**Install a NGINX web server**

Once SSH'ed, get root access using sudo:

sudo su -

As the root user, update your OS:

apt-get update

Install NGINX:

apt-get install nginx -y

Check that NGINX is running:

ps auwx | grep nginx

**Create a new instance with gcloud**

gcloud compute instances create gcelab2 --machine-type n1-standard-2 --zone us-central1-c

gcloud compute ssh gcelab2 --zone us-central1-c

Do you want to continue? (Y/n) Y

**Create VM in cloud shell**

gcloud config set compute/zone us-central1-b

gcloud compute instances create "my-vm-2" \

--machine-type "n1-standard-1" \

--image-project "debian-cloud" \

--image "debian-9-stretch-v20190213" \

--subnet "default"

**install the Nginx web server**

sudo apt-get install nginx-light -y

sudo nano /var/www/html/index.nginx-debian.html

**Create the Cloud SQL instance**

Click Create instance.

For Choose a database engine, select MySQL.

For Instance ID, type blog-db, and for Root password type a password of your choice.

Click Create

From the SQL instances details page, copy the Public IP address for your SQL instance to a text editor for use later in this lab.

Click on Users menu on the left-hand side, and then click ADD USER ACCOUNT.

For User name, type blogdbuser

For Password, type a password of your choice. Make a note of it.

Click Create to create the user account in the database.

Click the Connections tab, and then click Add network.

**Getting Started with GKE**

Start a Kubernetes Engine cluster

export MY\_ZONE=us-central1-a

gcloud container clusters create webfrontend --zone $MY\_ZONE --num-nodes 2

Run and deploy a container

kubectl create deploy nginx --image=nginx:1.17.10

kubectl get pods

kubectl expose deployment nginx --port 80 --type LoadBalancer

kubectl get services

kubectl scale deployment nginx --replicas 3

**Getting Started with App Engine**

git clone https://github.com/GoogleCloudPlatform/python-docs-samples

Navigate to the source directory:

cd python-docs-samples/appengine/standard\_python3/hello\_world

sudo apt-get update

sudo apt-get install virtualenv

virtualenv -p python3 venv

source venv/bin/activate

pip install -r requirements.txt

python main.py #run the app

Deploy and run Hello World on App Engine

cd ~/python-docs-samples/appengine/standard\_python3/hello\_world

gcloud app deploy

gcloud app browse

**Getting Started with Deployment Manager**

export MY\_ZONE=us-central1-a

export PROJECT\_ID=

gsutil cp gs://cloud-training/gcpfcoreinfra/mydeploy.yaml mydeploy.yaml

sed -i -e "s/ZONE/$MY\_ZONE/" mydeploy.yaml

cat mydeploy.yaml

gcloud deployment-manager deployments create my-first-depl --config mydeploy.yaml

nano mydeploy.yaml

value: "apt-get update; apt-get install nginx-light -y"

gcloud deployment-manager deployments update my-first-depl --config mydeploy.yaml

**VPC Networking**

Create an auto mode network

On the Navigation menu (Navigation menu), click VPC network > VPC networks.

Click Create VPC network.

For Name, type mynetwork

For Subnet creation mode, click Automatic. Auto mode networks create subnets in each region automatically.

For Firewall rules, select all available rules.

Create custom mode networks

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

Click Create VPC Network.

For Name, type managementnet

For Subnet creation mode, click Custom.

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

Property Value (type value or select option as specified)

Name managementsubnet-us

Region us-central1

IP address range 10.130.0.0/20

Create the privatenet network

gcloud compute networks create privatenet --subnet-mode=custom

gcloud compute networks subnets create privatesubnet-us --network=privatenet --region=us-central1 --range=172.16.0.0/24

Create the firewall rules for managementnet

In the Cloud Console, on the Navigation menu (Navigation menu), click VPC network > Firewall.

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 managementnet-allow-icmp-ssh-rdp

Network managementnet

Targets All instances in the network

Source filter IP Ranges

Source IP ranges 0.0.0.0/0

Protocols and ports Specified protocols and ports

**Implement Private Google Access and Cloud NAT**

Create a VPC network and firewall rules

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

Click Create VPC network.

For Name, type privatenet.

For Subnet creation mode, click Custom.

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

Property Value (type value or select option as specified)

Name privatenet-us

Region us-central1

IP address range 10.130.0.0/20

In the left pane, click Firewall.

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 privatenet-allow-ssh

Network privatenet

Targets All instances in the network

Source filter IP ranges

Source IP ranges 35.235.240.0/20

Protocols and ports Specified protocols and ports

For tcp, click the checkbox and specify port 22.

Create the VM instance with no public IP address

Click Networking.

For Network interfaces, click the pencil icon to edit.

SSH to vm-internal to test the IAP tunnel

gcloud compute ssh vm-internal --zone us-central1-c --tunnel-through-iap

Enable Private Google Access

In the Cloud Console, on the Navigation menu (Navigation 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

Location type Multi-region

Configure a Cloud NAT gateway

In the Cloud Console, on the Navigation menu (Navigation menu), click Network services > Cloud NAT.

Click Get started to configure a NAT gateway.

Specify the following:

Property Value (type value or select option as specified)

Gateway name nat-config

VPC network privatenet

Region us-central1

For Cloud Router, select Create new router.

For Name, type nat-router

**Working with Virtual Machines**

In the Cloud Console, on the Navigation menu (Navigation menu), click Compute Engine > VM instances.

Click Create.

Click Management, security, disks, networking, sole tenancy.

Click Disks. You will add a disk to be used for game storage.

Click Add new disk.

Property Value (type value or select option as specified)

Name minecraft-disk

Disk type SSD Persistent Disk

Source type None (blank disk)

Size (GB) 50

Encryption Google-managed key

Click Networking.

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

Property Value (type value or select option as specified)

Network tags minecraft-server

Network interfaces Click default to edit the interface

External IP Create IP Address

Name mc-server-ip

Click Reserve.

Click Create.

Create a directory and format and mount the disk

For mc-server, click SSH to open a terminal and connect.

sudo mkdir -p /home/minecraft

sudo mkfs.ext4 -F -E lazy\_itable\_init=0,\

lazy\_journal\_init=0,discard \

/dev/disk/by-id/google-minecraft-disk

sudo mount -o discard,defaults /dev/disk/by-id/google-minecraft-disk /home/minecraft

Install and run the application

sudo apt-get update

sudo apt-get install -y default-jre-headless

cd /home/minecraft

sudo apt-get install wget

sudo wget https://launcher.mojang.com/v1/objects/d0d0fe2b1dc6ab4c65554cb734270872b72dadd6/server.jar

Initialize the Minecraft server

sudo java -Xmx1024M -Xms1024M -jar server.jar nogui

sudo ls -l

sudo nano eula.txt

Change the last line of the file from eula=false to eula=true

sudo apt-get install -y screen

sudo screen -S mcs java -Xmx1024M -Xms1024M -jar server.jar nogui

sudo screen -r mcs

Allow client traffic

In the Cloud Console, on the Navigation menu (Navigation menu), click VPC network > Firewall.

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 minecraft-rule

Target Specified target tags

Target tags minecraft-server

Source filter IP ranges

Source IP ranges 0.0.0.0/0

Protocols and ports Specified protocols and ports

For tcp, specify port 25565.

**Implementing Cloud SQL**

On the Navigation menu (Navigation menu), click SQL.

Click Create instance.

Click Choose MySQL.

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

Property Value

Instance ID wordpress-db

Root password type a password

Region us-central1

Zone Any

Database Version MySQL 5.7

Expand Show configuration options.

Expand the Connectivity section.

Select Private IP.

In the dialog box, click Enable API, click Allocate and connect, and then click Close. This enables Private Services Access and attaches a Private IP address to your Cloud SQL server.

Configure a proxy on a virtual machine

On the Navigation menu (Navigation menu) click Compute Engine.

Click SSH next to wordpress-europe-proxy.

Download the Cloud SQL Proxy and make it executable:

wget https://dl.google.com/cloudsql/cloud\_sql\_proxy.linux.amd64 -O cloud\_sql\_proxy && chmod +x cloud\_sql\_proxy

On the Navigation menu (Navigation menu), click SQL.

Click on the wordpress-db instance and wait for a green checkmark next to its name, which indicates that it is operational (this could take a couple of minutes).

Note the Instance connection name; it will be used later and referred to as [SQL\_CONNECTION\_NAME].

In addition, for the application to work, you need to create a table. Click Databases.

Click Create database, type wordpress, which is the name the application expects, and then click Create.

Return to the SSH window and save the connection name in an environment variable, replacing [SQL\_CONNECTION\_NAME] with the unique name you copied in a previous step.

export SQL\_CONNECTION=[SQL\_CONNECTION\_NAME]

To activate the proxy connection to your Cloud SQL database and send the process to the background, run the following command:

./cloud\_sql\_proxy -instances=$SQL\_CONNECTION=tcp:3306 &

Press ENTER

Connect an application to the Cloud SQL instance

curl -H "Metadata-Flavor: Google" http://169.254.169.254/computeMetadata/v1/instance/network-interfaces/0/access-configs/0/external-ip && echo

Go to the wordpress-europe-proxy external IP address in your browser and configure the Wordpress application.

Click Let's Go.

Specify the following, replacing [ROOT\_PASSWORD] with the password you configured upon machine creation, and leave the remaining settings as their defaults:

Property Value

Username root

Password [ROOT\_PASSWORD]

Database Host 127.0.0.1

Click Submit.

When a connection has been made, click Run the installation