

Notice that for the other networks, the default network and the learnauto network, GCP automatically created default firewall rules allowing SSH traffic (tcp:22), icmp traffic, and rdp (tcp:3389) traffic for Windows VMs. Add a firewall rule to provide the same access for the learncustom network.

- Click **Create firewall rule**.
- Specify the following, and leave the remaining settings as their defaults:

Property	**Value** (type value or select option as specified)
Name	**allow-ssh-icmp-rdp-learncustom**
Network	**learncustom**
Target tags	**allow-defaults**
Source IP ranges	**0.0.0.0/0**
Protocols and ports	**Specified protocols and ports** Type: **icmp; tcp:22; tcp:3389**

Make sure that the source filter address includes the final "/0". If you specify 0.0.0.0 instead of 0.0.0.0/0, the filter defaults to 0.0.0.0/32, which is an exact host address that doesn't exist.

- Click **Create**.

Create firewall rules for the learncustom network

In this subtask, you attempt to modify the network by adding a subnet with an overlapping address range but in a different region. What do you predict will happen?

- In the left pane, click **VPC networks**.
- Click **learncustom**.
- Click **Add subnet**.
- Specify the following, leaving all other values with their defaults:

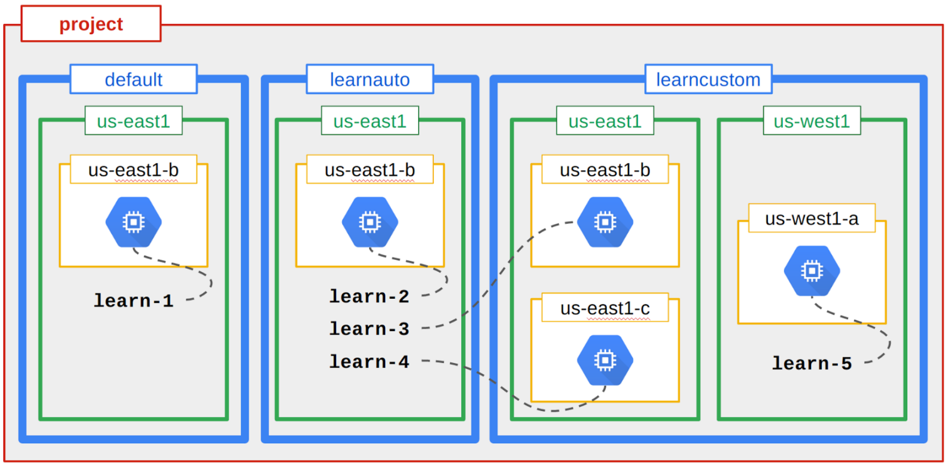
Property	**Value** (type value or select option as specified)
Name	**subnet-3**
Region	**europe-west1**
IP address range	**192.168.5.0/24**

The IP address range label is displayed in red with the following error message: "This IP address range overlaps with a subnet you already added. Enter an address range that doesn't overlap."

- Click **CANCEL**.

Task 2: Create the VM instances

To explore the Cloud Virtual Network, you create five micro VMs in different locations in the network. You will not install any additional software on them. They will not run any applications. You will just use them to explore the connectivity across the topologies in the network.



Name	Network	Region	Zone
learn-1	default	us-east1	us-east1-b
learn-2	learnauto	us-east1	us-east1-b
learn-3	learncustom	us-east1	us-east1-b
learn-4	learncustom	us-east1	us-east1-c
learn-5	learncustom	us-west1	us-west1-a

Create the learn-1 VM

- On the **Products & services** menu (☰), click **Compute Engine** > **VM instances**.
- Click **Create**.
- Specify the following, and leave the remaining settings as their defaults:

Property	**Value** (type value or select option as specified)
Name	**learn-1**
Zone	**us-east1-b**
Machine type	**micro (1 shared vCPU)**

- Click **Management, disks, networking, SSH keys** to access the advanced options.
- Click **Networking**. The default network interface should already be selected.
- Click **Create**.

Create the learn-2 VM

- Click **Create instance**.
- Specify the following, and leave the remaining settings as their defaults:

Property	**Value** (type value or select option as specified)
Name	**learn-2**

Zone	**us-east1-b**
Machine type	**micro (1 shared vCPU)**

- Click **Management, disks, networking, SSH keys** to access the advanced options.
- Click **Networking**.
- Click the pencil icon to edit **Network interfaces**.
- Specify the following, and leave the remaining settings as their defaults:

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

- Click **Done**.
- Click **Create**.

Create the learn-3 VM

- Click **Create instance**.
- Specify the following, and leave the remaining settings as their defaults:

Property	**Value** (type value or select option as specified)
Name	**learn-3**
Zone	**us-east1-b**
Machine type	**micro (1 shared vCPU)**

- Click **Management, disks, networking, SSH keys** to access the advanced options.
- Click **Networking**.
- Click the pencil icon to edit **Network interfaces**.
- Specify the following, and leave the remaining settings as their defaults:

Property	**Value** (type value or select option as specified)
Network	**learncustom**
Subnetwork	**subnet-1a**

- Click **Done**.
- Click **Create**.

Create the learn-4 VM

- Click **Create instance**.
- Specify the following, and leave the remaining settings as their defaults:

Property	**Value** (type value or select option as specified)
Name	**learn-4**
Zone	**us-east1-c**
Machine type	**micro (1 shared vCPU)**

- Click **Management, disks, networking, SSH keys** to access the advanced options.
- Click **Networking**.
- Click the pencil icon to edit **Network interfaces**.
- Specify the following, and leave the remaining settings as their defaults:

Property	**Value** (type value or select option as specified)

Network	**learncustom**
Subnetwork	**subnet-1b**

- Click **Done**.
- Click **Create**.

Create the learn-5 VM

- Click **Create instance**.
- Specify the following, and leave the remaining settings as their defaults:

Property	Value (type value or select option as specified)
Name	**learn-5**
Zone	**us-west1-a**
Machine type	**micro (1 shared vCPU)**

- Click **Management, disks, networking, SSH keys** to access the advanced options.
- Click **Networking**.
- Click the pencil icon to edit **Network interfaces**.
- Specify the following, and leave the remaining settings as their defaults:

Property	Value (type value or select option as specified)
Network	**learncustom**
Subnetwork	**subnet-2**

- Click **Done**.
- Click **Create**.

Verify that all the test VMs are running

- On the **VM instances** page, verify that all 5 instances are running.

Task 3: Work with routes and firewall rules

In this task, you connect via SSH to the VMs and use ping to test connectivity between VMs. This helps you understand how the Cloud Virtual Network topology behaves.

Google Cloud Platform (GCP) Virtual Private Cloud (VPC) networks have an internal DNS service that allows you to use instance names instead of instance IP addresses to refer to Compute Engine virtual machine (VM) instances.

Each instance has a metadata server that also acts as a DNS resolver for that instance. DNS lookups are performed for instance names. The metadata server itself stores all DNS information for the local network and queries Google's public DNS servers for any addresses outside of the local network.

An instance is not aware of any external IP address assigned to it. Instead, the network stores a lookup table that matches external IP addresses with the internal IP addresses of the relevant instances.

To break out of the ping command at any time, press **Ctrl+C**.

ping from learn-1 and learn-2

- On the **VM instances** page, for **learn-1**, click **SSH**.
- Run the following command:

```
ping learn-1
```

Notice how DNS translates for you. This should execute and display no packet loss.

- Now try to reach learn-2:

```
ping learn-2
```

Can you explain why this fails?

It is because DNS is scoped to network. The VM learn-2 is not in the default network where learn-1 is located. So the symbolic name can't be translated.

Locate the internal IP address and the external IP address for learn-2.

- Try to ping learn-2's internal IP address:

```
ping <learn-2's internal IP>
```

Did this work?

In a few cases you may try to ping the internal IP of the other machine and it succeeds! Do you know why this would be the case?

Because ... the internal IP of the machine you are using could be the same as the internal IP of the VM in the other network. In this case, the ping would succeed because you are actually pinging your own local VM's interface, not the one on the other VM in the other network. You can't ping an internal IP address that exists in a separate network than your own.

When you create a new auto-mode network, the IP ranges will be identical to the ranges in the default network. The first address in the range is always reserved for the gateway address. So it is actually likely that the first VM in a zone will have the same address as the first VM in the corresponding zone in another network.

If it didn't work... learn-1 is in the default network and learn-2 is in the learnauto network. Even though both VMs are located in the same region, us-east1, and in the same zone, us-east-1b, they cannot communicate over internal IP.

- Try to ping learn-2's external IP address:

```
ping <learn-2's external IP>
```

This works.

traceroute from learn-1

- From the learn-1 SSH terminal, install traceroute:

```
sudo apt-get install traceroute
```

- Verify that traceroute is working by tracing the route to a public website:

```
sudo traceroute google.com -I
```

Did it work? Yes.

- Now use traceroute to find the path to learn-2's external IP:

```
sudo traceroute <learn-2's external IP> -I
```

How many hops was it from learn-1 to learn-2's external IP? One.

ping to learn-3

You already know that learn-3 is in a different network from learn-1, so the internal IP for learn-3 will not be reachable.

- Try to ping learn-3's external IP address:

```
ping <learn-3's external IP>
```

- Press **Ctrl+C** to stop the command.

Why didn't this work? You were able to reach learn-2's external IP; why not learn-3's?

Recall that learn-2 is in an auto-mode network, so firewall rules were automatically created that enabled ingress traffic to reach its external IP. However, learn-3 is in a custom-mode network, and no firewall rules were established. You created a firewall rule to permit access.

Take another look at that firewall rule.

- In the GCP Console, on the **Products & services** menu (☰), click **VPC network** > **Firewall rules**.

Notice that the default firewall rules were established to apply to all targets. You created the firewall rule with tighter security. It will only permit traffic to VMs that have the Target tag allow-defaults.

- On the **Products & services** menu (☰), click **Compute Engine** > **VM instances**.
- Click **learn-3** to access details about the VM.
- Click **edit**.
- For **Network tags**, type **allow-defaults**
- Click **Save**.
- Return to the SSH terminal for **learn-1** (or reconnect if needed).
- Try again to ping learn-3's external IP address:

```
ping <learn-3's external IP>
```

The firewall rule and network tags can take time to take effect; if the last step didn't work, wait a few minutes and try again.

Edit the firewall rule

You already know that learn-3 is in a different network from learn-1, so the internal IP for learn-3 will not be reachable.

- Open an SSH terminal to **learn-3**.
- Try the following:

```
ping learn-4
ping learn-5
sudo apt-get install traceroute
sudo traceroute learn-5 -I
```

Can you explain all the behaviors?

DNS translation works for both learn-4 and learn-5 because all of these VMs are in the same network as learn-3, the learncustom network. Pinging the IP addresses will work after the firewall rules have been added.

- In the GCP Console, in the left pane, click **VM instances**.
- Try to connect via SSH to learn-4.

The firewall rule for the learncustom network only delivers traffic to VMs with the target tag allow-defaults.

- In the GCP Console, on the **Products & services** menu (☰), click **VPC network** > **Firewall rules**.
- Click **allow-ssh-icmp-rdp-learncustom** to access the firewall rule details.
- Click **Edit**.
- For **Targets**, click **All instances in the network**.
- Click **Save**.
- Try the commands again:

```
ping learn-4
ping learn-5
sudo apt-get install traceroute
sudo traceroute learn-5 -I
```

Everything should work this time.

- Verify that you can now connect via SSH to learn-4.

Convert an auto-mode network to a custom-mode network

In this section, you convert an auto-mode network to a custom-mode network to gain more fine-grained control over the subnetworks.

A new policy for network learnauto will be implemented. There will no longer be assets in us-central1 region. New projects instead shift planned assets from us-central1 to a new subnetwork in us-east1 region named new-useast.

To implement the policy, you delete the learnauto us-central1 subnetwork and create the new subnetwork in us-east1 to allow for the work that was originally planned for the us-central1 region.

- In the GCP Console, in the left pane, click **VPC networks**.
- Click **learnauto** to view network details.

Notice that there is no option to select the subnets. You can only delete the entire network.

You can't delete the subnetwork because this network is an auto-mode network. You will have to convert it to a custom-mode network to gain the ability to delete the subnetwork.

- Return to the **VPC networks** page.
- For **learnauto**, in the **Mode** column, switch from **Auto** to **Custom**.
- In the confirmation dialog, click **OK**.

Delete the learnauto subnet and create a new subnet

- Click **learnauto** to view network details.
- Click **learnauto** for the **us-central1** subnet.
- Click **Delete subnet**.
- In the confirmation dialog, click **Delete**.

Reflecting the new tighter policies, the new subnetwork is CIDR /26. How many VMs can that support?/26 = 64 addresses, minus broadcast, subnet, and gateway = 61 VMs.

- Return to the **VPC networks** page and click **learnauto** to return to the network details.
- Click **Add subnet**.
- Specify the following, and leave the remaining settings as their defaults:

Property	**Value** (type value or select option as specified)
Name	**new-useast**
Region	**us-east1**
IP address range	**10.133.5.0/26**

- Click **Add**.

Expand a subnet

The projects in the new-useast subnet have been a success; however, the original range of /26 was too restrictive. Expand the subnet to support at least 500 VMs.

- In the left pane, click **VPC networks**.
- Click **learnauto** to view network details.
- On the **Google Cloud Platform** menu, click **Activate Google Cloud Shell** (>_)to open Cloud Shell. If prompted, click **Start Cloud Shell**.
- To increase the address range, run the following command:

```
gcloud compute networks subnets \
expand-ip-range new-useast  \
--prefix-length 23 \
--region us-east1
```

- When prompted, type **Y** to continue.
- There is no refresh button on the **VPC network details** page to see the result; in the left pane, click **VPC networks**.
- Click **Refresh** until you see that the range has expanded.

Delete resources that are no longer needed

If you end the lab now, the lab infrastructure will clean up and delete all the resources. However, in this section, you delete objects so that you can explore the dependent relationship in deleting. Objects must be deleted in a specific order.

Before you can delete networks and subnets, you must delete all VMs and firewall rules.

- On the **Products & services** menu (☰), click **Compute Engine** > **VM instances**.
- Select all the VMs, and then click **Delete**.
- In the confirmation dialog, click **Delete**.
- On the **Products & services** menu (☰), click **VPC network** > **Firewall rules.**
- Delete all firewall rules that are part of the **learnauto** and **learncustom** networks.
- In the left pane, click **VPC networks**.
- Click **learncustom** to view network details.
- Click **Delete VPC network**.
- In the confirmation dialog, click **Delete**.
- Repeat steps 7–9 for the **learnauto** network.

Do **not** delete the Default network.

Review the delete project procedure

You do not have the IAM role necessary to delete the project. The following steps illustrate what the activity would look like if you could perform it.

However, if you were to delete the project, the process would look like this:

- In the Console, navigate to **Products & services** > **IAM & admin** > **Settings**.
- Click **Delete Project**.
- To shut down the project, you would need to type in the Project ID and click **Shut down**.

Task 4: Review

In this lab you created networks and subnetworks of many different varieties, started VMs in each location, and then explored the network relationship between them.

Cleanup

- In the **Cloud Platform Console**, sign out of the Google account.
- Close the browser tab.

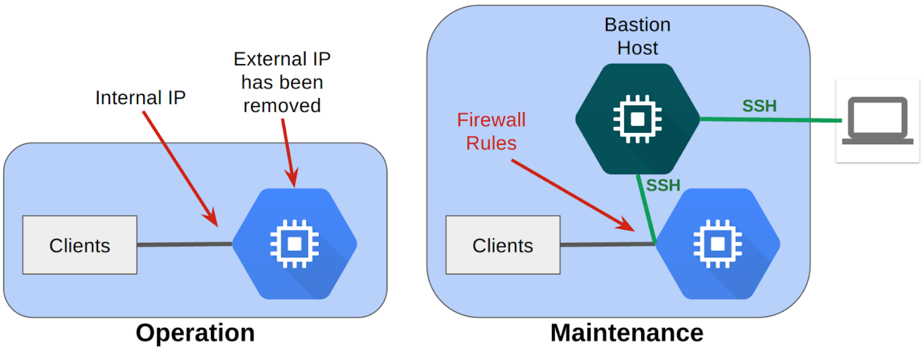
Last Updated: 2018-02-15

End your lab

Lab 2: Bastion Host

Overview

A best practice for infrastructure administration is to limit access to the resources. In this lab, you learn one method of hardening an infrastructure called a Bastion Host.



During operations, you harden the server by removing its external IP address, which prevents connections from the internet. During maintenance, you start up a bastion host that has an external IP address. You then connect via SSH to the bastion host, and from there to the server over the internal IP address. You can further restrict access with firewall rules.

Objectives

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

- Create an application web server to represent a service provided to an internal corporate audience
- Prevent the web server from access to or from the internet

Create a maintenance server, called a Bastion Host, to gain access to and verify internal connectivity to the application server

Task 1: Launch an instance and verify access

Launch an instance

- In the Console, on the **Products & services** menu (☰) click **Compute Engine** > **VM instances**.
- Click **Create**.
- Specify the following, and leave the remaining settings as their defaults:

Property	**Value** (type value or select option as specified)
Name	**webserver**
Zone	**us-central1-c**
Firewall	**Allow HTTP traffic**

- Click **Create**.

Verify IP access

- For **webserver**, click **SSH** to launch a terminal and connect.

Tip: Setting the Source IP at creation time is a best practice for this lab because it allows the initial SSH credentials to be set for you behind the scenes.

- Enter a few commands to test connectivity:

```
ls
pwd
```

- Enter the following command to close the terminal:

```
exit
```

Task 2: Restrict firewall rule settings for SSH

The default setting for a default or auto-type network is to allow SSH access from any source IP address. Restrict access to just your source IP address to see what happens when you try to connect from the GCP Console.

Find your IP address

Find the IP address of the computer you are using. One easy way to do this is to go to a website that provides this address.

- Open a browser in a new tab.
- Go to www.google.com and search for "what's my IP." It will either directly reply with your IP or give you a list of sites that perform this service.
- Ensure that the IP address only contains numerals (IPv4) and is not represented in hexadecimals (IPv6).
- Copy your IP address. It will be referred to as YOUR_IP_ADDRESS. You will be using it to modify the default firewall rule.

Edit the default SSH rule

- In the GCP Console, on the **Products & services** menu (), click **VPC network** > **Firewall rules**.
- Click the **default-allow-ssh** rule, and then click **Edit**.
- Specify the following, and leave the remaining settings as their defaults:

Property	**Value** (type value or select option as specified)
Description	**Allow SSH from my IP only**
Source IP ranges	Remove **0.0.0.0/0** Add **[YOUR_IP_ADDRESS]**

- Click **Save**. Wait until the firewall rule is updated (the status in the bottom pane is **Updating firewall rule**; when it closes, you can continue).

Test connectivity

- On the **Products & services** menu (), click **Compute Engine** > **VM instances**.
- For **webserver**, click **SSH** to launch a terminal and connect.

What happened?

When you connect via SSH to an instance from your browser, you need to allow SSH from Cloud Platform resources, so you must allow connections from either any IP address or from Google's IP address range, which you can get from Public SPF records. If you want to restrict SSH access to just your IP address, you need to SSH

from a terminal session.
For this lab, leaving SSH open to any connections is sufficient.

Reset the IP address range in the firewall rule

- In the GCP Console, on the **Products & services** menu (), click **VPC network** > **Firewall rules**.
- Click the **default-allow-ssh** rule, and then click **Edit**.
- Specify the following, and leave the remaining settings as their defaults:

Property	**Value** (type value or select option as specified)
Description	**Allow SSH from all IPs**
Source IP ranges	Add **0.0.0.0/0**

- Click **Save**. Wait until the firewall rule is updated (the status in the bottom pane is **Updating firewall rule**; when it closes, you can continue).

Verify the change

- On the **Products & services** menu (), click **Compute Engine** > **VM instances**.
- For **webserver**, click **SSH** to launch a terminal and connect. Leave the terminal open for the next task.

Task 3: Install a simple web application

Install a simple web application on your instance to represent an internal application. You then secure it by preventing access from the internet.

Install and configure a web server

- In the webserver SSH terminal, update the package index:

```
sudo apt-get update
```

- Install the apache2 package:

```
sudo apt-get install apache2 -y
```

- To create a new default web page by overwriting the default, run the following:

```
echo '<!doctype html><html><body><h1>Hello World!</h1></body></html>' | sudo
tee /var/www/html/index.html
```

Verify that the web server is working

Test that your instance is serving traffic on its external IP.

- In the GCP Console, on the **Products & services** menu (), click **Compute Engine** > **VM instances**.
- For **webserver**, click the **external IP** to open in a new tab.You should see the "Hello World!" page you updated earlier.

Task 4: Restrict firewall rule settings for HTTP

Restrict access to the web interface by changing the source IP address in the **default-allow-http** rule to your IP address.

Restrict HTTP access

- In the GCP Console, on the **Products & services** menu (), click **VPC network** > **Firewall rules**.
- Click the **default-allow-http** rule, and then click **Edit**.
- Specify the following, and leave the remaining settings as their defaults:

Property	Value (type value or select option as specified)
Description	**Allow HTTP from my IP only**
Source IP ranges	Remove **0.0.0.0/0** Add **[YOUR_IP_ADDRESS]**

- Click **Save**. Wait until the firewall rule is updated (the status in the bottom pane is **Updating firewall rule**; when it closes, you can continue).

Verify that you still have access to the web server

- On the **Products & services** menu (), click **Compute Engine** > **VM instances**.
- For **webserver**, click the **external IP** to open in a new tab.You should still see the "Hello World!" page.

Task 5: Restrict access to the VM from the internet

Edit the VM Properties

- Return to the **VM instances** page of the GCP Console.
- Click **webserver** to access the instance details.
- Click **Edit**.
- For **Network interfaces**, click the default network and change **External IP** from **Ephemeral** to **None**.
- Click **Done**.
- Click **Save**.

Try to access the VM

- First try HTTP: In the left pane, click **VM instances**. Notice that **webserver** doesn't have a value under **External IP**.
- Try SSH: for **webserver**, try to use the **SSH** link to launch a terminal and connect.

What happened?

The VM is no longer associated with an External IP. It is no longer reachable from the internet.

Task 6: Create a Bastion Host

Launch another instance

- Click **Create instance**.
- Specify the following, and leave the remaining settings as their defaults:

Property	Value (type value or select option as specified)
Name	**bastion**
Zone	**us-central1-c**

- Click **Create**.

Connect to the Bastion Host via SSH and verify access to webserver

- For **bastion**, click **SSH** to launch a terminal and connect.
- Verify that the home page on **webserver** is reachable from **bastion** by running the

following command:

```
curl webserver
```

Even though **webserver** is no longer associated with an external IP address, clients inside your network can still view and use the web service on this VM over the internal IP address.

- From the **bastion** SSH terminal, connect to **webserver** by running the following command:

```
ssh -a webserver
```

- When prompted, type **yes** to continue.

For more information on SSL Certificates in GCP, see:
https://cloud.google.com/compute/docs/load-balancing/http/ssl-certificates

When instances do not have external IP addresses, they can only be reached by other instances on the network or via a managed VPN gateway.

In this case, the bastion VM serves as a management and maintenance interface to the webserver VM.

Task 7: Review

You restricted access to the **webserver** VM by removing the external IP address. You created a bastion host named **bastion** to gain access to the webserver VM over its internal IP. Normally, you would harden the bastion host by restricting the source IPs that can access the bastion host, by editing the firewall rules just as you did earlier in this lab. When you're not using the bastion host, you can shut it down.

Cleanup

- In the **Cloud Platform Console**, sign out of the Google account.
- Close the browser tab.

Last Updated: 2018-03-27

End your lab

Module 3: Labs and Demos

Lab 1 Creating Virtual Machine

Task 1: Create a utility virtual machine

Create a VM

- In the GCP Console, on the **Products & Services** menu (☰), click **Compute Engine** > **VM instances**.
- Click **Create**.
- For **Name**, type a name for your instance. Hover over the question mark icon for advice about what constitutes a properly formed name.
- For **Zone**, examine all the zones available, and select any zone for the new VM.
- For **Machine type**, examine the options.

Notice that the menu lists the number of vCPUs, the amount of memory, and a symbolic name such as n1-standard-1. The symbolic name is the parameter you would use to select the machine type if you were creating a VM using the gcloud command. Notice to the right of the zone and machine type that there is a per-month estimated cost.

- Click **Details** to the right of the **Machine type** list to see the breakdown of estimated costs.
- For **Machine type**, click **16 vCPUs (n1-standard-16)**. How did the cost change?
- For **Machine type,** click **micro (1 shared vCPU)**. The micro type is a shared tenant VM that is inexpensive.
- Leave the remaining settings as their defaults, and click **Create**. Wait until the new VM is created.

Explore the VM details

- On the **VM instances** page, click on the name of your VM.
- Locate **CPU platform** and note the value. Click **Edit**.

Notice that you can't change the machine type, the CPU platform, or the zone.

You can add network tags and allow specific network traffic from the internet through firewalls.

Some properties of a VM are integral to the VM, are established when the VM is created, and cannot be changed. Other properties can be edited. You can add additional disks and you can also determine whether the boot disk is deleted when the instance is deleted. Normally the boot disk defaults to being deleted automatically when the instance is deleted. But sometimes you will want to override this behavior. This feature is very important because you cannot create an image from a boot disk when it is attached to a running instance. So you would need to disable **Delete boot disk when instance is deleted** to enable creating a system image from the boot disk.

- Examine **Availability policies**.

You can't convert a non-preemptible instance into a preemptible one. This choice must be made at VM creation. A preemptible instance can be interrupted at any time and is available at a lower cost.

If a VM is stopped for any reason, (for example an outage or a hardware failure) the automatic restart feature will start it back up. Is this the behavior you want? Are your applications idempotent (written to handle a second startup properly)?

During host maintenance, the VM is set for live migration. However, you can have the VM terminated instead of migrated.

If you make changes, they can sometimes take several minutes to be implemented, especially if they involve networking changes like adding firewalls or changing the external IP.

- Click **Cancel**.

Explore the VM logs

- On the **VM instance details** page for your VM, click **Stackdriver Logging**.

Notice that you have now navigated to the Stackdriver Logging page.

This is a structured log view. At the top you can filter by using the pull-down menus, and there is a search box for searching based on labels or text.

- Click the small triangle to the left of one of the lines to see the kind of information it contains.
- On the far right, click **View Options** > **Expand All**.

Task 2: Create a Windows virtual machine

Create a VM

- On the **Products & services** menu (), click **Compute Engine** > **VM instances**.
- Click **Create instance**.
- Specify the following, and leave the remaining settings as their defaults:

Property	**Value** (type value or select option as specified)
Name	Type a name for your VM
Zone	Choose any zone
Machine type	**2 vCPUs (7.5 GB of memory, n1-standard-2)**
Boot disk	**Change**
OS images:	**Windows Server 2016** **(Server with Desktop Experience, x64 built on ...)**
Boot disk type:	**SSD persistent disk**
Size (GB):	**100**

- Click **Select**.
- For **Firewall**, enable **Allow HTTP traffic** and **Allow HTTPS traffic**.
- Click **Create**.

When the VM is running, notice that the connection option in the far right column is RDP, not SSH. RDP is the Remote Desktop Protocol. You would need the RDP client installed on your local machine to connect to the Windows desktop.

Note: Installing an RDP client on your local machine is outside the scope of this lab and of the class. For this reason, you will not be connecting to the Windows VM during this lab. However, you will step through the usual procedures up to the point of requiring the RDP client.

Instructions for connecting to Windows VMs are here: https://cloud.google.com/compute/docs/instances/windows/connecting-to-windows-instance

Set the password for the VM

- Click on the name of your Windows VM to access the **VM instance details**.
- You don't have a valid password for this Windows VM: you cannot log in to the Windows VM without a password. Click **Set Windows password**.
- Click **Set**.
- Copy the provided password, and click **CLOSE**.

You will **not** connect to the Windows VM during this lab. However, the process would look something like the following (depending on the RDP client you installed). The RDP client shown can be installed for Chrome here: https://chrome.google.com/webstore/detail/chrome-rdp-for-google-clo/mpbbnannobiobpnfblimoapbephgifkm?hl=en-US

On the **VM instances** page, you would click **RDP** for your Windows VM and connect with the password copied earlier.

Task 3: Create a custom virtual machine

Create a VM

- On the **Products & Services** menu (), click **Compute Engine** > **VM instances**.
- Click **Create instance**.
- Specify the following, and leave the remaining settings as their defaults:

Property	**Value** (type value or select option as specified)
Name	Type a name for your VM
Zone	Choose any zone
Machine type	**Customize**
Cores	**12 vCPU**
Memory	**68 GB**

- Click **Create**.

Connect via SSH to your custom VM

- For the custom VM you just created, click **SSH**.
- To see information about unused and used memory and swap space on your custom VM, run the following command:

```
free
```

- To see details about the RAM installed on your VM, run the following command:

```
sudo dmidecode -t 17
```

- To verify the number of processors, run the following command:

```
nproc
```

- To see details about the CPUs installed on your VM, run the following command:

```
lscpu
```

- To exit the SSH terminal, run the following command:

```
exit
```

Task 4: Delete the VMs

Delete all your created VMs

- Return to the VM instances page in the GCP Console, and select all three VMs that you created.
- Click **Delete**.
- In the confirmation dialog, click **Delete**.

Task 5: Review

In this lab, you created several virtual machine instances of different types with different characteristics. One was a small utility VM for administration purposes. You also created a standard VM and a custom VM. You launched both Windows and Linux VMs and deleted VMs.

Cleanup

- In the **Cloud Platform Console**, sign out of the Google account.
- Close the browser tab.

Last Updated: 2018-03-27

End your lab

Module 4: Labs and Demos

Cloud Identity and Access Management (IAM)

Task 1: Setup for two users

Sign in to the GCP Console as the first user

- For this lab, Qwiklabs has provisioned you with two user names available in the **Connection Details** dialog. Sign in to the GCP Console in an Incognito window as usual with the **Username 1** provided in Qwiklabs. Note that both user names use the same single password.

Sign in to the GCP Console as the second user

- Open another tab in your incognito window.
- Browse to **console.cloud.google.com**.
- Click on the user icon in the top-right corner of the screen, and then click **Add account**.
- Sign in to the GCP Console with the **Username 2** provided in Qwiklabs.

Note: At some points in this lab, if you sign out of the **Username 1** account, the **Username 2** account is deleted by Qwiklabs. So remain signed in to **Username 1** until you are done using **Username 2**.

Task 2: Explore the IAM console

Make sure you are on the **Username 1** GCP Console tab.

Navigate to the IAM console and explore roles

- On the **Products & Services** menu (☰), click **IAM & admin** > **IAM**.
- Click **Add** and explore the roles in the drop-down menu. Note the various roles associated with each resource by navigating the **Roles** menu.
- Click **Cancel**.
- Switch to the **Username 2** GCP Console tab.
- On the **Products & Services** menu (☰), click **IAM & admin** > **IAM**. Browse the list for the lines with the names associated with **Username 1** and **Username 2** in the Qwiklabs **Connection Details** dialog.

Username 2 currently has access to the project, but does not have the Project Owner role, so it cannot view the details.

- Switch back to the **Username 1** GCP Console tab.
- In the IAM console, for **Username 2**, click **Viewer**. **Username 2** currently has **Project Viewer** roles. Do not change the Project Role.

Task 3: Prepare a resource for access testing

Create a bucket and upload a sample file

- Switch to the **Username 1** GCP Console tab if you aren't already there.
- On the **Products & Services** menu (☰), click **Storage** > **Browser**.
- Click **Create bucket**.
- Specify the following, and leave the remaining settings as their defaults:

Property	**Value** (type value or select option as specified)
Name	Enter a globally unique name
Default storage class	**Multi-Regional**

Note the bucket name: it will be used in a later step and referred to as [YOUR_BUCKET_NAME]

- Click **Create**.
- Click **Upload files**.
- Upload any sample file from your local machine.
- After the upload completes, click **Close** on the upload window.
- When the file has been uploaded, click on the three dots at the end of the line containing the file, and click **Rename**.
- Rename the file to **sample.txt**, and click **Rename**.

Verify project viewer access

- Switch to the **Username 2** GCP Console tab.
- In the Console, navigate to **Products & services** > **Storage** > **Browser**.
- Verify that **Username 2** can see the bucket.

Task 4: Remove project access

Remove Project Viewer role for Username 2

- Switch to the **Username 1** GCP console tab.
- On the **Products & Services** menu (☰), click **IAM & admin** > **IAM**.
- For **Username 2**, click the **Delete** icon.
- Confirm by clicking **Remove**.

Notice that the user has disappeared from the list! The user has no access now.

Verify that Username 2 has lost access

- Switch to the **Username 2** GCP Console tab.
- On the **Products & Services** menu (☰), click **Home**.
- On the **Products & Services** menu (☰), click **Storage** > **Browser**. An error will be displayed. If not, refresh the page. **Username 2** still has a GCP account, but has no access to the project.

Task 5: Add storage access

Add storage permissions

- Copy the value of **Username 2** from the Qwiklabs **Connection Details** dialog.
- Switch to the **Username 1** GCP Console tab.
- On the **Products & Services** menu (☰), click **IAM & admin** > **IAM**.
- Click **Add** to add the user.
- For **New members**, paste the **Username 2** value you copied from the Qwiklabs **Connection Details** dialog.
- For **Select a role**, select **Storage** > **Storage Object Viewer**.

- Click **Add**.

Verify that Username 2 has storage access

- Switch to the **Username 2** GCP Console tab.

Username 2 doesn't have Project Viewer roles, so that user can't see the project or any of its resources in the Console. However, the user has specific access to Cloud Storage.

- To start Cloud Shell, click **Activate Google Cloud Shell** (>_). If prompted, click **START CLOUD SHELL**.
- To view the contents of the bucket you created earlier, run the following command, replacing [YOUR_BUCKET_NAME] with the unique name of the Cloud Storage bucket you created:

```
gsutil ls gs://[YOUR_BUCKET_NAME]
```

As you can see, **Username 2** has limited access to Cloud Storage.

- Close the **Username 2** GCP Console tab.The rest of the lab is performed on the **Username 1** GCP Console tab.
- Switch to the **Username 1** GCP Console tab.

Task 6: Set up the Service Account User

In this part of the lab, you assign narrow permissions to service accounts and learn how to use the Service Account User role.

Create a service account

- On the **Products & Services** menu (☰), click **IAM & admin** > **Service accounts**.
- Click **Create service account**.
- Specify the following, and leave the remaining settings as their defaults:

Property	**Value** (type value or select option as specified)
Service account name	**read-bucket-objects**
Role	**Storage** > **Storage Object Viewer**

- Click **Create**.
- In the confirmation dialog, click **Close**.

Add the user to the service account

- Click **read-bucket-objects**.
- Click **Permissions**.

You will grant the user the role of Service Account User, which allows that person to use a service account on a VM, if they have access to the VM.

You could perform this activity for a specific user, group, or domain.

For training purposes, you will grant the Service Account User role to everyone at a company called Altostrat.com. Altostrat.com is a fake company used for demonstration and training.

- Specify the following, and leave the remaining settings as their defaults:

Property	**Value** (type value or select option as specified)
Add members	**altostrat.com**

Select a role	**Service Account User**

- Click **Add**.

Grant Compute Engine access

You now give the entire organization at Altostrat the Compute Engine Admin role.

- On the **Products & Services** menu (☰), click **IAM & admin** > **IAM**.
- Click **Add**.
- Specify the following, and leave the remaining settings as their defaults:

Property	**Value** (type value or select option as specified)
New members	**altostrat.com**
Select a role	**Compute Engine** > **Compute Instance Admin (v1)**

- Click **Add**.
- In the confirmation dialog, click **Add**.

This step is a rehearsal of the activity you would perform for a specific user.

This action gives the user limited abilities with a VM instance. The user will be able to connect via SSH to a VM and perform some administration tasks.

Create a VM with the Service Account User

- On the **Products & Services** menu (☰), click **Compute Engine** > **VM instances**.
- Click **Create**.
- Specify the following, and leave the remaining settings as their defaults:

Property	**Value** (type value or select option as specified)
Name	**demoiam**
Zone	**us-central1-c**
Machine type	**micro (1 shared vCPU)**
Service account:	**read-bucket-objects**

- Click **Create**.

Task 7: Explore the Service Account User role

At this point, you might have the user test access by connecting via SSH to the VM and performing the next actions. As the owner of the project, you already possess the Service Account User role. So you can simulate what the user would experience by just using SSH to access the VM from the GCP Console.

The actions you perform and results will be the same as if you were the target user.

Use the Service Account User

- For **demoiam**, click **SSH** to launch a terminal and connect.
- Run the following command:

```
gcloud compute instances list
```

Result **(do not copy; this is example output)**:

```
ERROR: (gcloud.compute.instances.list) Some requests did not succeed:
 - Required 'compute.zones.list' permission for 'projects/qwiklabs-gcp'
```

What happened? Why?

- Copy the sample.txt file from the bucket you created earlier. Note that the trailing period is part of the command below. It means copy to "this location":

```
gsutil cp gs://[YOUR_BUCKET_NAME]/sample.txt .
```

Result **(do not copy; this is example output)**:

```
Copying gs://train-test-iam/sample.txt...
/ [1 files][   28.0 B/   28.0 B]
Operation completed over 1 objects/28.0 B.
```

- To rename the file you copied, run the following command:

```
mv sample.txt sample2.txt
```

- To copy the renamed file back to the bucket, run the following command:

```
gsutil cp sample2.txt gs://[YOUR_BUCKET_NAME]
```

Result **(do not copy; this is example output)**:

```
AccessDeniedException: 403 Caller does not have storage.objects.create access to
bucket train-test-iam.
```

What happened?

Because you connected via SSH to the instance, you can "act as the service account," essentially assuming the same permissions.The service account the instance was started with had the Storage Viewer role, which permits downloading objects from GCS buckets in the project.To list instances in a project, you need to grant the compute.instance.list permission. Because the service account did not have this permission, you could not list instances running in the project. Because the service account did have permission to download objects, it could download an object from the bucket. It did not have permission to write objects, so you got a "403 access denied" message.

Task 8: Review

In this lab you exercised granting and revoking Cloud IAM roles, first to a user, **Username 2**, and then to a Service Account User. You could allocate Service Account User credentials and "bake" them into a VM to create specific-purpose authorized bastion hosts.

Cleanup

- In the **Cloud Platform Console**, sign out of the Google account.
- Close the browser tab.

Last Updated: 2018-05-01

End your lab

Module 5: Labs and Demos

Cloud Storage

Overview

Cloud Storage is a fundamental resource in GCP, with many advanced features. In this lab, you exercise many Cloud Storage features that could be useful in your designs. You explore Cloud Storage using both the console and the gsutil tool.

Objectives

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

- Create and use buckets
- Set access control lists to restrict access
- Use your own encryption keys
- Implement version controls
- Use directory synchronization
- Share a bucket across projects using IAM

Task 1: Preparation

Create an IAM service account

- In the GCP Console, on the **Products & Services** menu (☰), click **IAM & admin** > **Service accounts**.
- Click **Create service account**.
- Specify the following, and leave the remaining settings as their defaults:

Property	**Value** (type value or select option as specified)
Service account name	**storecore**
Role	**Project** > **Editor**

- Enable **Furnish a new private key**, and select **JSON**.
- Click **Create**. A JSON key file will be downloaded. You will need to find this key file and upload it in into the VM in a later step.
- Click **Close**.
- On your hard drive, rename the JSON key file to **credentials.json**

Create a Cloud Storage bucket

- On the **Products & Services** menu, click **Storage** > **Browser**.

A bucket must have a globally unique name. You could use part of your PROJECT_ID_1 in the name to help make it unique. For example, if the PROJECT_ID_1 is "myproj-154920," your bucket name might be "storecore154920."

- Click **Create bucket**.
- Specify the following, and leave the remaining settings as their defaults:

Property	Value (type value or select option as specified)
Name	Enter a globally unique name
Default storage class	**Multi-Regional**

- Make a note of the bucket name. It will be used later in this lab and referred to as [BUCKET_NAME_1].
- Click **Create**.

Create a VM

- On the **Products & services** menu, click **Compute Engine** > **VM instances**.
- Click **Create**.
- Specify the following, and leave the remaining settings as their defaults:

Property	Value (type value or select option as specified)
Name	**storecore**
Zone	**us-central1-c**
Machine type	**n1-standard-1**

- Click **Create**.

Connect via SSH to the VM and authorize it to use the GCP API

In this lab, you change the configuration files for the Google Cloud SDK tools (gcloud and gsutil). Instead of making changes that could impact the functioning of Cloud Shell, you create a VM, authorize it to use the SDK, and make the changes to the configuration files on that VM. This will simplify cleanup, because all you will need to do is delete the VM.

- For **storecore**, click **SSH** to launch a terminal and connect.
- Store [BUCKET_NAME_1] in an environment variable:

```
export BUCKET_NAME_1=<enter bucket name 1 here>
```

- Find the downloaded JSON file from Task 1 on your computer. Ensure that you have changed its name to credentials.json.
- To upload credentials.json through the SSH VM terminal, click on the gear icon (⚙) in the upper-right corner, and then click **Upload file**.
- Select credentials.json and upload it.
- Click **Close** in the File Transfer window.
- To verify that the JSON file has been uploaded to the VM, run the following command:

```
ls
```

Result **(do not copy; this is example output)**:

```
credentials.json
```

- To authorize the **storecore** VM to use the Google Cloud API, run the following command:

```
gcloud auth activate-service-account --key-file credentials.json
```

The image you are using has the Google Cloud SDK pre-installed; therefore, you don't need to initialize the Google Cloud SDK. If you are attempting this lab in a different environment, make sure you have followed these procedures regarding installing the Google Cloud SDK:
https://cloud.google.com/sdk/downloads

Download a sample file using CURL and make two copies

-
- In the **storecore** SSH terminal, run the following command to download a sample file (this sample file is a publicly available Hadoop documentation HTML file):

```
curl \
http://hadoop.apache.org/docs/current/\
hadoop-project-dist/hadoop-common/\
ClusterSetup.html > setup.html
```

- To make copies of the file, run the following commands:

```
cp setup.html setup2.html
cp setup.html setup3.html
```

Task 2: Access control lists (ACLs)

Copy the file to the bucket and configure the access control list

- In the **storecore** SSH terminal, run the following command to copy the first file to the bucket:

```
gsutil cp setup.html gs://$BUCKET_NAME_1/
```

- To get the default access list that's been assigned to setup.html, run the following command:

```
gsutil acl get gs://$BUCKET_NAME_1/setup.html  > acl.txt
cat acl.txt
```

- To set the access list to private and verify the results, run the following commands:

```
gsutil acl set private gs://$BUCKET_NAME_1/setup.html
gsutil acl get gs://$BUCKET_NAME_1/setup.html  > acl2.txt
cat acl2.txt
```

- To update the access list to make the file publicly readable, run the following commands:

```
gsutil acl ch -u AllUsers:R gs://$BUCKET_NAME_1/setup.html
gsutil acl get gs://$BUCKET_NAME_1/setup.html  > acl3.txt
cat acl3.txt
```

Examine the file in the GCP Console

- In the GCP Console, on the **Products & Services** menu (☰), click **Storage** > **Browser**.
- Click [BUCKET_NAME_1].
- Verify that for file setup.html, **Share publicly** has a **Public link** available.

Delete the local file and copy back from Cloud Storage

- In the **storecore** SSH terminal, run the following command to delete the setup file:

```
rm setup.html
```

- To verify that the file has been deleted, run the following command:

```
ls
```

- To copy the file from the bucket again, run the following command:

```
gsutil cp gs://$BUCKET_NAME_1/setup.html setup.html
```

Task 3: Customer-supplied encryption keys (CSEK)

Generate a CSEK key

For the next step, you need an AES-256 base-64 key.

- In the **storecore** SSH terminal, run the following command to create a key:

```
python -c 'import base64; import os; print(base64.encodestring(os.urandom(32)))'
```

Result **(do not copy; this is example output)**:

```
python -c 'import base64; import os; print(base64.encodestring(os.urandom(32)))'
tmxElCaabWvJqR7uXEWQF39DhWTcDvChzuCmpHe6sb0=
```

- Copy the value of the key.

Modify the boto file

The encryption controls are contained in a gsutil configuration file named .boto.

- To view and open the boto file, run the following commands in the SSH terminal:

```
ls -al

nano .boto
```

If the .boto file is empty, close the nano editor with **Ctrl+X** and generate a new .boto file using the gsutil config -n command in the SSH terminal. Then, try opening the file again with the above commands.

If the .boto file is still empty, you might have to locate it using the gsutil version -l command in the SSH terminal.

- Locate the line with "#encryption_key="
- Uncomment the line by removing the # character, and paste the key you generated earlier at the end.

Example **(do not copy; this is an example)**:

```
Before:
# encryption_key=

After:
encryption_key=tmxElCaabWvJqR7uXEWQF39DhWTcDvChzuCmpHe6sb0=
```

- Press **Ctrl+O**, **ENTER** to save the boto file, and then press **Ctrl+X** to exit nano.

Upload the remaining setup files (encrypted) and verify in the GCP Console

- To upload the remaining setup.html files, run the following commands:

```
gsutil cp setup2.html gs://$BUCKET_NAME_1/
gsutil cp setup3.html gs://$BUCKET_NAME_1/
```

- Return to the GCP Console.
- Click [BUCKET_NAME_1]. Both setup2.html and setup3.html files show that they are customer-encrypted.

Delete local files, copy new files, and verify encryption

- To delete your local files, run the following command:

```
rm setup*
```

- To copy the files from the bucket again, run the following command:

```
gsutil cp gs://$BUCKET_NAME_1/setup* ./
```

- To cat the encrypted files to see whether they made it back, run the following commands:

```
cat setup.html
cat setup2.html
cat setup3.html
```

Task 4: Rotate CSEK keys

Move the current CSEK encrypt key to decrypt key

- In the **storecore** SSH terminal, run the following command to open the .boto file:

```
nano .boto
```

- Comment out the current encrypt_key line by adding the # character to the beginning of the line.
- Uncomment decrypt_key1 by removing the # character, and copy the current key from the encrypt_key line to the decrypt_key1 line.

Result **(do not copy; this is example output)**:

```
Before:
encryption_key=2dFWQGnKhjOcz4h0CudPdVHLG2g+OoxP8FQOIKKTzsg=

# decryption_key1=

After:
# encryption_key=2dFWQGnKhjOcz4h0CudPdVHLG2g+OoxP8FQOIKKTzsg=

decryption_key1=2dFWQGnKhjOcz4h0CudPdVHLG2g+OoxP8FQOIKKTzsg=
```

- Press **Ctrl+O**, **ENTER** to save the boto file, and then press **Ctrl+X** to exit nano.

Note: In practice, you would delete the old CSEK key from the encryption_key line.

Generate another CSEK key and add to the boto file

- In the **storecore** SSH terminal, run the following command to generate a new key:

```
python -c 'import base64; import os; print(base64.encodestring(os.urandom(32)))'
```

- Copy the value of the generated key.
- To open the boto file, run the following command:

```
nano .boto
```

- Add a new line with encryption_key= and paste the new key value.

Result **(do not copy; this is example output)**:

```
Before:
# encryption_key=2dFWQGnKhjOcz4h0CudPdVHLG2g+OoxP8FQOIKKTzsg=

After:
# encryption_key=2dFWQGnKhjOcz4h0CudPdVHLG2g+OoxP8FQOIKKTzsg=

encryption_key=HbFK4I8CaStcvKKIx6aNpdTse0kTsfZNUjFpM+YUEjY==
```

- Press **Ctrl+O**, **ENTER** to save the boto file, and then press **Ctrl+X** to exit nano.

Rewrite the key for file 1 and comment out the old decrypt key

When a file is encrypted, rewriting the file decrypts it using the decryption_key1 that you previously set, and encrypts the file with the new encryption_key.

You are rewriting the key for setup2.html, but not for setup3.html, so that you can see what happens if you don't rotate the keys properly.

- Run the following command:

```
gsutil rewrite -k gs://$BUCKET_NAME_1/setup2.html
```

- To open the boto file, run the following command:

```
nano .boto
```

- Comment out the current decryption_key line by adding the # character back in.

Result **(do not copy; this is example output)**:

```
Before:
decryption_key1=2dFWQGnKhjOcz4h0CudPdVHLG2g+OoxP8FQOIKKTzsg=

After:
# decryption_key1=2dFWQGnKhjOcz4h0CudPdVHLG2g+OoxP8FQOIKKTzsg=
```

- Press **Ctrl+O**, **ENTER** to save the boto file, and then press **Ctrl+X** to exit nano.

Note: In practice, you would delete the old CSEK key from the decryption_key1 line.

Download setup 2 and setup3

- To download setup2.html, run the following command:

```
gsutil cp  gs://$BUCKET_NAME_1/setup2.html recover2.html
```

- To download setup3.html, run the following command:

```
gsutil cp  gs://$BUCKET_NAME_1/setup3.html recover3.html
```

What happened? setup3.html was not rewritten with the new key, so it can no longer be decrypted, and the copy will fail.

You have successfully rotated the CSEK keys.

Task 5: Enable lifecycle management

View the current lifecycle policy for the bucket

- In the **storecore** SSH terminal, run the following command to view the current lifecycle policy:

```
gsutil lifecycle get gs://$BUCKET_NAME_1
```

There is no lifecycle configuration. You create one in the next steps.

Create a JSON lifecycle policy file

- To create a file named life.json, run the following command:

```
nano life.json
```

- Paste the following value into the life.json file:

```
{
  "rule":
  [
    {
      "action": {"type": "Delete"},
      "condition": {"age": 31}
    }
  ]
        }
```

These instructions tell Cloud Storage to delete the object after 31 days.

- Press **Ctrl+O**, **ENTER** to save the file, and then press **Ctrl+X** to exit nano.

Set the policy and verify

- To set the policy, run the following command:

```
gsutil lifecycle set life.json gs://$BUCKET_NAME_1
```

- To verify the policy, run the following command:

```
gsutil lifecycle get gs://$BUCKET_NAME_1
```

Task 6: Enable versioning

View the versioning status for the bucket and enable versioning

- In the **storecore** SSH terminal, run the following command to view the current versioning status for the bucket:

```
gsutil versioning get gs://$BUCKET_NAME_1
```

The Suspended policy means that it is not enabled.

- To enable versioning, run the following command:

```
gsutil versioning set on gs://$BUCKET_NAME_1
```

- To verify that versioning was enabled, run the following command:

```
gsutil versioning get gs://$BUCKET_NAME_1
```

Create several versions of the sample file in the bucket

- Check the size of the sample file:

```
ls -al setup.html
```

- Open the setup.html file:

```
nano setup.html
```

- Delete any 5 lines from setup.html to change the size of the file.
- Press **Ctrl+O**, **ENTER** to save the file, and then press **Ctrl+X** to exit nano.
- Copy the file to the bucket with the -v versioning option:

```
gsutil cp -v setup.html gs://$BUCKET_NAME_1
```

- Open the setup.html file:

```
nano setup.html
```

- Delete another 5 lines from setup.html to change the size of the file.
- Press **Ctrl+O**, **ENTER** to save the file, and then press **Ctrl+X** to exit nano.
- Copy the file to the bucket with the -v versioning option:

```
gsutil cp -v setup.html gs://$BUCKET_NAME_1
```

List all versions of the file

- To list all versions of the file, run the following command:

```
gsutil ls -a gs://$BUCKET_NAME_1/setup.html
```

- Highlight and copy the name of the oldest version of the file (the first listed), referred to as [VERSION_NAME] in the next step.
- Store the version value in the environment variable [VERSION_NAME].

```
export VERSION_NAME=<Enter VERSION name here>
```

Download the oldest, original version of the file and verify recovery

- Download the original version of the file:

```
gsutil cp $VERSION_NAME recovered.txt
```

- To verify recovery, run the following commands:

```
ls -al setup.html

ls -al recovered.txt
```

You have recovered the original file from the backup version. Notice that the original is bigger than the current version because you deleted lines.

Task 7: Synchronize a directory to a bucket

Make a nested directory and sync with a bucket

Make a nested directory structure so that you can examine what happens when it is recursively copied to a bucket.

1 Run the following commands:

```
mkdir firstlevel
mkdir ./firstlevel/secondlevel
cp setup.html firstlevel
cp setup.html firstlevel/secondlevel
```

2 To sync the home directory on the VM with your bucket, run the following command:

```
gsutil rsync -r ./firstlevel gs://$BUCKET_NAME_1/firstlevel
```

3 To verify that versioning was enabled, run the following command:

```
gsutil versioning get gs://$BUCKET_NAME_1
```

Examine the results

1 In the GCP Console, on the **Products & Services** menu (☰), click **Storage** > **Browser**.

2 Click [BUCKET_NAME_1]. Notice the subfolders in the bucket.

3 Click on **/firstlevel** and then on **/secondlevel**.

4 Compare what you see in the GCP Console with the results of the following command:

```
gsutil ls -r gs://$BUCKET_NAME_1/firstlevel
```

5 Exit the SSH terminal:

```
exit
```

Task 8: Cross-project sharing

Switch to the second project

1 Open a new incognito tab.

2 Navigate to **console.cloud.google.com**.

3 Click the project selector dropdown in the title bar.

4 Click **All**, and then click the second project provided for you in the Qwiklabs Connection Details dialog. Remember that the Project ID is a unique name across all Google Cloud projects. The second project ID will be referred to as [PROJECT_ID_2].

Prepare the bucket

1 In the GCP Console, on the **Products & Services** menu (☰), click **Storage** > **Browser**.

2 Click **Create bucket**.

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

Property	**Value** (type value or select option as specified)
Name	Enter a globally unique name
Default storage class	**Multi-Regional**

4 Note the bucket name. It will be referred to as [BUCKET_NAME_2] in the following

steps.

5 Click **Create**.

Upload a text file to the bucket

1 Upload a file to [BUCKET_NAME_2]. Any small example file or text file will do.

2 Note the file name (referred to as [FILE_NAME]); you will use it later.

Create an IAM Service Account

1 On the **Products & services** menu, click > **IAM & admin** > **Service accounts**.

2 Click **Create service account**.

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

Property	**Value** (type value or select option as specified)
Service account name	**cross-project-storage**
Role	**Storage** > **Storage Object Viewer**

4 Enable **Furnish a new private key**, and then click **JSON**.

5 Click **Create**. A JSON key file will be downloaded. You will need to find this key file and upload it in into the VM in a later step.

6 Rename the JSON key file on your local machine to **credentials.json**

You might have to delete the previously downloaded JSON key file to change the new file's name to credentials.json.

7 Click **Close**.

8 In the upper pane, switch back to [PROJECT_ID_1].

Create a VM

1 On the **Products & Services** menu, click **Compute Engine** > **VM instances**.

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	**crossproject**
Zone	**europe-west1-d**
Machine type	**micro (1 shared vCPU)**

4 Click **Create**.

SSH to the VM

1 For **crossproject**, click **SSH** to launch a terminal and connect.

2 Store [BUCKET_NAME_2] in an environment variable:

```
export BUCKET_NAME_2=<enter bucket name 2 here>
```

3 Store [FILE_NAME] in an environment variable:

```
export FILE_NAME=<enter FILE_NAME here>
```

4 List the files in [PROJECT_ID_2]:

```
gsutil ls gs://$BUCKET_NAME_2/
```

Result **(do not copy; this is example output)**:

```
AccessDeniedException: 403 Caller does not have storage.objects.list access to
bucket [BUCKET_NAME_2].
```

Authorize the VM

1 To upload credentials.json through the SSH VM terminal, click on the gear icon (⚙) in the upper-right corner, and then click **Upload file**.

2 Select credentials.json and upload it.

3 Click **Close** in the File Transfer window.

4 Verify that the JSON file has been uploaded to the VM:

```
ls
```

Result **(do not copy; this is example output)**:

```
credentials.json
```

5 Enter the following command in the terminal to authorize the VM to use the Google Cloud API:

```
gcloud auth activate-service-account --key-file credentials.json
```

The image you are using has the Google Cloud SDK pre-installed; therefore, you don't need to initialize the Google Cloud SDK. If you are attempting this lab in a different environment, make sure you have followed these procedures regarding installing the Google Cloud SDK:
https://cloud.google.com/sdk/downloads

Verify access

1 Retry this command:

```
gsutil ls gs://$BUCKET_NAME_2/
```

2 Retry this command:

```
gsutil cat gs://$BUCKET_NAME_2/$FILE_NAME
```

3 Try to copy the credentials file to the bucket:

```
gsutil cp credentials.json gs://$BUCKET_NAME_2/
```

Result **(do not copy; this is example output)**:

```
Copying file://credentials.json [Content-Type=application/json]...
AccessDeniedException: 403 Caller does not have storage.objects.create access to
bucket [BUCKET_NAME_2].
```

Modify role

1 In the upper pane, switch back to [PROJECT_ID_2].

2 In the GCP Console, on the **Products & Services** menu (☰), click **IAM & admin** > **IAM**.

3 Select the **cross-project-storage** service account.

4 Click **Role(s)**, and then click **Storage** > **Storage Object Admin**.

5 Click **Save**. If you don't click **Save**, the change will not be made. The service account **Roles** should now say **Multiple**.

Verify changed access

1 Return to the SSH terminal for **crossproject**.

2 Copy the credentials file to the bucket:

```
gsutil cp credentials.json gs://$BUCKET_NAME_2/
```

Result **(do not copy; this is example output)**:

```
Copying file://credentials.json [Content-Type=application/json]...
- [1 files][  2.3 KiB/  2.3 KiB]
Operation completed over 1 objects/2.3 KiB.
```

In this example the VM in PROJECT_ID_1 can now upload files to Cloud Storage in a bucket that was created in another project.

Note that the project where the bucket was created is the billing project for this activity. That means if the VM uploads a ton of files, it will not be billed to PROJECT_ID_1, but instead to PROJECT_ID_2.

Task 9: Review

In this lab you learned to create and work with buckets and objects, and you learned about the following features for Cloud Storage:

- CSEK: Customer-supplied encryption key
- Use your own encryption keys
- Rotate keys
- ACL: Access control list
- Set an ACL for private, and modify to public
- Lifecycle management
- Set policy to delete objects after 31 days
- Versioning
- Create a version and restore a previous version
- Directory synchronization
- Recursively synchronize a VM directory with a bucket
- Cross-project resource sharing using IAM
- Use IAM to enable access to resources across projects

Cleanup

1 In the **Cloud Platform Console**, sign out of the Google account.
2 Close the browser tab.

Last Updated: 2018-05-01

End your lab

Module 5: Labs and Demos

Overview

In this lab, you create a Cloud SQL instance and a client VM instance. You then configure encrypted access using SSL certificates

Objectives

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

- Create a Cloud SQL instance
- Create a VM to serve as a database client and install software
- Restrict access to the Cloud SQL instance to a single IP address
- Download sample GCP billing data in *.csv format and load that into the database
- Configure the Cloud SQL instance and the client to use SSL encryption

Task 1: Create a VM to serve as the database client

- In the GCP Console, on the **Products & Services** menu (☰), click **Compute Engine** > **VM instances**.
- Click **Create**.
- Specify the following, and leave the remaining settings as their defaults:

Property	**Value** (type value or select option as specified)
Name	**mydb-client**
Zone	Select a zone of your choice - referred to as YOUR_ZONE
Machine type	**1 vCPU (n1-standard-1)**

- Click **Create**.
- When the VM is operational, copy the **External IP** address to be used in the next section as **<IP address of VM>**.

Task 2: Create a Cloud SQL instance

- On the **Products & Services** menu, click **SQL**.
- Click **Create instance**.
- Click **MySQL**, and then click **Next**.
- Click **Choose Second Generation**.
- Specify the following, and leave the remaining settings as their defaults:

Property	Value (type value or select option as specified)
Instance ID	**infra-db**
Root password	<type a password you'll remember>
Region	Select a region of your choice, referred to as YOUR_REGION
Zone	YOUR_ZONE

Make sure you set the root password. You will be prompted for this password whenever you access the mysql client in the following steps.

- Expand **Show configuration options**.
- Expand **Choose database version**, and for **Database version**, select **MySQL 5.7**.
- Expand **Configure machine type and storage**, and leave the remaining settings as their defaults:

Property	Value (type value or select option as specified)
Storage type	**SSD (Recommended)**
Storage capacity	**10 GB**

- Expand **Authorized networks**, and then click **Add network**.
- Specify the following:

Property	Value (type value or select option as specified)
Name	**mydb-client**
Network	<ip address of vm>**/32**

The /32 in CIDR notation means "this exact address of a single VM." This is the external IP. There are several ways to connect to a Cloud SQL server.

- Click **Done**.
- Click **Create** to create the database instance. The database instance will take a while to launch. In the meantime, you can continue working on the next step.

Task 3: Install the MySQL database client

Install the mysql-client

- On the **Products & Services** menu, click **Compute Engine** > **VM instances**.
- For **mydb-client**, click **SSH**.
- Update the packages:

```
sudo apt-get update
```

- Install the mysql-client:

```
sudo apt-get install mysql-client
```

- Type **Y** when prompted to continue the installation.

Verify database access

- On the **Products & Services** menu, click **SQL**.
- When the **infra-db** instance is operational, copy the **IP address** to be used next as <IP of CloudSQL instance>.
- Return to the SSH window of the **mydb-client** VM.
- Store <IP of CloudSQL instance> in the environment variable **IP_ADDRESS_SQL**:

```
export IP_ADDRESS_SQL=<Enter IP address of CloudSQL instance here>
```

- Start the mysql-client:

```
mysql --host=$IP_ADDRESS_SQL -u root -p
```

- When prompted, enter the root password that you specified for the SQL instance.
- Verify that the Cloud SQL instance is working by listing all databases:

```
show databases;
```

- Exit the database server to return to the **mydb-client** VM:

```
exit;
```

Task 4: Populate the database

- In the SSH window of the **mydb-client** VM, download a sample Google Cloud billing data *.csv file:

```
curl -O https://storage.googleapis.com/cloud-training/archinfra/Example-billing-export.csv
```

- Return to the mysql client and run the following command:

```
mysql --host=$IP_ADDRESS_SQL -u root -p
```

- When prompted, enter the root password that you specified for the SQL instance.
- Create a database named **resources**:

```
create database resources;
use resources;
```

- Create the table named **billing** in the **resources** database:

```
CREATE TABLE billing
(
Account_ID varchar(255),
Line_Item varchar(255),
Start_Time varchar(255),
End_Time varchar(255),
Project varchar(255),
Measurement1 varchar(255),
Measurement1_Total_Consumption bigint(255),
Measurement1_Units varchar(255),
Cost varchar(255),
Currency int,
Project_Number varchar(255),
Project_ID varchar(255),
Project_Name varchar(255),
Project_Labels varchar(255),
Description varchar(255)
);
```

- Verify that the table was created properly:

```
show tables;
describe billing;
```

- Exit the database server to return to the **mydb-client** VM:

```
exit;
```

- You now use the command mysqlimport to import the example billing data that you downloaded into the table in the database you created. However, mysqlimport requires that the *.csv file have the same name as the table. In the SSH window of the **mydb-client** VM, change the name of the data file:

```
mv Example-billing-export.csv billing.csv
```

- Load the data into the table:

```
mysqlimport --host=$IP_ADDRESS_SQL --ignore-lines=1 --fields-terminated-by=,
resources --verbose --local -u root -p  billing.csv
```

- When prompted, enter the root password that you specified for the SQL instance.

- Return to the mysql client and run the following command:

```
mysql --host=$IP_ADDRESS_SQL -u root -p
```

- When prompted, enter the root password that you specified for the SQL instance.
- Verify that the data is now in the database:

```
use resources;
describe billing;
SELECT Measurement1, Cost from billing;
```

- Exit the database server to return to the **mydb-client** VM:

```
exit;
```

Task 5: Set up SSL encryption

The method you used to connect to the Cloud SQL database is fast to set up.

So far you have provided access management, in the form of a user ID and password. And you have provided authorization in the form of the single authorized IP address from the **mydb-client** VM.

But this setup does not afford security because the communications are being sent over an unencrypted connection over the internet.

View the details of your Cloud SQL database and modify encryption

- In the Console, navigate to **Products & services** > **SQL**.
- Click **infra-db**.
- Click **SSL**.
- For **SSL Connections**, click **Allow only SSL connections**. This updates the instance, which may take a few minutes.
- For **Client Certificates**, click **Create a Client Certificate**.
- On the popup menu, name the certificate **mydb-client**, and click **Create**. This updates the security settings on the instance. It then provides you with the security files in a dialog.
- Download the three files onto your computer:
- client-key.pem
- client-cert.pem
- server-ca.pem
- Click **Close**.

Alternatively, you can copy the contents of each file and recreate them on the **mydb-client** VM using the nano text editor.

Upload the client files

- Return to the SSH window of the **mydb-client** VM.
- Click on the gear icon (⚙) in the upper-right corner, and then click **Upload file**.
- Select the client-key.pem file and upload it.
- Repeat this process for the other two files:
- client-cert.pem
- server-ca.pem
- Click **Close**.
- Verify that all three files have been uploaded:

```
ls -l
```

Result **(do not copy; this is example output)**:

```
-rw-r--r-- 1 gcpstaging5359_student gcpstaging5359_student 14475 Oct  3 13:32 billing.csv
```

```
-rw-r--r-- 1 gcpstaging5359_student gcpstaging5359_student  1183 Oct  3 13:46 client-cert.pem
-rw-r--r-- 1 gcpstaging5359_student gcpstaging5359_student  1674 Oct  3 13:47 client-key.pem
-rw-r--r-- 1 gcpstaging5359_student gcpstaging5359_student  1146 Oct  3 13:50 server-ca.pem
```

Access the database over a secure link

- Run the following command:

```
mysql -uroot -p -h $IP_ADDRESS_SQL \
    --ssl-ca=server-ca.pem \
    --ssl-cert=client-cert.pem \
    --ssl-key=client-key.pem
```

- When prompted, enter the root password that you specified for the SQL instance.
- Verify the connection by listing all databases on the Cloud SQL instance:

```
show databases;
```

You successfully connected to the Cloud SQL instance using SSL encryption.
There are more methods to securely connect to a Cloud SQL instance.
You can read about them here:
https://cloud.google.com/sql/docs/mysql/connect-admin-ip

Task 6: Review

In this lab you configured Cloud SQL for use by a client. Then you improved security by requiring SSL certificates.

Cleanup

- In the **Cloud Platform Console**, sign out of the Google account.
- Close the browser tab.

Last Updated: 2018-05-01

End your lab

Cloud Datastore

Overview

In this lab, you create a Cloud Datastore database, run a query, and access the Cloud Datastore Admin console.

Objectives

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

- Initialize Cloud Datastore
- Create content in the database
- Query the content using both GQL and Kind queries
- Access the Cloud Datastore Admin console

Task 1: Create a Cloud Datastore database

Create an entity

- In the GCP Console, on the **Products & Services** menu (☰), click **Datastore** > **Entities**.
- Click **Create entity**.
- If asked for a region, select a region and click **Next**.
- For **Kind**, type **Task** and leave all other values with their defaults.
- Click **Add property**.
- Specify the following:

Property	**Value** (type value or select option as specified)
Name	**Description**
Type	**String**
Value	**Cloud Datastore Infrastructure**

- Disable **Index this property**.
- Click **Done**.
- Click **Add property**.
- For **Name**, type **created**, and for **Type**, select **Date and time**.
- Leave **Index this property** enabled.
- Click **Done**.
- Click **Add property**.
- Specify the following:

Property	**Value** (type value or select option as specified)
Name	**done**
Type	**Boolean**
Value	**False**

- Leave **Index this property** enabled.

- Click **Done**.
- Click **Create**.

You have just stored your first data in Datastore.

Task 2: Explore the query capability

Filter

- On the **Entities** page, click **Query by kind**.
- Click **Filter entities**.
- For **Key**, click **done**.
- Leave the remaining settings as their defaults, and click **Apply filters**. Observe the query results.
- For **that is false**, click **that is true**.
- Click **Apply filters**. Observe the query results.

Task 3: Create more entities

Create a second entity

- Click **Create entity**.
- For **Description**, click the pencil icon to edit the property.
- For **Value**, type **Second entity in the datastore**.
- Disable **Index this property**.
- Click **Done**.
- Leave the **done** property set to **false**, and click **Create**.

Create a third entity

- Click **Create entity**.
- For **Description**, click the pencil icon to edit the property.
- For **Value**, type **Third entity in the datastore**.
- Click **Done**.
- For **done**, click the pencil icon to edit the property.
- For **Value**, select **True**.
- Click **Done**.
- Click **Create**.

Task 4: Conduct more queries

Use Query by kind

- Click **Filter entities**.
- For **Key**, click **done**.
- Ensure that **that is false** is selected, and click **Apply filters**. Observe the results.
- Select **that is true** and click **Apply filters**. Observe the results.

Use Query by GQL

- Click **Query by GQL**.
- Enter the following in the query box:

```
SELECT * from Task
```

- Click **Run query**. Observe the results.
- Enter the following in the query box:

```
SELECT * from Task where done = false
```

- Click **Run query**. Observe the results.

Task 5: Explore Datastore Admin

- In the left pane, click **Admin**.

Datastore Admin enables you to halt changes to the database using **Disable writes** and gives you access to the full Administration console. Notice that Datastore Admin is disabled by default and you have to intentionally enable it to use it. This is done to prevent accidental bulk delete operations.

- Click **Enable Datastore Admin**.
- Click **Open Datastore Admin**. The Datastore Admin console opens in a new tab.
- Select the **Task** Entity Kind.
- Click **Delete Entities**.

Now you understand why Cloud Datastore Admin is disabled by default. If you create a Kind of Entities (a collection of data) for testing purposes and need to delete them from Cloud Datastore, this is where you queue them for deletion.

- You don't need to delete the Task data. Click **Cancel**.
- Close the Cloud Datastore Admin console and return to the GCP Console.
- Click **Disable Datastore Admin**.

Task 6: Review

In this lab you created a Cloud Datastore database. You populated the database with data entities and ran both **Query by kind** and **Query by GQL** queries. You then enabled the **Cloud Datastore Admin** console so that you could learn how to clean up and remove test data.

Cleanup

- In the **Cloud Platform Console**, sign out of the Google account.
- Close the browser tab.

Last Updated: 2018-03-27

End your lab