# **Migrate for Compute Engine**

# **Overview**

In this lab you use Migrate for Compute Engine to migrate a VM instance (EC2) that exists on AWS to Google Cloud. This will be a "lift and shift" operation. When completed, the VM instance that was running on AWS will be running on Google Cloud.

**Build a Virtual Private Network between AWS and Google Cloud**

Migrate for Compute Engine requires a Virtual Private Network (VPN) between the Google Cloud environment and the environment from which you are sourcing the VM. There can be many steps involved in creating a VPN between AWS and Google Cloud which involve the exchange of IP addresses, keys, and many other definitions. Rather than configuring these by hand and introducing all the associated opportunities for errors, this lab provides you with a Terraform script. Terraform is an infrastructure as code tool used to provision environments. Terraform supports both Google Cloud and AWS and can configure both environments against each other. When you run the Terraform script, it will perform the following tasks:

* Create an AWS EC2 instance
* Create a Google Cloud Compute Engine
* Create an AWS network
* Create a custom VPC in Google Cloud
* Create the AWS side of the VPN connection
* Create the Google Cloud side of the VPN connection
* Enable appropriate firewall rules for the Google Cloud VPC network

**Setup**

**Before you click the Start Lab button**

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

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

**What you need**

To complete this lab, you need:

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

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

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

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

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



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



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

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

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

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

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

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

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

Activate Cloud Shell

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

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



Click **Continue**.



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



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

You can list the active account name with this command:

gcloud auth list

Copied!

content\_copy

(Output)

Credentialed accounts:

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

(Example output)

Credentialed accounts:

- google1623327\_student@qwiklabs.net

You can list the project ID with this command:

gcloud config list project

Copied!

content\_copy

(Output)

[core]

project = <project\_ID>

(Example output)

[core]

project = qwiklabs-gcp-44776a13dea667a6

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

Make sure you are logged out of your personal or corporate gmail.

Prepare your working environment

Set your Project ID:

gcloud config set project $DEVSHELL\_PROJECT\_ID

Copied!

content\_copy

In Cloud Shell, retrieve the Terraform scripts by copying them and unzipping the files:

gsutil cp gs://spls/gsp677/autonetdeploy-multicloudvpn.tar .

tar xvf autonetdeploy-multicloudvpn.tar

Copied!

content\_copy

Change into the directory with the scripts:

cd autonetdeploy-multicloudvpn

Copied!

content\_copy

Create Google Cloud Access Credentials

Open [Create Service Account key page](https://console.cloud.google.com/apis/credentials/serviceaccountkey?_ga=2.115587654.1139958982.1599500490-1814939485.1591127035) in a new tab.

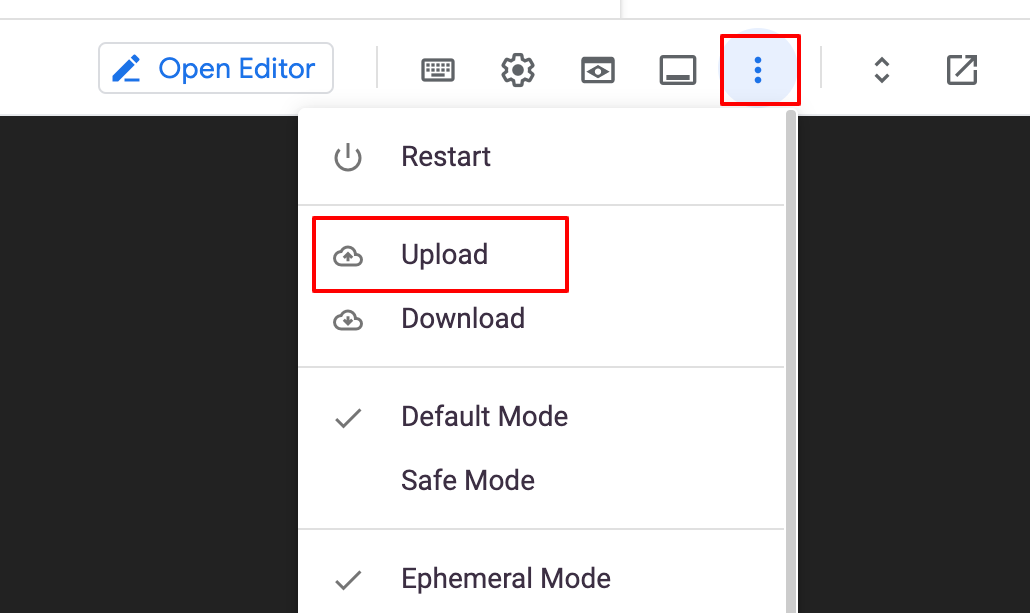
In the **Service Accounts** page, select your Project ID.

Select three dots unders Actions column of **Qwiklabs User Service Account** and select **Manage keys**.

Click **ADD KEY** and select **Create new key**. Select **JSON** as the Key type and click **CREATE**.

This will automatically download a JSON file of the key onto your local machine. Close the key window.

Move this file from your local machine by clicking the **three vertical dots** icon in the Cloud Shell ribbon, then select **Upload**.



Navigate to the JSON file you downloaded and click **Open**. The file is placed in the home (~) directory.

Next, use the ./gcp\_set\_credentials.sh script provided to create the ~/.config/gcloud/credentials\_autonetdeploy.json file. This script also creates terraform/terraform.tfvars with a reference to the new credentials.

Run the following, replacing [YOUR-CREDENTIALS]with the name of the JSON file you just downloaded:

./gcp\_set\_credentials.sh ~/[YOUR-CREDENTIALS].json

Copied!

content\_copy

Set AWS access credentials

For this lab, you will use the AWS access credentials that are created for the lab environment. These credentials are split into an **Access Key** and a **Secret Access Key** which are both displayed in the resources panel for this lab.

Run this command, replacing [YOUR\_ACCESS\_KEY] with your **AWS Access Key** from your resources panel, to store your **Access Key** in an environment variable:

export AWS\_ACCESS\_KEY=[YOUR\_ACCESS\_KEY]

Copied!

content\_copy

Run this command, replacing [YOUR\_SECRET\_KEY] with your **AWS Secret Key** from your resources panel, to store your **Secret Key** in an environment variable:

export AWS\_SECRET\_KEY=[YOUR\_SECRET\_KEY]

Copied!

content\_copy

To set your credentials, run the following in Cloud Shell:

./aws\_set\_credentials.sh $AWS\_ACCESS\_KEY $AWS\_SECRET\_KEY

Copied!

content\_copy

Run the following script to update the project value in your configuration files for Deployment Manager and Terraform.

./gcp\_set\_project.sh

Copied!

content\_copy

Generate Key-Pairs

Next, use ssh-keygen to generate a new key pair with your current account. For this lab, it's okay to use an empty passphrase.

ssh-keygen -t rsa -f ~/.ssh/vm-ssh-key -C $(whoami)

Copied!

content\_copy

When asked for passphrasee, press **Enter** twice to leave it blank.

Run the following to restrict access to your private key. This is a best practice.

chmod 400 ~/.ssh/vm-ssh-key

Copied!

content\_copy

Import Key to Google Cloud

In Cloud Shell, register your public key with Google Cloud:

gcloud compute config-ssh --ssh-key-file=~/.ssh/vm-ssh-key

Copied!

content\_copy

You will see this statement which is ok to ignore:

WARNING: No host aliases were added to your SSH configs because you do not have any running instances. Try running this command again after running some instances.

Copied!

content\_copy

Download Key

Get the filepath to your public key:

readlink -f ~/.ssh/vm-ssh-key.pub

Copied!

content\_copy

**Copy the file path** from the output.

To download the public key file from Cloud Shell, click the **three vertical dots**, and then click **Download**.

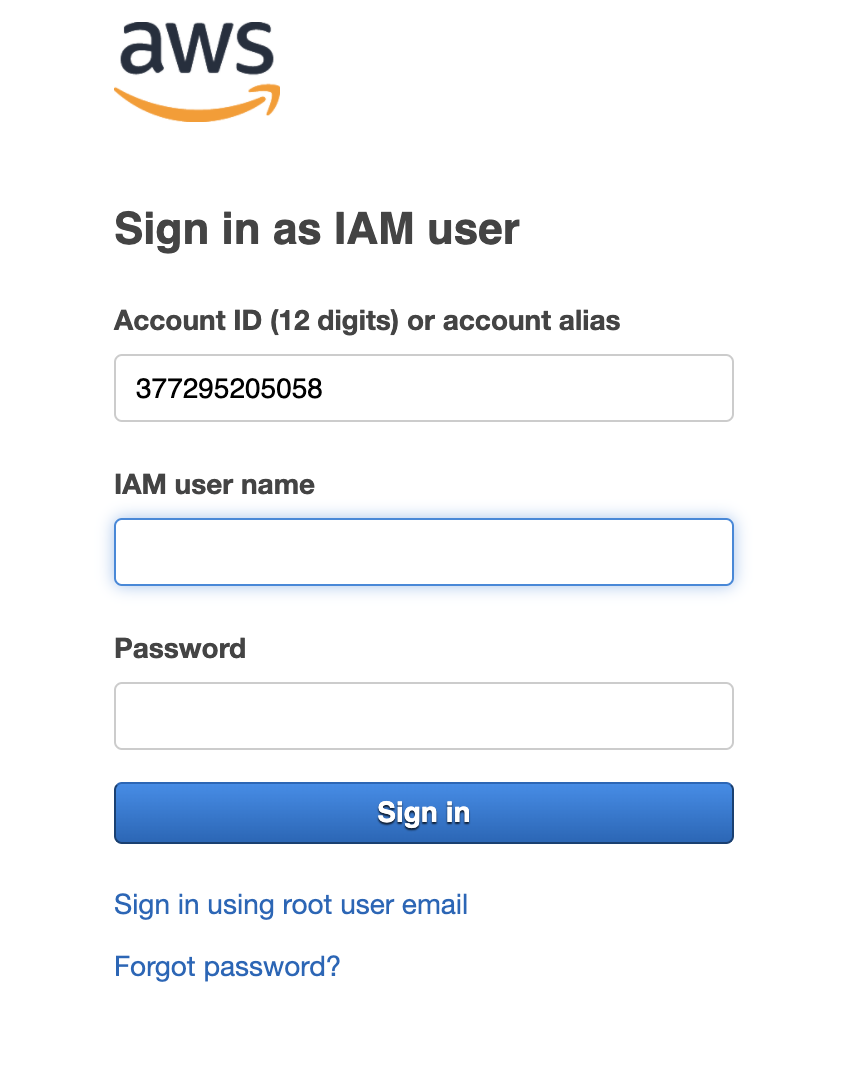
Now, paste in the file path to the key copied from the readlink command.

Click **Download**.

This will download the key file to your local machine.

Import Key to AWS

Click the **Open AWS Console** button listed in the panel of resources for this lab. This will take you to the AWS console login page:



At the login page, enter your **AWS Username** value for the **IAM user name** and enter your **AWS Password** value for the **Password**. Both of these can be found in the resources panel at the top left part of this page.

Click **Sign In**.

In the AWS console, look in the upper right corner next to your user name. If you are not signed in to the **N. Virginia** region, use the dropdown menu to select it.

aws-region

In the AWS Management Console, click on **All Services** > **EC2**. Then click on **Key pairs** located in the **Resources** panel.

In the top right corner, cllick **Actions** > **Import Key Pair**.

Name the key pair *vm-ssh-key*.

Click **Browse** and navigate to where the downloaded key pair file is.

Select **vm-ssh-key.pub** and click **Open**. The key pair will be added to your Import Settings page.

Click **Import key pair**.

Deploy with Terraform

Back in Cloud Shell, navigate to the autonetdeploy-multicloudvpn directory:

cd ~/autonetdeploy-multicloudvpn/

Copied!

content\_copy

Run the one-time terraform init command to install the Terraform providers for this deployment:

pushd ./terraform && terraform init && popd > /dev/null

Copied!

content\_copy

Run the terraform plan command to verify your credentials:

pushd ./terraform && terraform plan && popd > /dev/null

Copied!

content\_copy

If you don't see red error text, your authentication is working properly.

Now, navigate to the terraform directory:

pushd terraform

Copied!

content\_copy

Use the terraform validate command to validate the syntax of your configuration files. This validation check is simpler than those performed as part of the plan and apply commands in subsequent steps. The validate command does not authenticate with any providers.

terraform validate

Copied!

content\_copy

Use the terraform apply command to create your deployment:

terraform apply -auto-approve

Copied!

content\_copy

this will take about 10 minutes to complete.

What is happening: you are creating a Virtual Private Network (VPN) between Google Cloud and AWS. This requires resource definitions on both AWS and Google Cloud to be created that refer to each other. Items include IP addresses, routing information, shared keys and much more.

Once complete, you have both an AWS environment and a Google Cloud environment configured. An EC2 (VM) instance has also been created on AWS. This will be the VM instance that you will be migrating to Google Cloud.

Next, customize the EC2 instance so that you can easily see that, once migrated, it is the same VM on Google Cloud that was on AWS.

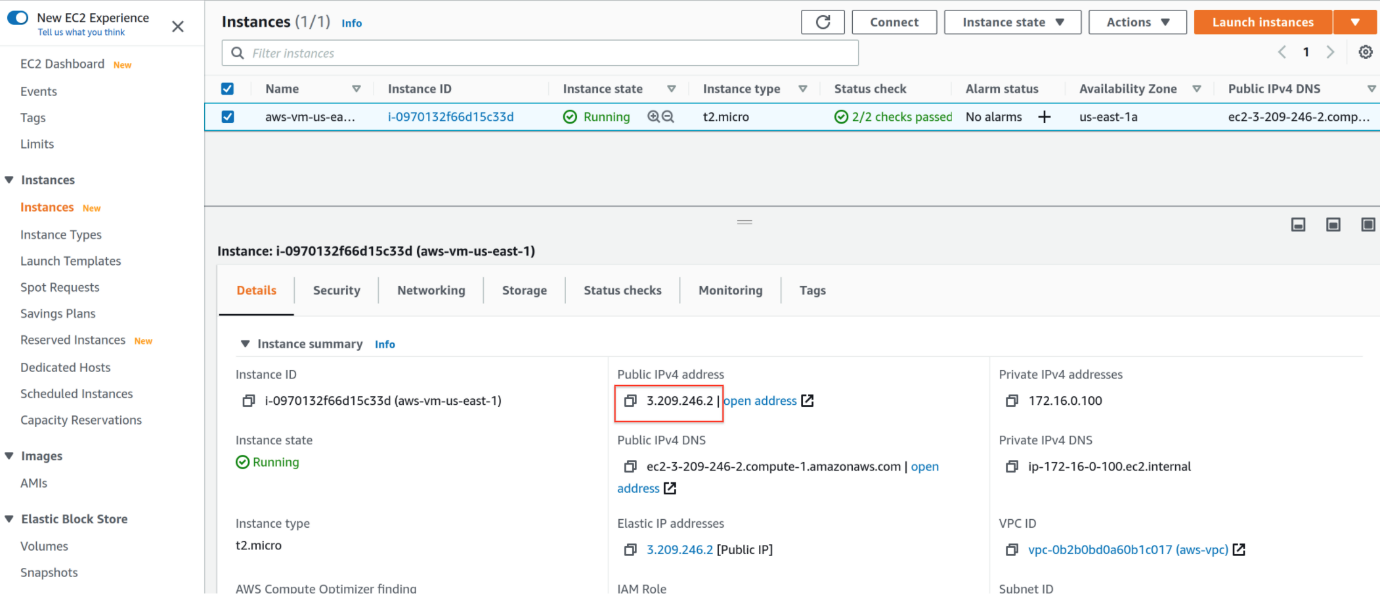
Click *Check my progress* to verify the objective.

Build a Virtual Private Network between AWS and Google Cloud

Check my progress

**Configure AWS EC2 instance for Migration**

On the AWS tab, visit the EC2 running instance by selecting **Instances** in the left hand navigation menu. Select the instance and find the public IP address.:



You will use your public key to ssh into your AWS instance.

Run the following, replacing **AWS\_INSTANCE\_EXTERNAL\_IP** with your instance's public IP:

ssh -i ~/.ssh/vm-ssh-key ubuntu@[AWS\_INSTANCE\_EXTERNAL\_IP]

Copied!

content\_copy

You will get a message asking to confirm the authenticity of the host. Type yes.

Once logged in, run the following commands:

sudo bash -c "apt-get update"

sudo bash -c "apt-get install apache2 -y"

Copied!

content\_copy

echo "Hello World" > MyText.txt

Copied!

content\_copy

When the Linux image runs on Google Cloud, it expects to find kernel drivers for the Migrate for Compute Engine mapped disks. These must be downloaded and installed prior to the migration up to Google Cloud. The driver installation must be performed upon the EC2 machine:

curl -LO https://storage.googleapis.com/velostrata-release/4.5/4.5.1/velostrata-prep-0.9-3.deb

Copied!

content\_copy

sudo dpkg -i velostrata-prep-0.9-3.deb

sudo apt-get update && sudo apt-get install -f -y

Copied!

content\_copy

You have now completed setting up the EC2 instance. Logout of the AWS VM instance by typing exit.

**Set up Migration Service Accounts**

In Cloud Shell, go back to the home directory and run this script to create the service accounts you will assign to Migrate for Compute Engine:

cd ~/autonetdeploy-multicloudvpn/

sh migrate\_sa\_roles.sh

Copied!

content\_copy

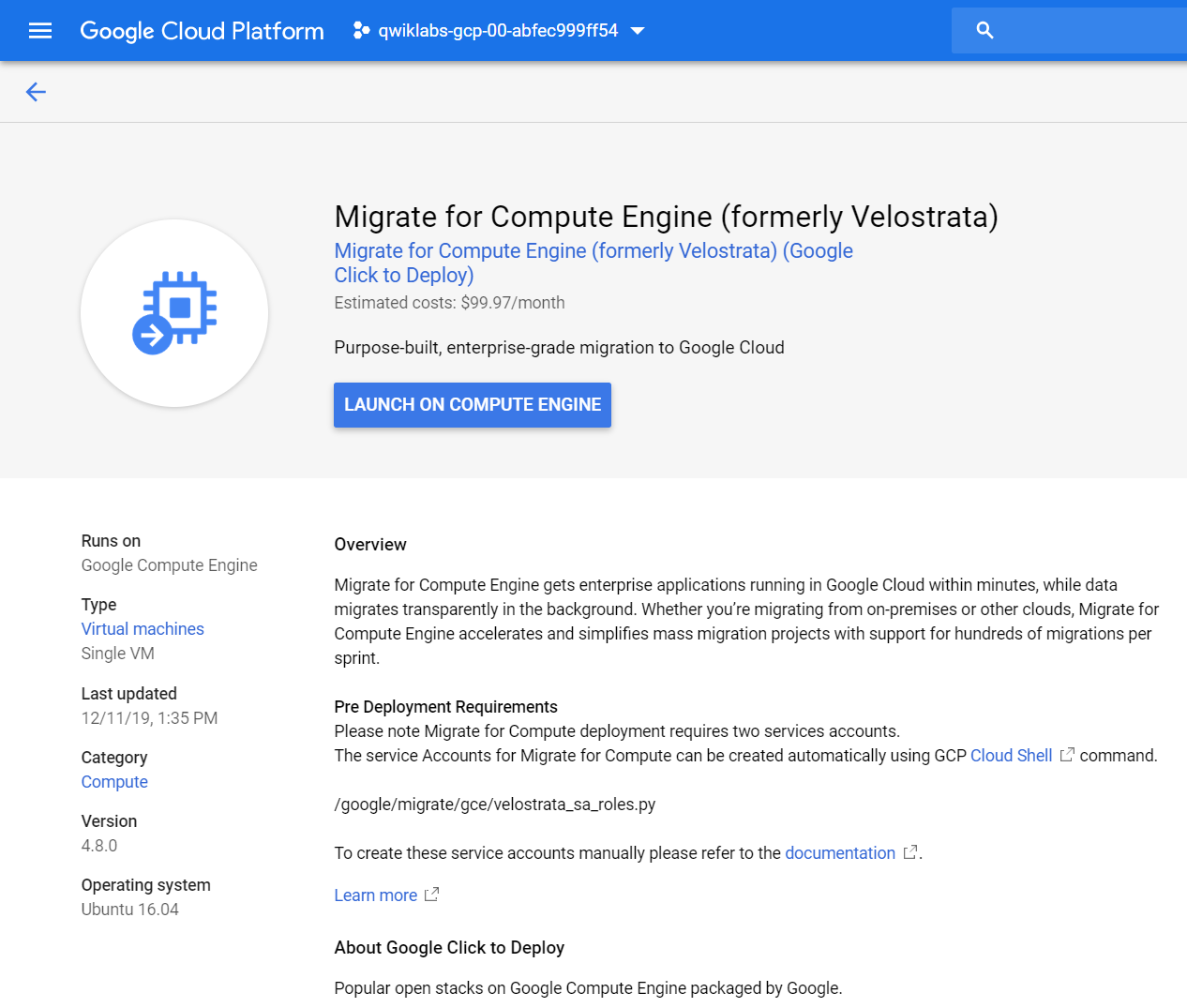
Click *Check my progress* to verify the objective.

Set up Migration Service Accounts

Check my progress

**Set up Migration Manager**

At this point you would *normally* go to the Console and Install Migrate for Compute Engine from the marketplace using the following two screens. This lab uses a special version of Migrate for Compute Engine which has lower resource needs and will run faster.



**Note**: The above is what should normally happen but since we are running in a QwikLab environment, we have an issue.

A script has been provided that will create a Compute Engine instance that works in QwikLabs. We can run this using:

bash create\_instance.sh

Copied!

content\_copy

You can ignore the warning:

WARNING: You have selected a disk size of under [200GB]. This may result in poor I/O performance. For more information, see: https://developers.google.com/compute/docs/disks#performance.

Copied!

content\_copy

This will take a few minutes to run.

Once finished, take note of the **External IP** field in the details of the velo-mgr vm. You will use it in the following command.

Click *Check my progress* to verify the objective.

Set up Migration Manager

Check my progress

**Migrate**

1. Test your velo-mgr vm has completed its start up and can be connected to. After creation, the Velostrata Manager often takes a minute or two to be launched and connectable.

In your Cloud Shell, run this command until you successfully get a response replacing [VELO-EXTERNAL-IP] with your velo-mgr vm's External IP:

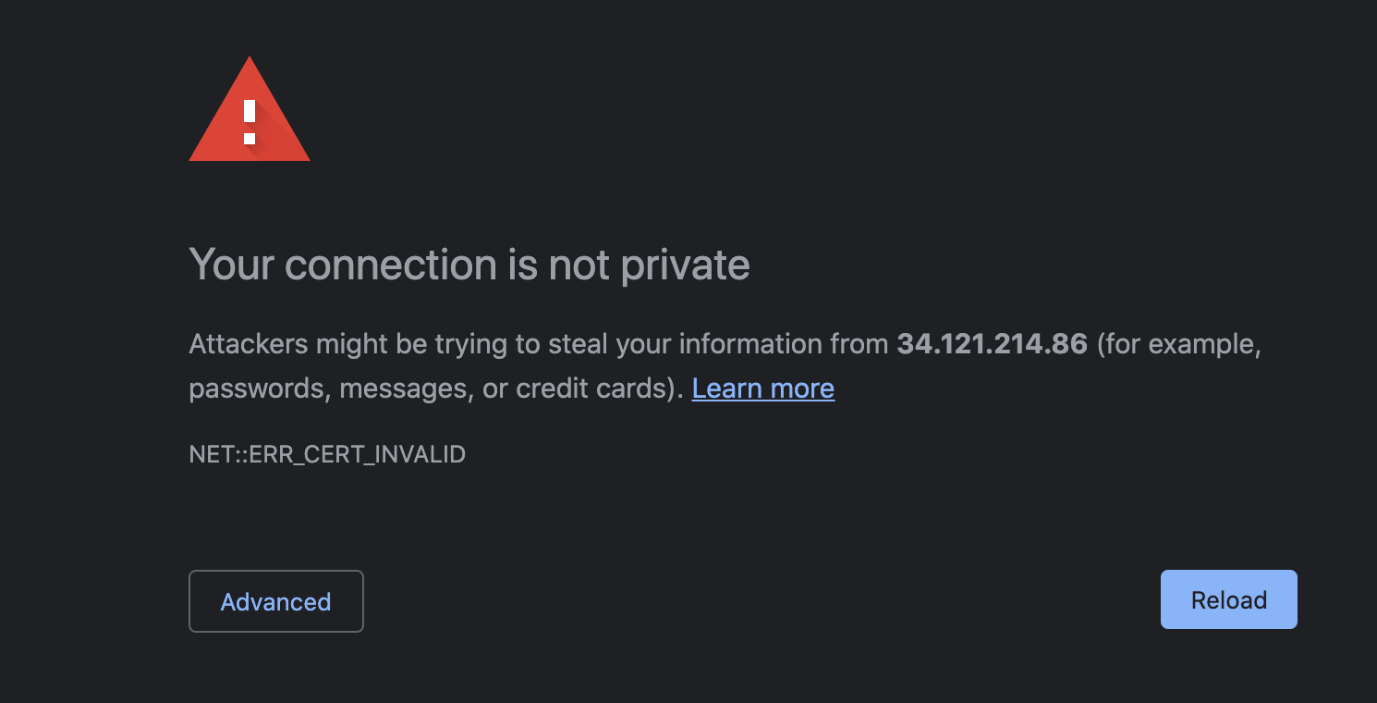
curl -k https://[VELO-EXTERNAL-IP]

Copied!

content\_copy

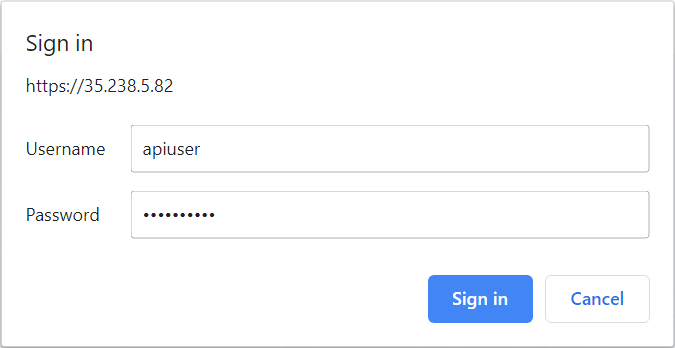
1. Access the Migrate for Compute Engine manager by copying the velo-mgr External IP and pasting it in a new tab in your browser.

Migrate for Compute Engine uses a self signed SSL certificate and your browser will most likely attempt to block your manager. You can get past this on Chrome by clicking **Advanced** > **Proceed Anyway** on the warning page that comes up when your VM finishes loading.

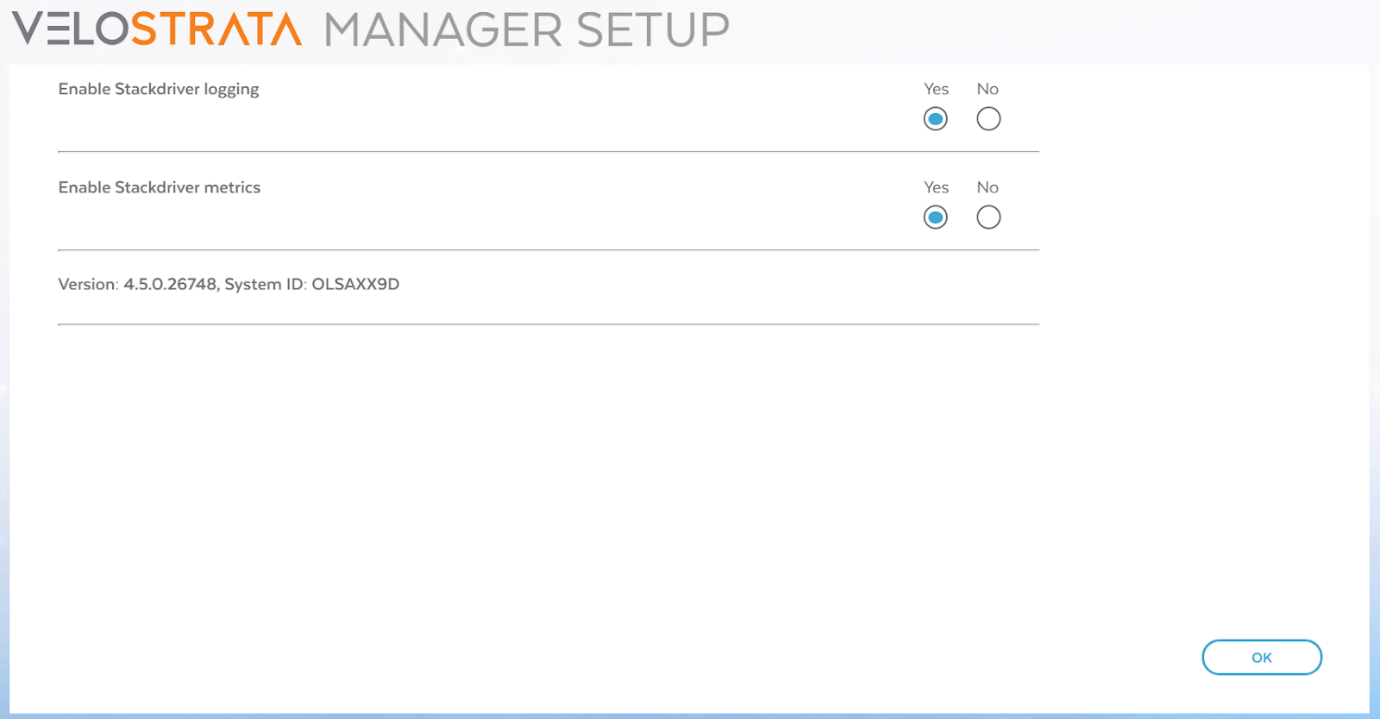


1. Log in with the following information:

* username: apiuser
* password: velo1234

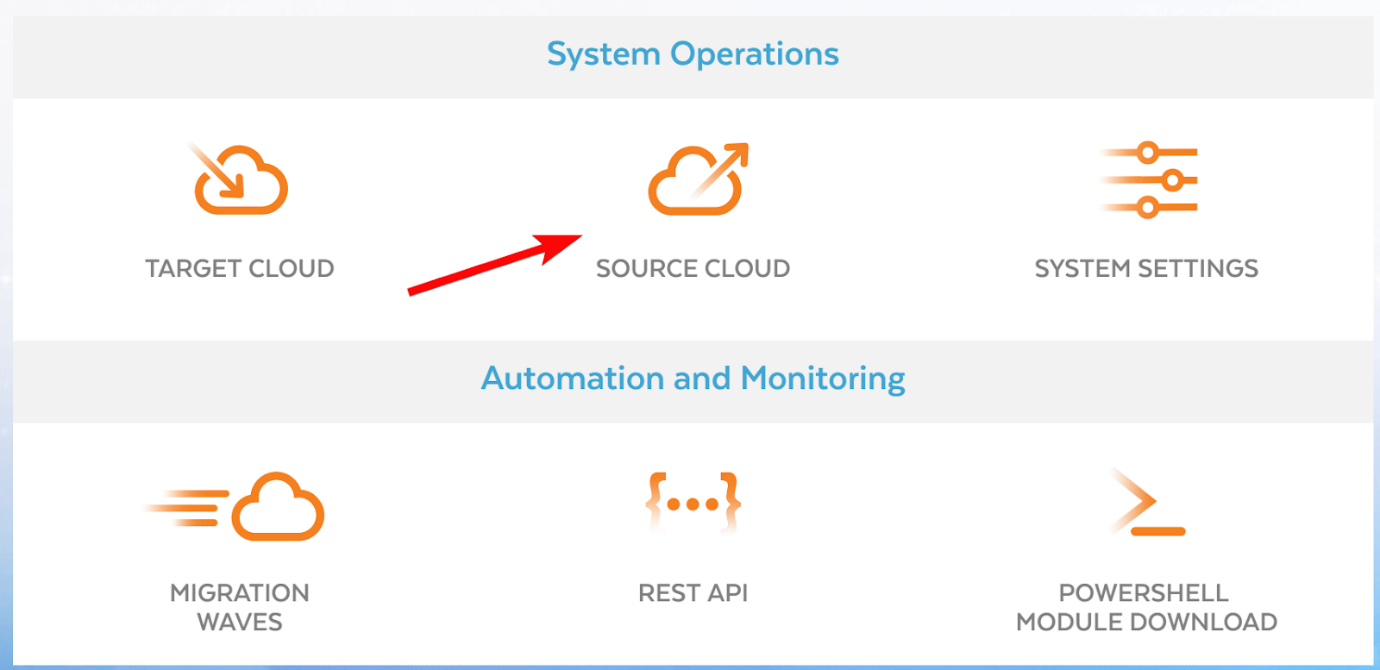


1. In the initial setup screen, enable Stackdriver for both Logging and Metrics. Click **OK**.

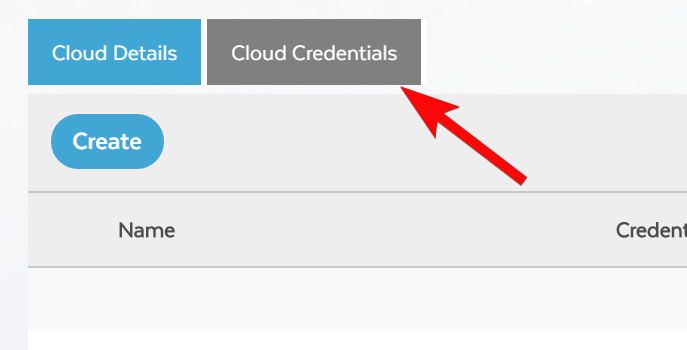


First, tell Migrate for Compute Engine about the AWS VM, which is your source.

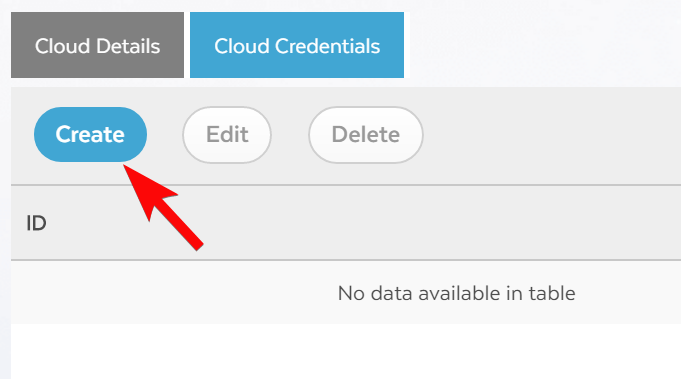
1. Click the **Source Cloud** icon:



Click the Cloud Credentials tab:



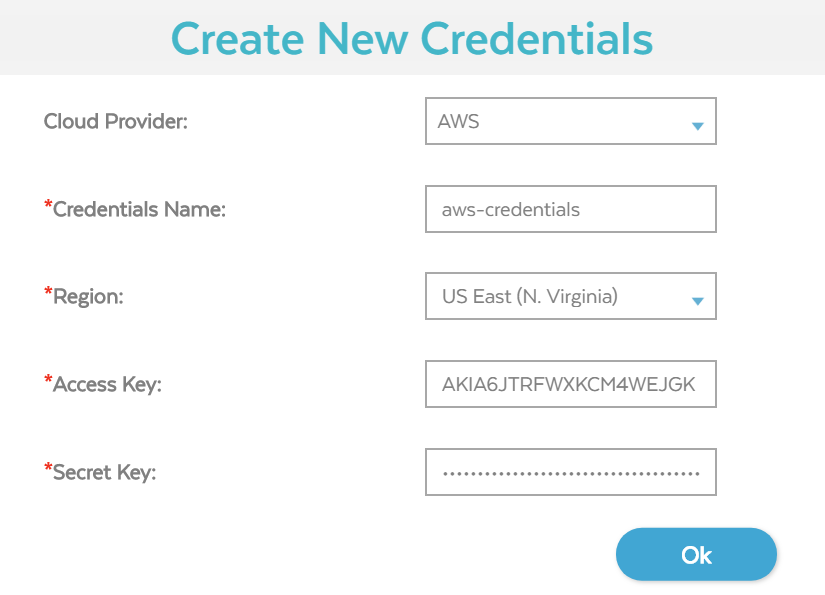
Click the **Create** button:



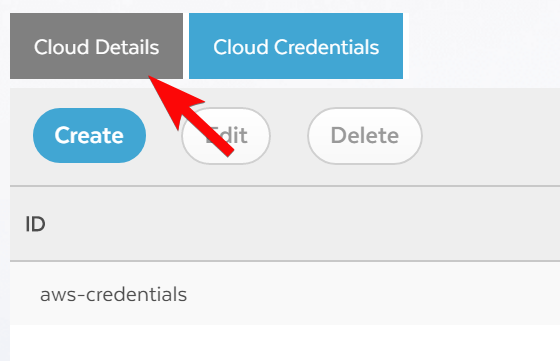
Fill in the form with the following information:

* **Cloud Provider:** AWS
* **Credentials Name:** a name of your choosing (eg. aws-credentials)
* **Region:** US East (N. Virginia)
* **Access Key:** The AWS Access key (listed on the resources panel on the left side of this page
* **Secret Key:** The AWS Secret key (listed on the resources panel on the left side of this page)

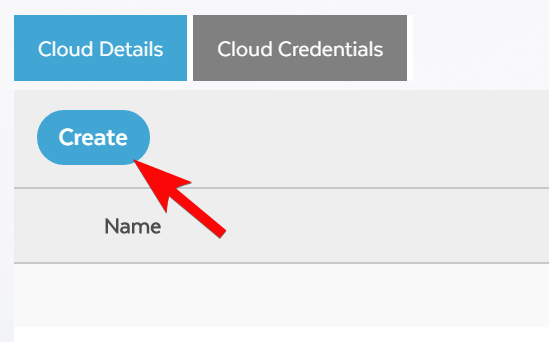
Click **Ok** to complete.



Now click the **Cloud Details** tab:

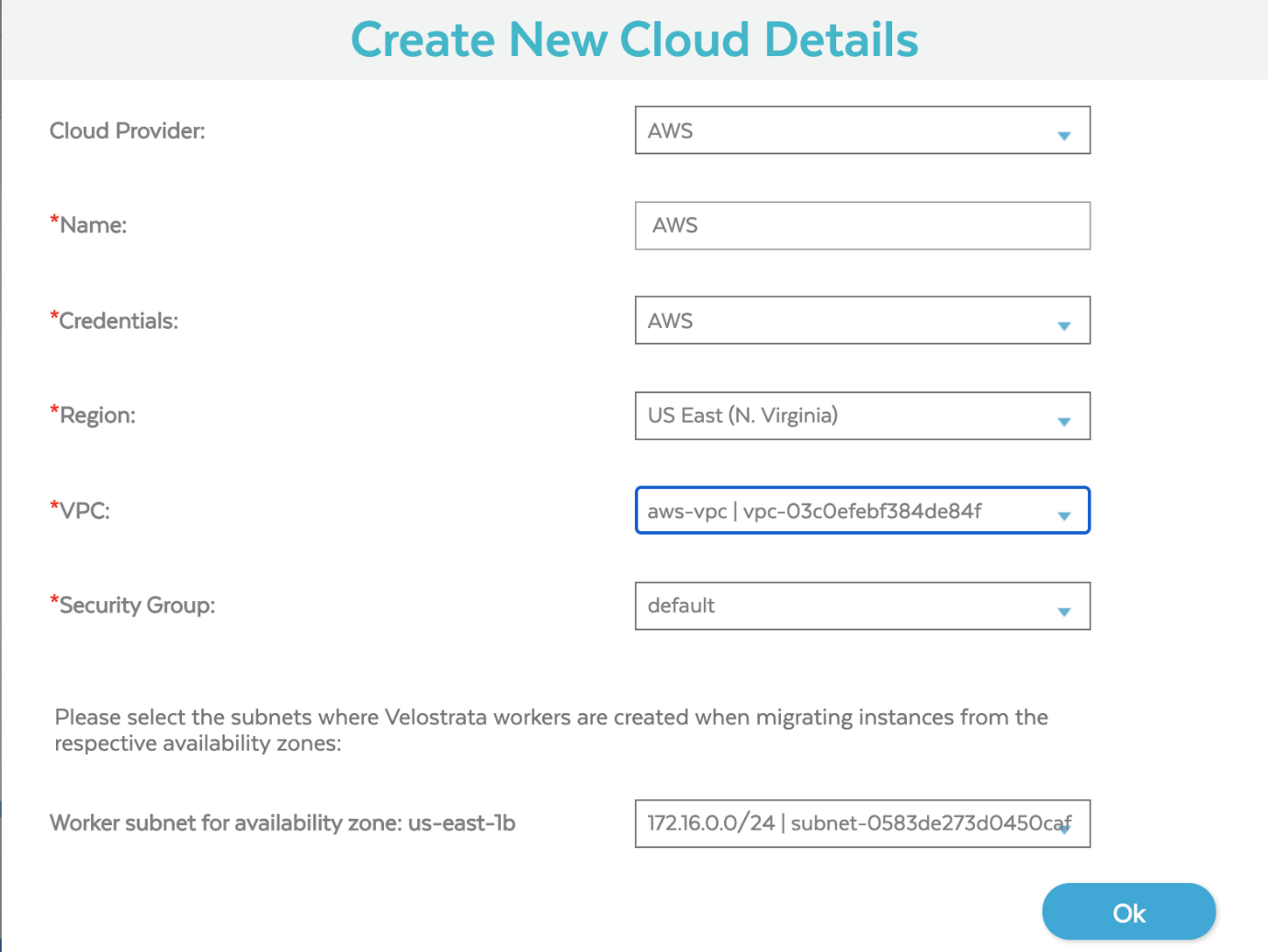


Click the **Create** button:



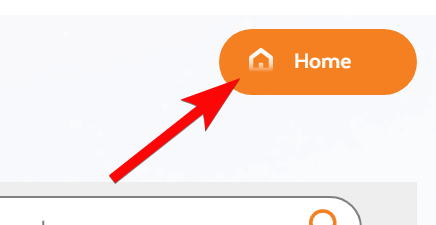
Fill in the form with the following information:

* **Cloud Provider:** AWS
* **Name:** AWS
* **Credentials:** Select the credentials created previously from dropdown menu
* **Region:** US East (N. Virginia)
* **VPC:** Select aws-vpc | vpc-xxx from the dropdown menu
* **Security Group:** default
* **Worker subnet for availability zone:** 172.16.0.0/24 | subnet-xxx



Click **Ok**.

Click the **Home** button:

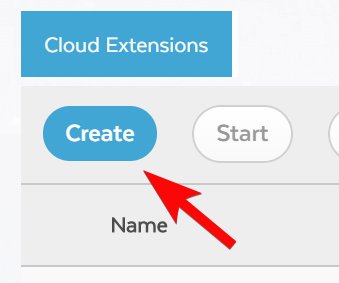


Now tell Migrate for Compute Engine about Google Cloud, which is your target.

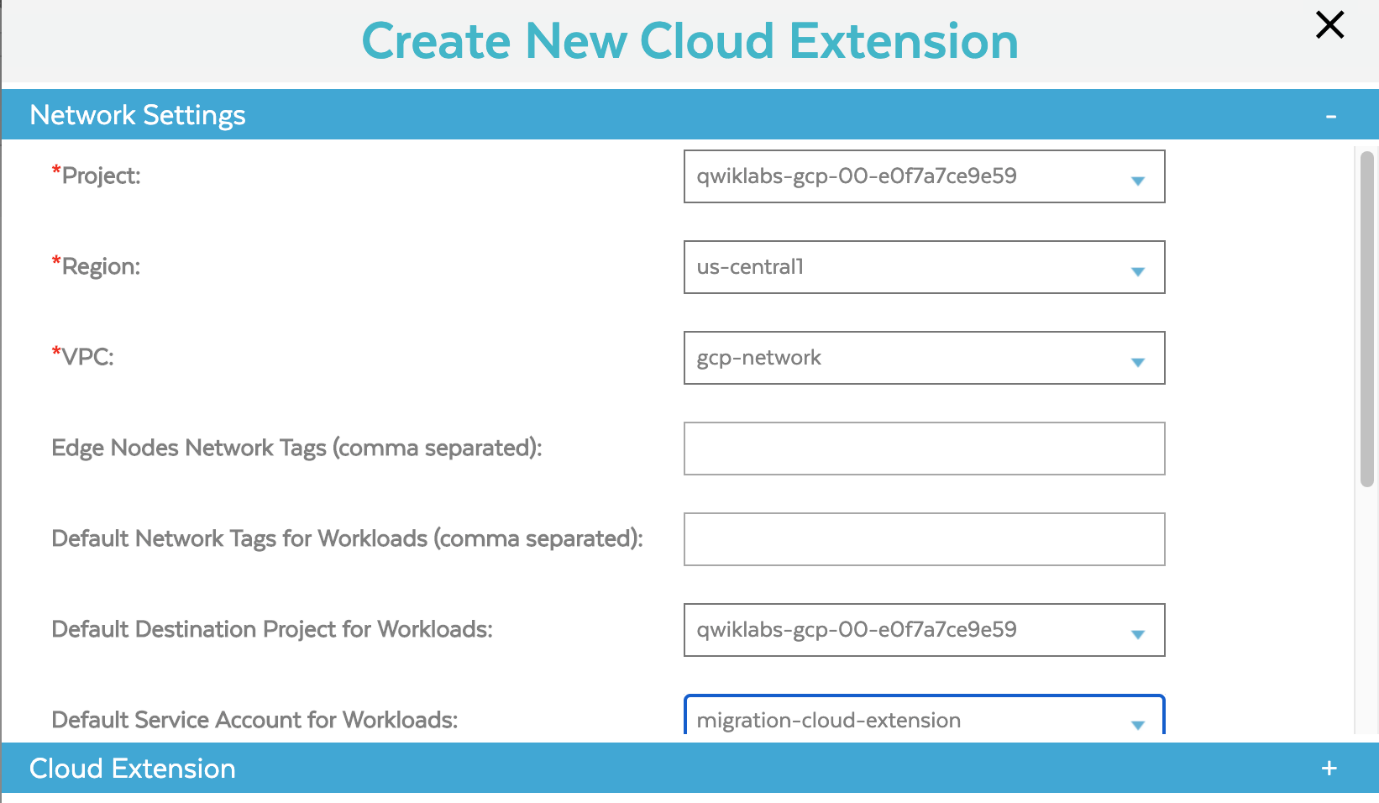
1. Click the **Target Cloud** icon:



1. On the **Cloud Extensions** tab, click the **Create** button:



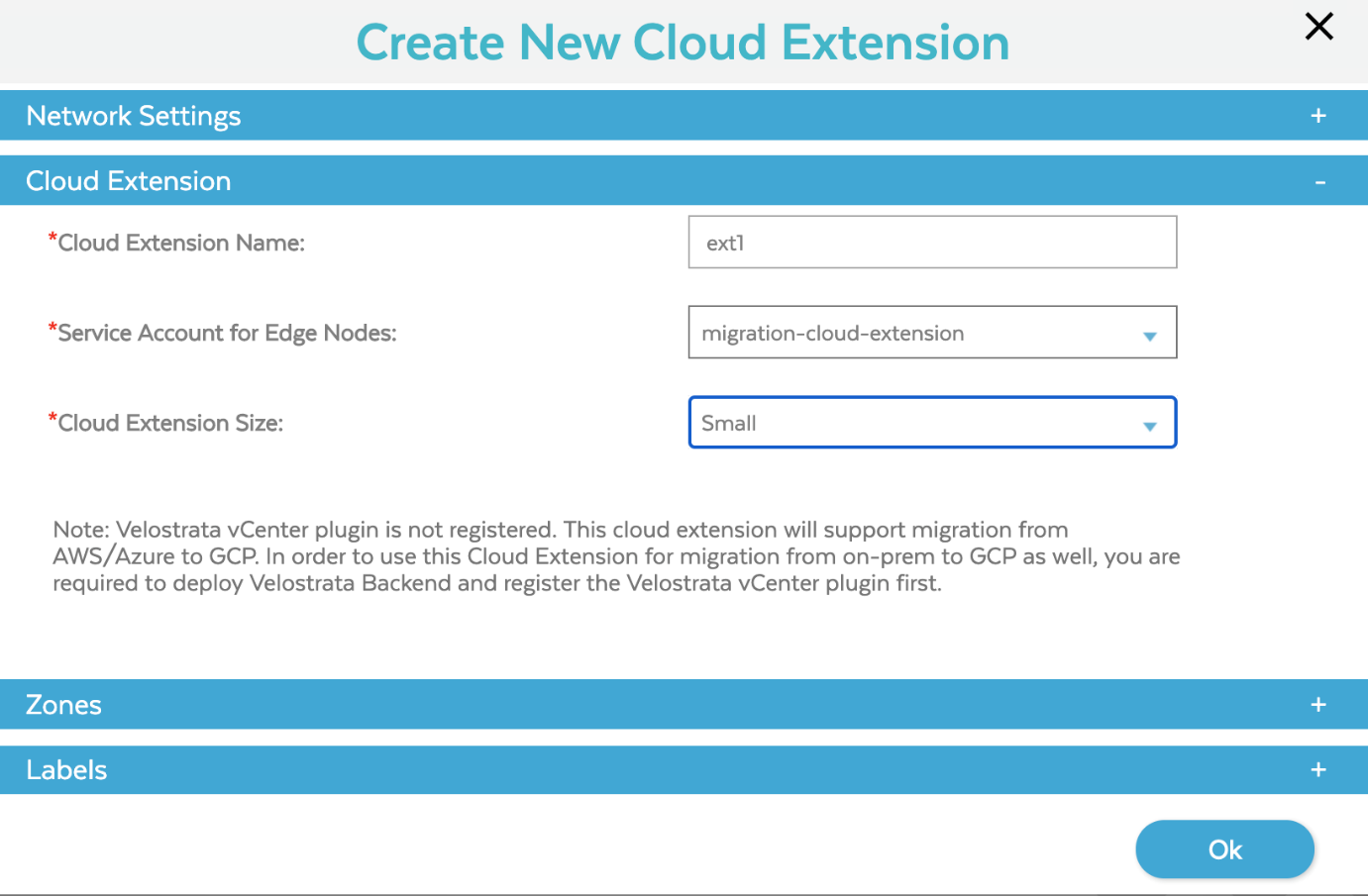
* **Project:** Select your qwiklabs-gcp-xxx
* **Region:** us-central1
* **VPC:** gcp-network
* **Default Destination Project for Workloads:** qwiklabs-gcp-xxx
* **Default Service Account for Workloads:** migration-cloud-extension



Now expand the next sections to complete the form:

**Cloud Extension**

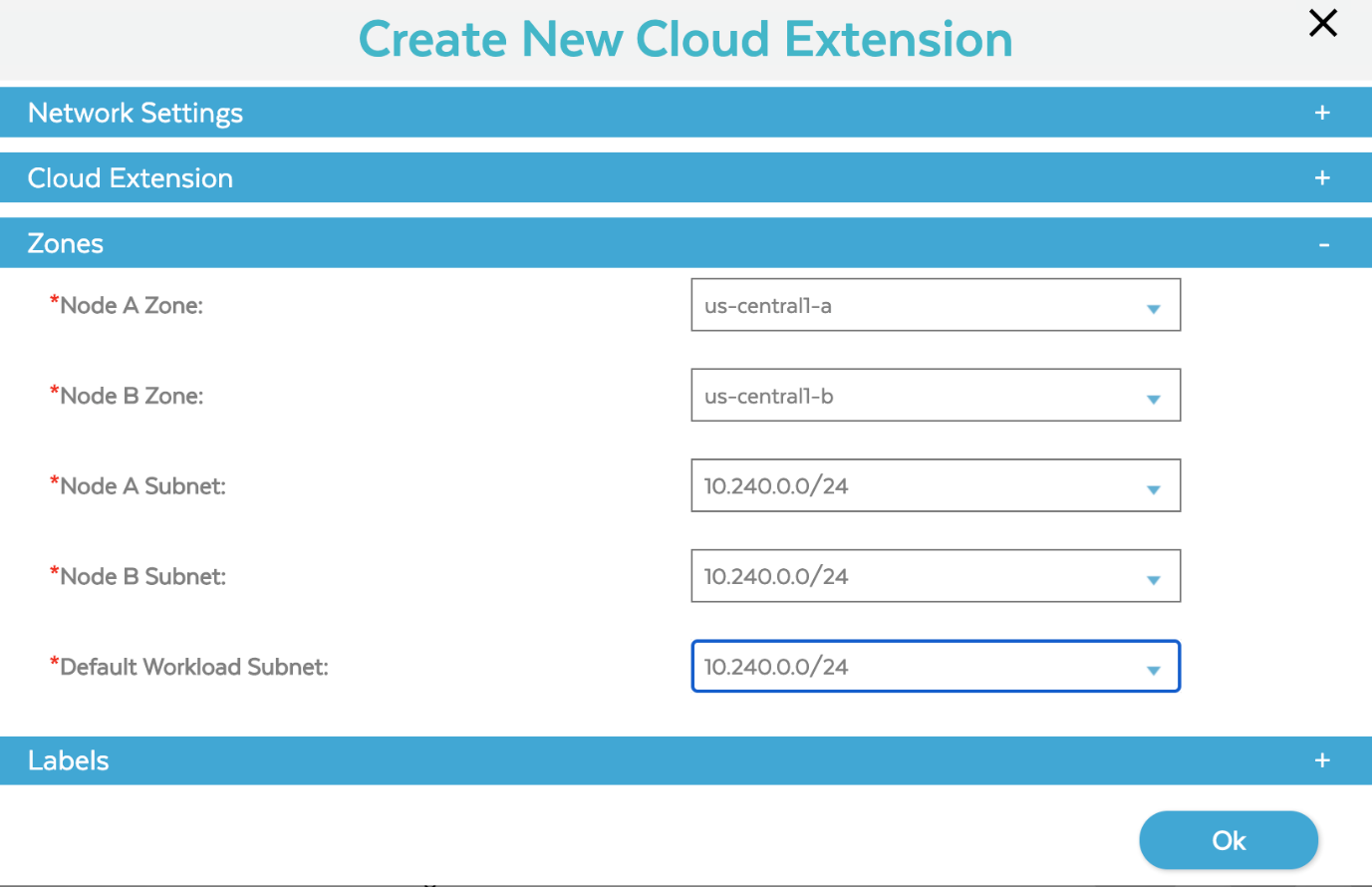
* **Cloud Extension Name:** ext1
* **Service Account for Edge Nodes:** migration-cloud-extension
* **Cloud Extension Size:** Small



**Zones**

* **Node A Zone:** us-central1-a
* **Node B Zone:** us-central1-b
* **Node A Subnet:** 10.240.0.0/24
* **Node B Subnet:** 10.240.0.0/24
* **Default Workload Subnet:** 10.240.0.0/24

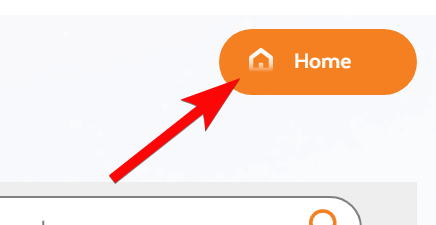
Click the **Ok** button.



You should now see **ext1** in the **Creating** state.

Wait until the creation is complete and it's in an **Active** state to continue. This will take a couple of minutes.

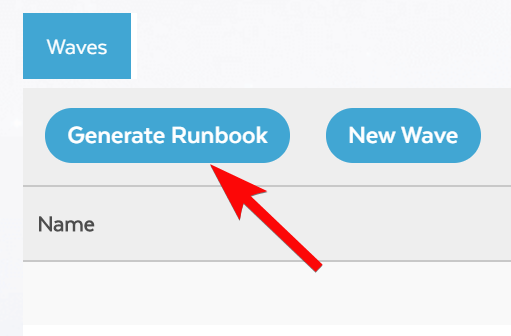
Click the **Home** button:



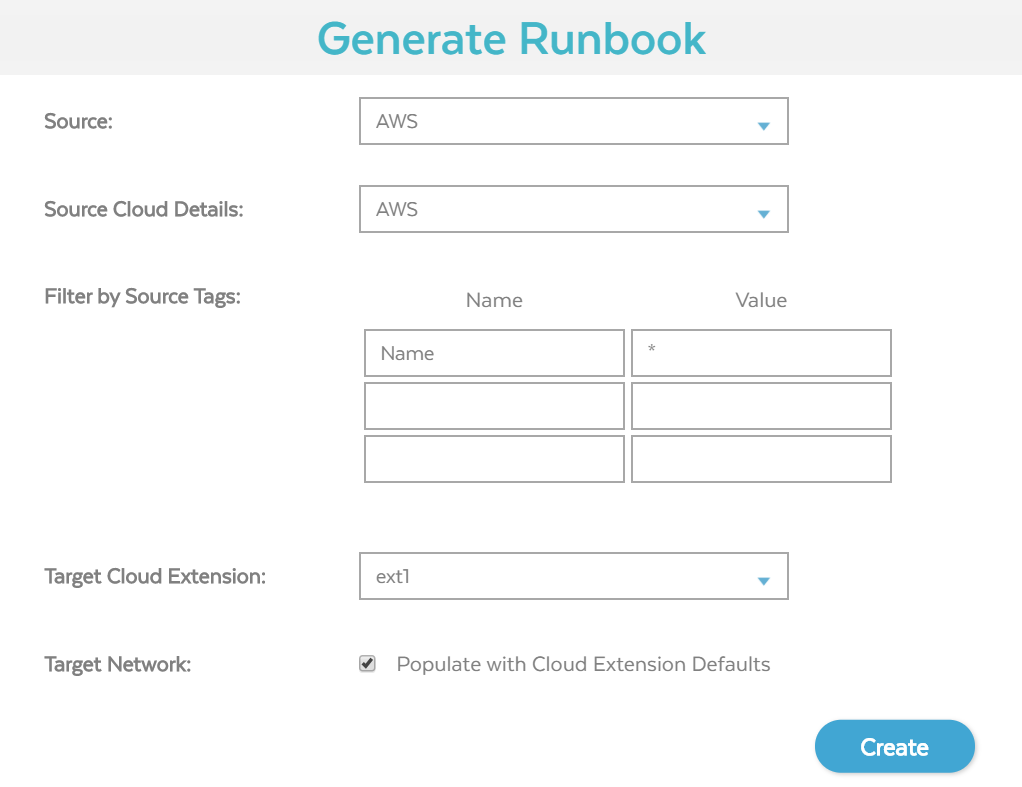
1. Click the **Migration Waves** icon:



Click **Generate Runbook**:



* **Source:** AWS
* **Source Cloud Details:** AWS
* **Filter by Source Tags:** Name: Name, Value: \*
* **Target Cloud Extension:** ext1
* **Target Network:** checked



Click **Create**.

A .csv file will be downloaded. You will edit this file to configure the migration settings.

Open [Google Sheets](https://docs.google.com/spreadsheets/u/0/?tgif=d) in a new tab. Make sure you're logged in with your lab credentials.

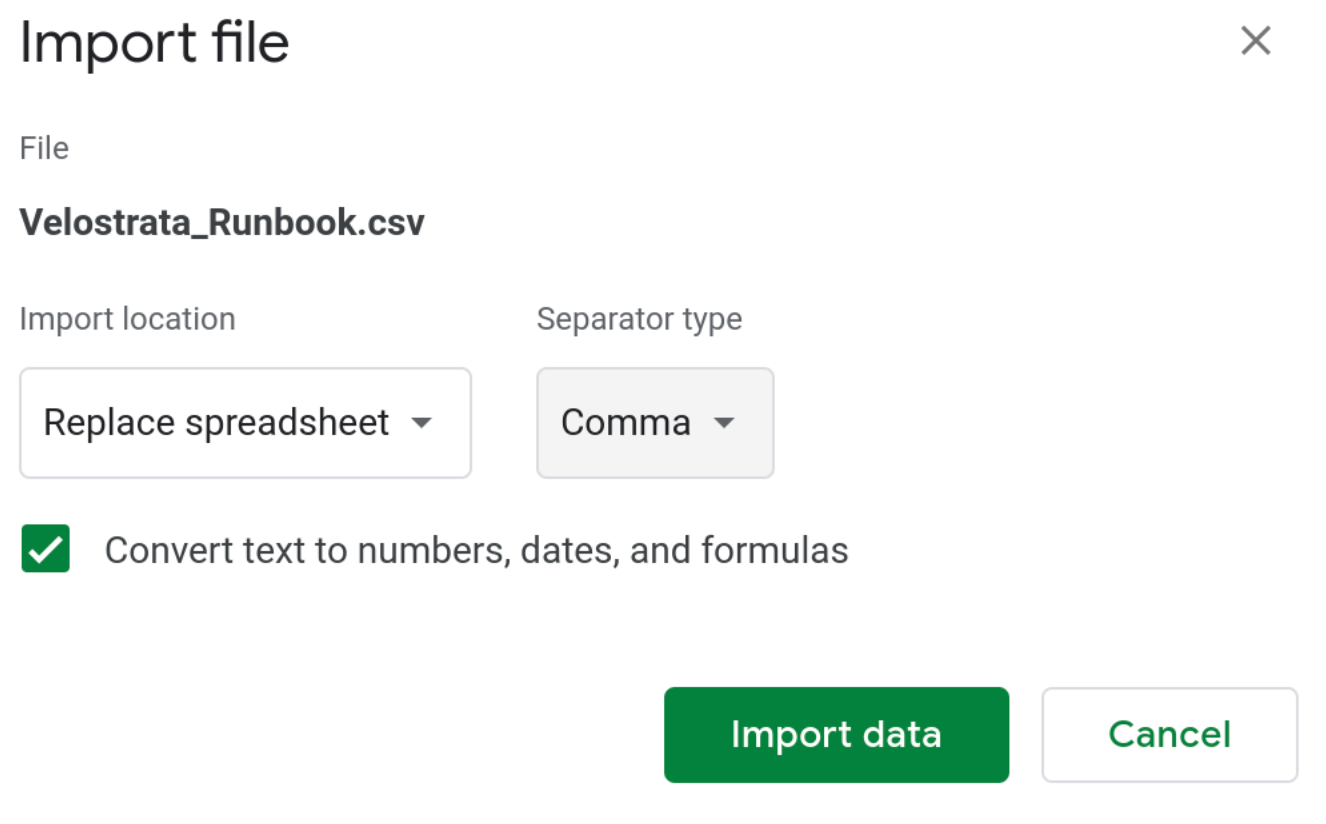
On the **Google Sheets** page, click **Blank** to start a new spread sheet.

Next, click **File** > **Import**.

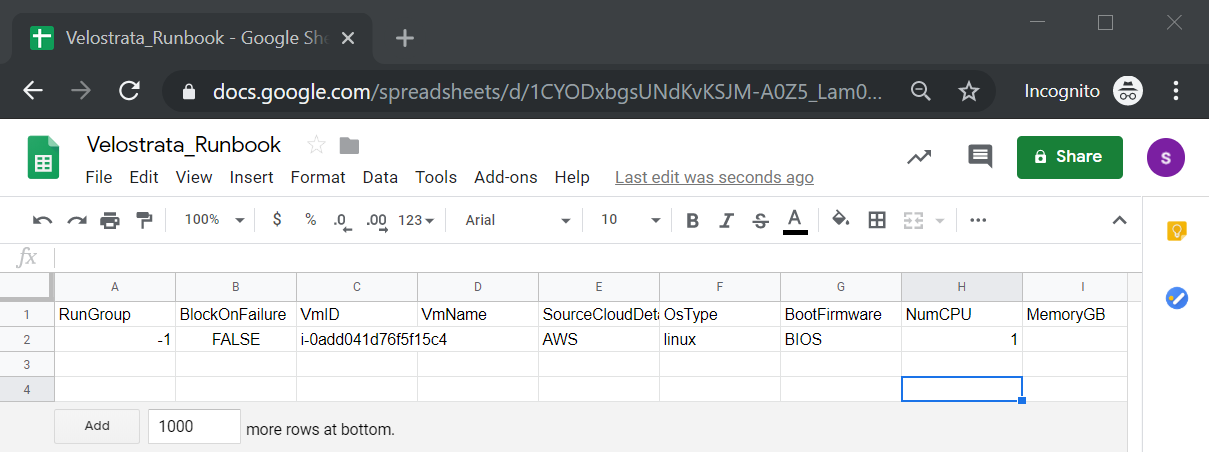
Click on the **Upload** tab and then drag your **Velostrata Runbook.csv** file into the window.

In the dialog that follows, select **Comma** as the Separator type.

Click **Import data**.



Edit the runbook

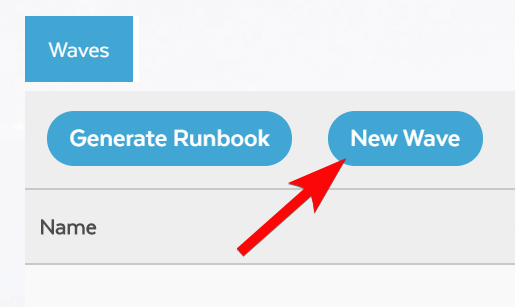


Change the RunGroup column value from -1 to 1 and set the TargetInstanceType column to be n1-standard-1.

Now, save the file back to your local file system: **File** > **Download** > **Comma-separated** values operation.

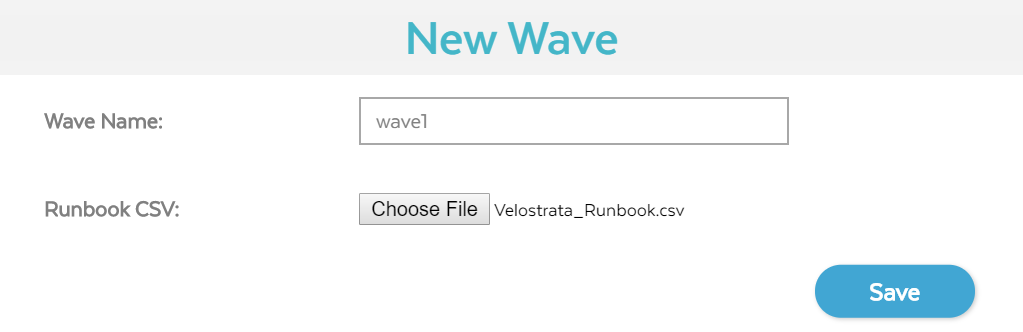
Create a new wave

Back in your Migrate for Compute Engine Manager (Velostrata Migration) tab, click the **New Wave** button.



* **Wave Name:** wave1
* **Runbook CSV:** Your edited CSV file

**Note**: Take care to select the CSV file that contains the edited changes and \*not\* the original one that was generated (assuming that you now have two files).

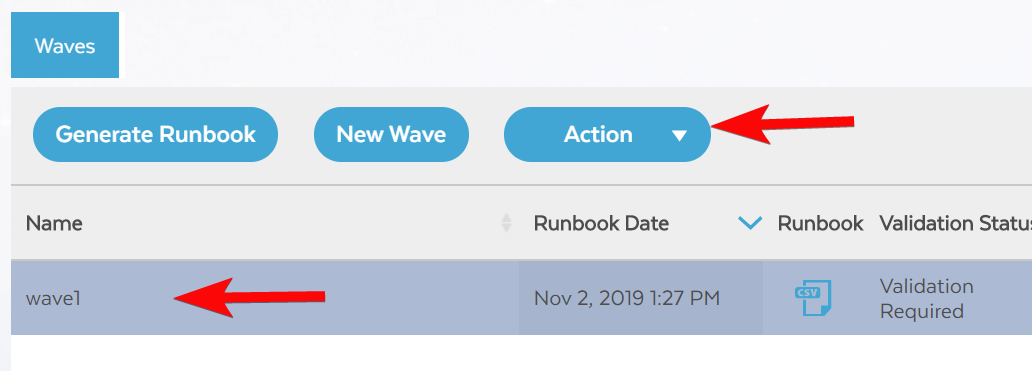


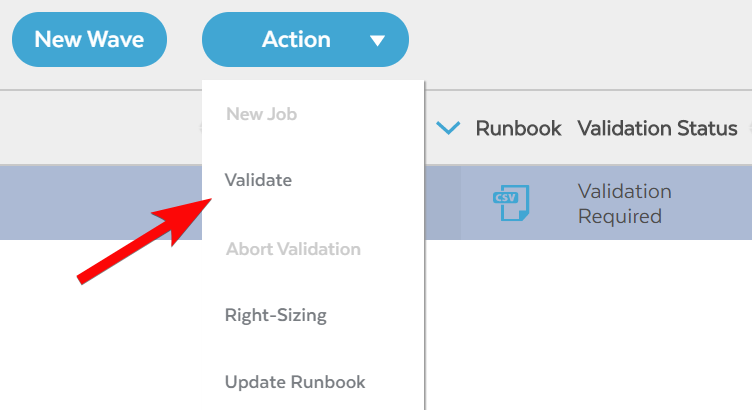
Click **Save**.

Validate the Wave

Click on the **wave1** line and it will become selected.

Click the **Action** dropdown menu and select **Validate** from within the menu.

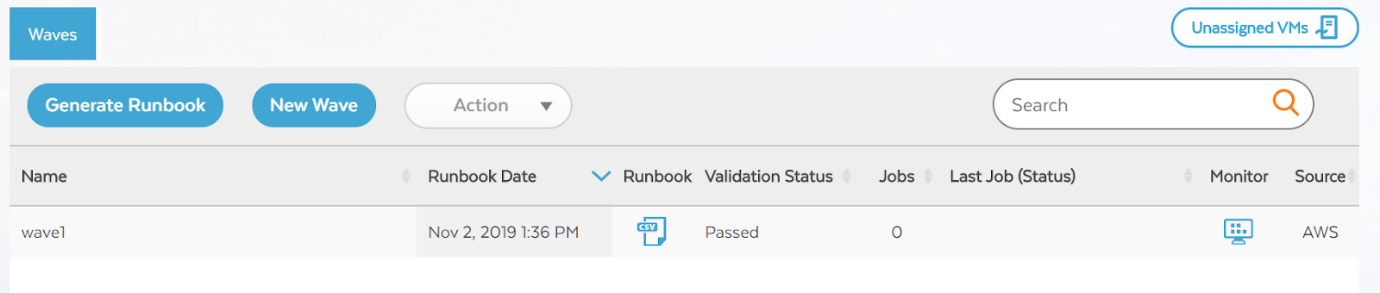




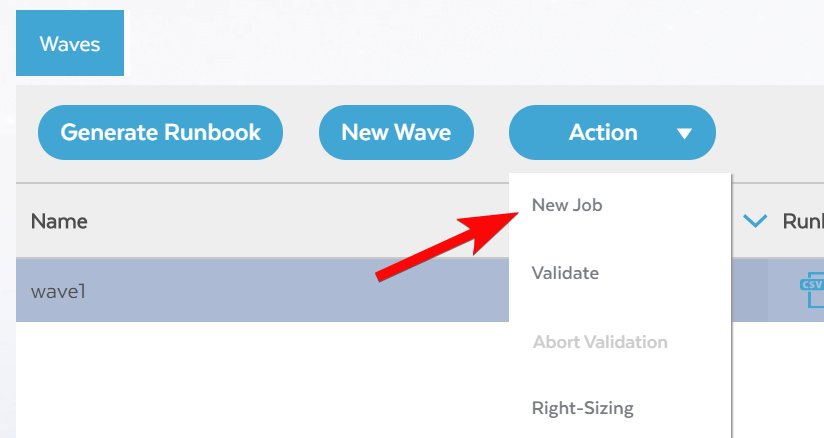
Click **Yes** in the Run Validation dialog box.



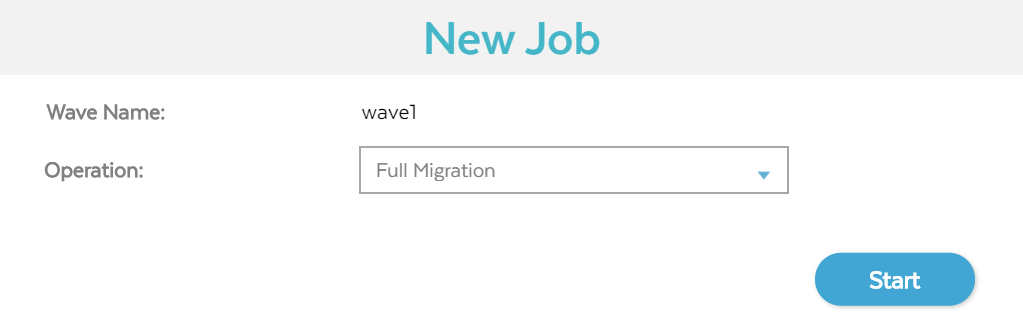
The status will change to Validating and then, after a few seconds it will change again to Passed.



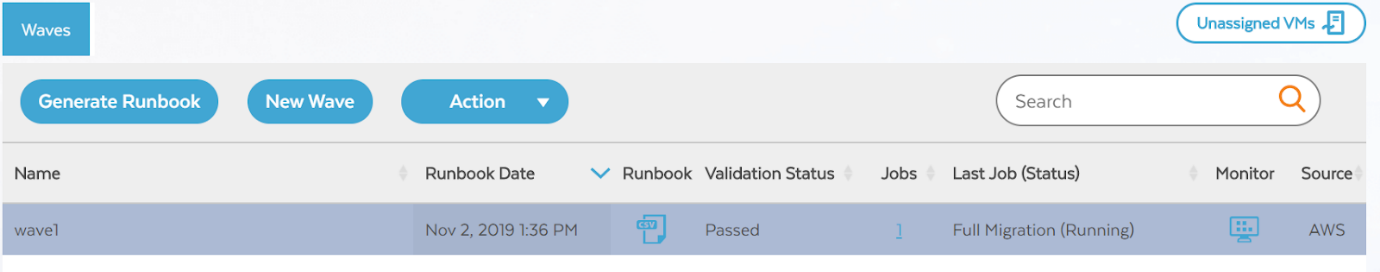
With the wave1 line still selected, click the **Action** pulldown menu and select the **New Job** entry.



In the New Job dialog, select **Full Migration** operation and click **Start**.



The status will now change to Full Migration (Running).

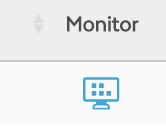


The migration is now progressing and you must wait for the corresponding Compute Engine to become available.

The migration proceeds in two major phases.

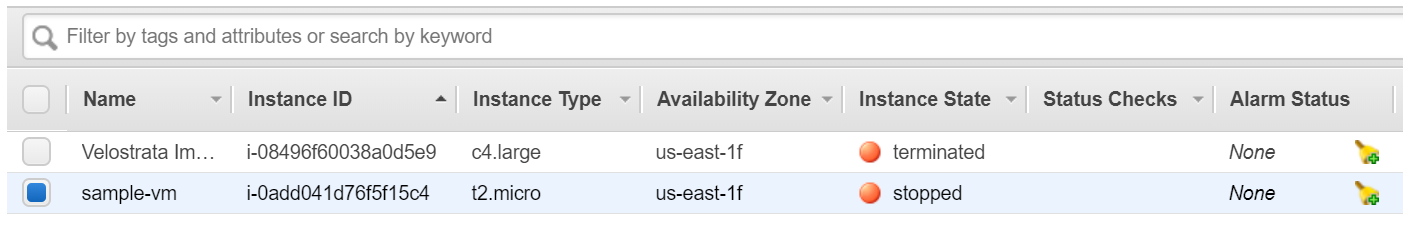
* The first is the start up of the Compute Engine by bringing in enough of the original VM to start.
* From there, the remainder of the VM disk will be streamed in the background.

Click on the **Monitor icon**:



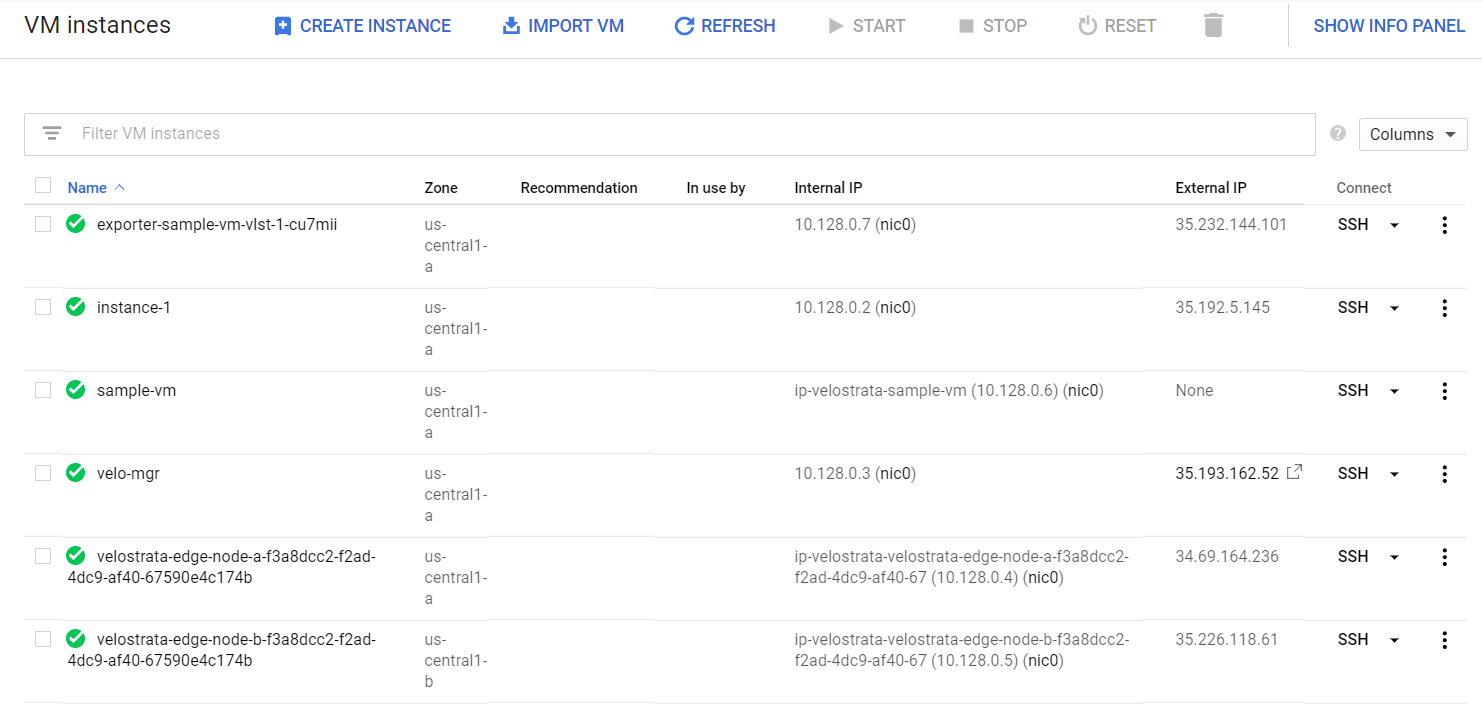
You now see a record line for the VM being migrated. This should take about 10 minutes.

While you're waiting, go look at the AWS instance, and see that the machine you're migrating has been stopped.



Use the refresh button to see the most up to date information.

Also look in the Cloud Console under **Compute Engine** - eventually you'll see the aws-vm-us-east-1 machine from AWS appear.

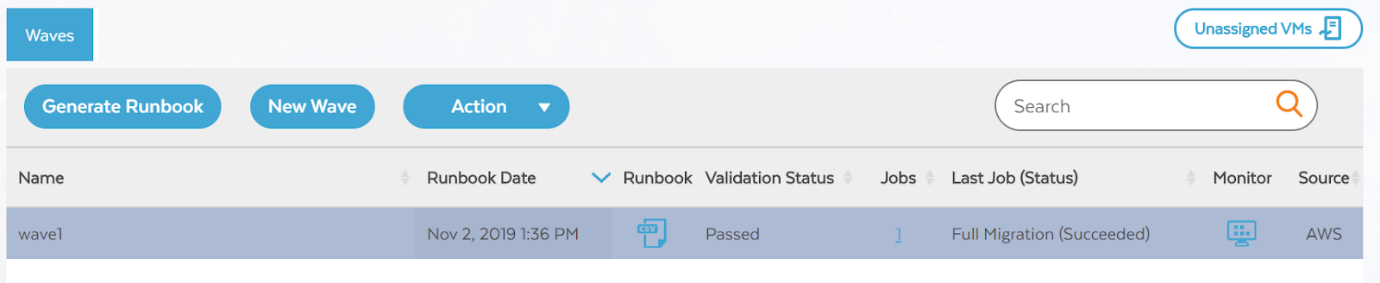


When the status changes from empty or Moving To Target Cloud to either:

* Cache on demand
* Migrating
* Preparing to Detach

then the VM is now available on Google Cloud and is ready to be used.

When the VM has been 100% migrated, the status will change to Fully Migrated.



You don't need to wait for this status, continue to the next section.

Click *Check my progress* to verify the objective.

Migrate

Check my progress

**Test the new Google Cloud machine**

In the Cloud Console, go select **Compute Engine** > **VM Instances** from the **Navigation Menu**.

You should see the **aws-vm-us-east-1** vm. Now you need to assign it a Public IP to enable SSH connection.

To attach an external IP address, follow these steps:

* Click on the machine name to open the VM instance details page.
* Click the **Edit** button.
* Scroll down to Network Interfaces, then click the **Edit/pencil** icon.
* Under External IP select **Ephemeral** then click **Done**.
* At the bottom of the page click **Save**.
* Click **VM instances** again, and now see the Public IP assigned.

Now, in your Cloud Shell, run this command, replacing **PUBLIC IP** with the public IP of your migrated VM:

ssh -i ~/.ssh/vm-ssh-key ubuntu@[PUBLIC\_IP]

Copied!

content\_copy

Type "yes"

This will log you into the migrated VM running on Google Cloud.

Run the ls command to see the file that was created earlier on the AWS EC2 instance.

Click *Check my progress* to verify the objective.

Test the new Google Cloud machine

Check my progress

**Congratulations!**

You have successfully migrated the AWS VM instance to a corresponding Google Cloud Compute Engine.



Finish Your Quest

This self-paced lab is part of the Qwiklabs [VM Migration](https://google.qwiklabs.com/quests/87) Quest. A Quest is a series of related labs that form a learning path. Completing a Quest earns you one of the badges above, to recognize your achievement. You can make your badge (or badges) public and link to them in your online resume or social media account. Enroll in a quest and get immediate completion credit if you've taken this lab. [See other available Qwiklabs Quests](http://google.qwiklabs.com/catalog).

Take your Next Lab

Continue your Quest with [VM Migration: Assessment](https://google.qwiklabs.com/catalog_lab/1492), or check out:

* [VM Migration: Introduction to StratoZone Assessments](https://google.qwiklabs.com/catalog_lab/2445)

Google Cloud Training & Certification

...helps you make the most of Google Cloud technologies. [Our classes](https://cloud.google.com/training/courses) include technical skills and best practices to help you get up to speed quickly and continue your learning journey. We offer fundamental to advanced level training, with on-demand, live, and virtual options to suit your busy schedule. [Certifications](https://cloud.google.com/certification/) help you validate and prove your skill and expertise in Google Cloud technologies.

**Manual Last Updated: December 24, 2021**

**Lab Last Tested: December 24, 2021**

Copyright 2021 Google LLC All rights reserved. Google and the Google logo are trademarks of Google LLC. All other company and product names may be trademarks of the respective companies with which they are associated.