The Best Team in this Course: Random Number Documentation

Overview

This repo consists of four applications, each of which sends a HTTP GET request to get a random number between 1 and 1,000,000. The test applications perform 1000 GET request to ensure that at least 750 of them are unique.

Python App Engine

Go to Google Cloud. Open the project selection dialog by clicking on the project name in the top left of the console. Create a new project and navigate to App Engine -> Dashboard.

Click "Create Application". Select a region (us-central is fine) and then click "Create App."

On the next screen, select Python for the language and a Standard runtime. Click next.

Now, open up a Cloud Shell using the button in the top right of the dashboard. When it opens, verify that your project is the current project. If it is not, run: `gcloud init` and select the correct project when prompted.

Make a directory for your app by running `mkdir myapp` in the Cloud Shell. Then, move into the directory with `cd myapp`.

Now, clone the git repo for this application using:

```
git clone https://github.com/katyfelkner/cs4263-rng.git
```

Move into the directory for the Python App Engine app using:

```
cd cs4263-rng/PythonAppEngine
```

Install flask (dependency for our application) using:

```
sudo pip3 install -r requirements.txt
```

When the installation is finished, deploy the App Engine app using:

```
gcloud app deploy
```

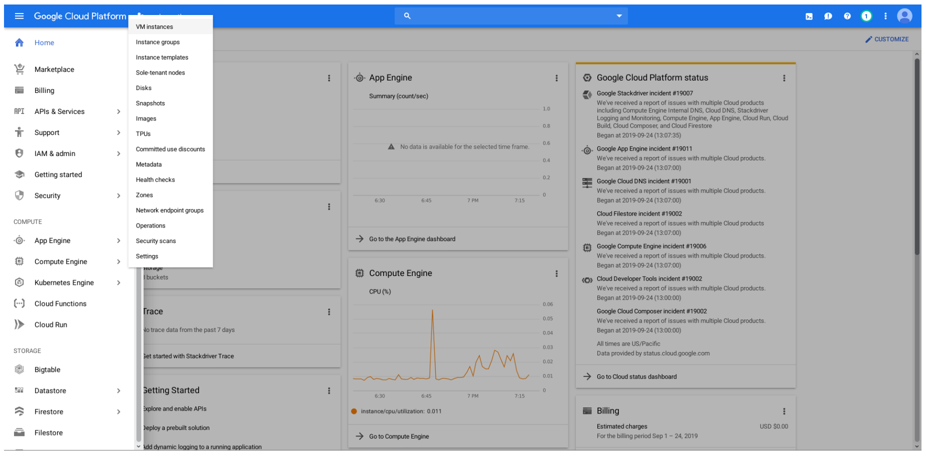
Press 'Y' to confirm when prompted.

Once your app is deployed, you can view it at <projectname>.appspot.com.

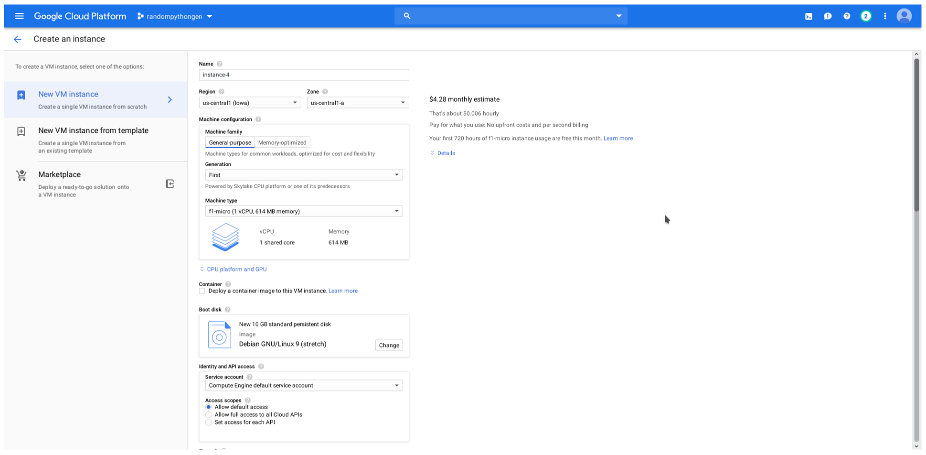
Setup The Python Instance

Configuring Google Cloud

Go to Cloud Console and navigate to compute engine --> vm instances.



Click create a new instance and setup your instances as follows.


Google Cloud Platform
randompythongen
Create an instance
To create a VM instance, select one of the options:
New VM instance
Create a single VM instance from scratch
New VM instance from template
Create a single VM instance from an existing template
Marketplace
Deploy a ready-to-go solution onto a VM instance
CPU platform and GPU
Details
Change

Name: instance-4
Region: us-central1 (Iowa)
Zone: us-central1-a
Machine configuration
Machine family: General-purpose | Memory-optimized
Machine types for common workloads, optimized for cost and flexibility
Generation: First
Powered by Skylake CPU platform or one of its predecessors
Machine type: f1-micro (1 vCPU, 614 MB memory)
vCPU: 1 shared core
Memory: 614 MB
Container: Deploy a container image to this VM instance. Learn more
Boot disk: New 10 GB standard persistent disk
Image: Debian GNU/Linux 9 (stretch)
Identity and API access
Service account: Compute Engine default service account
Access scopes:
● Allow default access
○ Allow full access to all Cloud APIs
○ Set access for each API

$4.28 monthly estimate
That's about $0.006 hourly
Pay for what you use: No upfront costs and per second billing
Your first 720 hours of f1-micro instance usage are free this month. Learn more

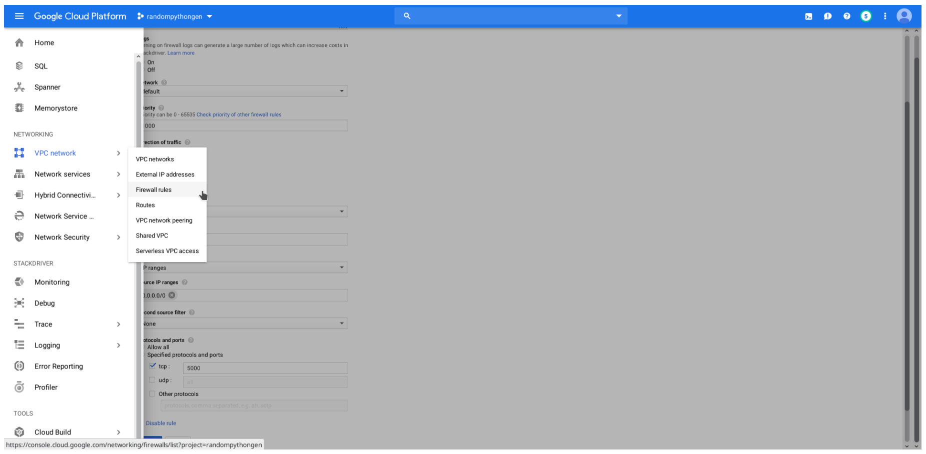
Change the machine type to f1-micro.

For machine type use DebianGNU/Linux 9 (stretch).

Allow HTTP(S) traffic.

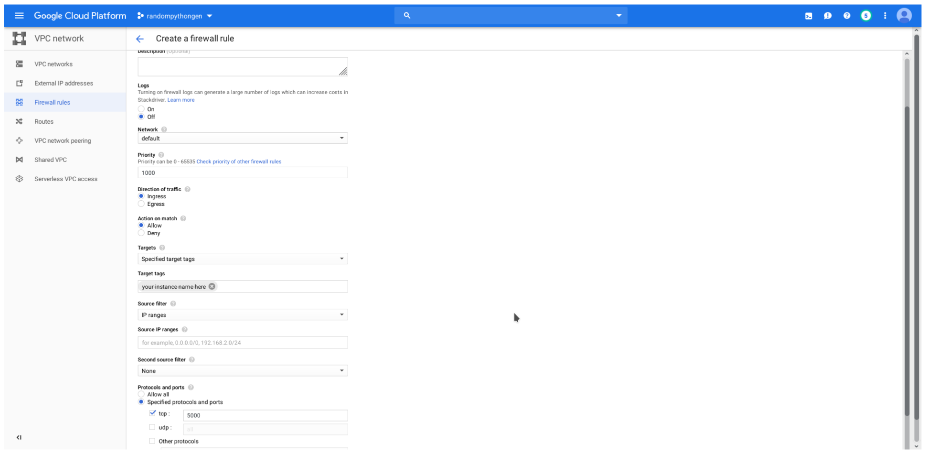
Click the arrows to expand the settings menu. I recommend setting a static IP address although this is not strictly necessary for testing purposes. Click create and wait for it to allocate resources and start.

Next we need to open up port 5000 to allow inbound traffic for TCP. Go to networking --> VPC Network --> Firewall Rules as follows.



https://console.cloud.google.com/networking/firewalls/list?project=randompythongen

Apply the following rules, naming the rule as you see fit.



Description (Optional)

Logs
Turning on firewall logs can generate a large number of logs which can increase costs in Stackdriver. Learn more
On [✓ Off]

Network: default

Priority
Priority can be 0 - 65535 Check priority of other firewall rules
1000

Direction of traffic: [✓ Ingress] Egress

Action on match: [✓ Allow] Deny

Targets: Specified target tags

Target tags: your-instance-name-here

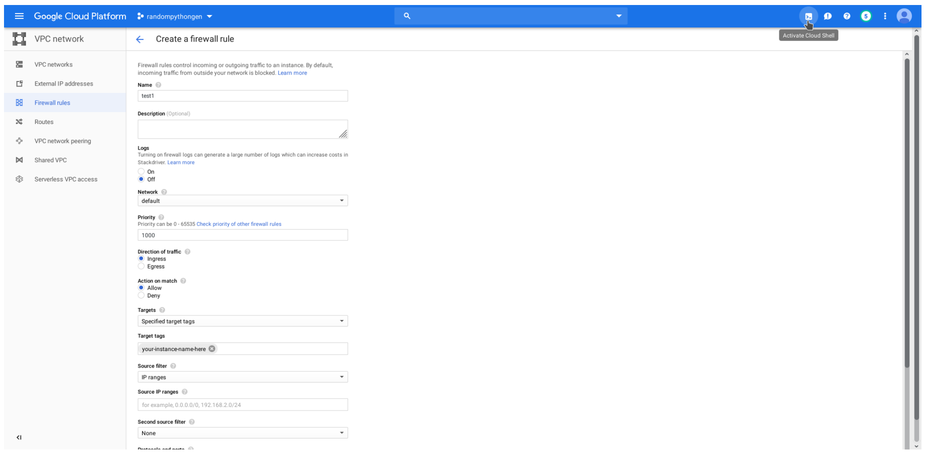
Source filter: IP ranges

Source IP ranges: for example, 0.0.0.0/0, 192.168.2.0/24

Second source filter: None

Protocols and ports: Allow all [✓ Specified protocols and ports]
[✓ tcp :] 5000
udp : all
Other protocols

The instance is now up and running. Let's go to the google cloud console now (or your prefferred method of connecting to the instance).



Create a firewall rule

Firewall rules control incoming or outgoing traffic to an instance. By default, incoming traffic from outside your network is blocked. Learn more

Name: test1

Description (Optional):

Logs
Turning on firewall logs can generate a large number of logs which can increase costs in Stackdriver. Learn more
On [✓ Off]

Network: default

Priority
Priority can be 0 - 65535 Check priority of other firewall rules
1000

Direction of traffic: [✓ Ingress] Egress

Action on match: [✓ Allow] Deny

Targets: Specified target tags

Target tags: your-instance-name-here

Source filter: IP ranges

Source IP ranges: for example, 0.0.0.0/0, 192.168.2.0/24

Second source filter: None

Install the Requirements

Enter the following lines into your console.

```
sudo apt install git

sudo apt install python-pip

pip install flask

git clone https://github.com/katyfelkner/cs4263-rng

cd cs4263-rng/PythonInstance/

python main.py
```

And you should be up and running!

Go to compute engine --> vm instances (where we started). View the external IP address associated with the instance you've been working in. Append :5000 to the end of it and type it in your browser page. For example, here is the link to our python instance which generates a random number. http://35.238.165.97:5000/

Final tip - to add permanence to this flask application do the following:

```
sudo apt install tmux

tmux

python ~/cs4263-rng/PythonInstance/main.py
```

Press control+'b' and then d. Then you are free to exit your instance and the flask server will continue to run.

Java App Engine

First you will need a google cloud instance. You can navigate to `https://console.cloud.google.com/getting-started?tutorial=java_gae_quickstart` for a quick guild to creating and deploying your first app engine application.

Tutorial basics

- From Login screen Click `Go to Console` Then click `Navigation menu` Which is a hamburger icon on the right.
- In Navigation Menu click AppEngine.
- Create a new project.
- Once Application is active and created you can click the `Activate Cloud Shell` in order to tunnel into your Engines instance.
- Once tunneled into your Engine. Type

```
git clone https://github.com/GoogleCloudPlatform/appengine-try-java
```

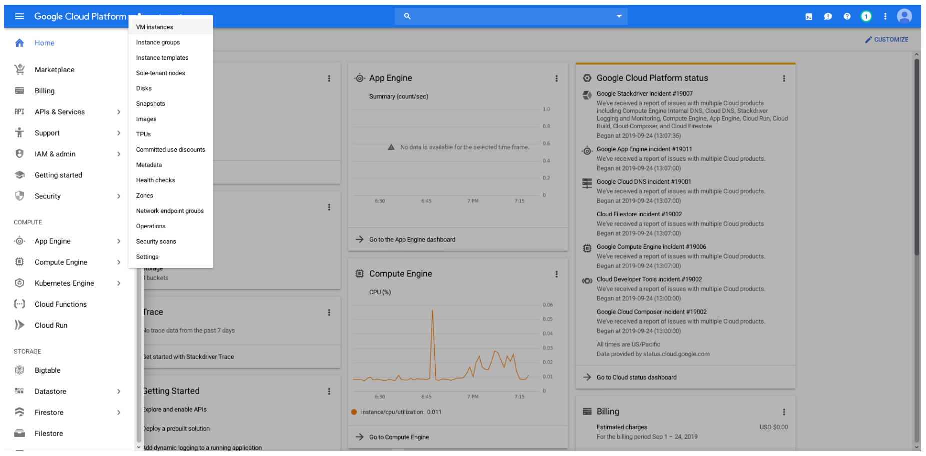
- Navigate to Java folder, `cd appengine-try-java/src/main/java/myapp` and open `DemoServlet.java` with your favorite editor.
  - EX: `emacs DemoServlet.java`
- Replace the `Hello world` response with desired response. It can either be in JSON format or plane string text.
- Save your changes. (in emacs it's Ctrl+S then Ctrl+C)
- Navigate to root folder.
  - `cd ~/appengine-try-java`
- Run Maven `mvn appengine:run` to handle all dependecies and run the server that will compile and run java on a server.
- You can then click the `Web Preview` button to view port 8080 and see your return result.
  - If you chose not to keep object in JSON format add `/demo` to the URL right after port.
- Once finished view your endpoints you can terminate the `Preview Instance` by hitting `Ctrl+c` inside `Cloud Shell`
- When you are ready to deploy you application you will need to create gcloud application.
  - type `gcloud app create` in Shell
  - Then configure your appengine domain by typing `gcloud config set project <YOUR-PROJECT>`
    - You will then be prompted to select a server location.
    - Choose the closest region to you or your desired audience
  - Lastly you can deploy by typing `mvn appengine:deploy`
  - You can now access your instance at `http://<Your-Project-Name>.appspot.com/demo`

Java VM Instance

Setup Java VM

Configuring Google Cloud

Go to Cloud Console and navigate to compute engine --> vm instances.



Google Stackdriver incident #19007
We've received a report of issues with multiple Cloud products including Compute Engine Internal DNS, Cloud DNS, Stackdriver Logging and Monitoring, Compute Engine, App Engine, Cloud Run, Cloud Build, Cloud Composer, and Cloud Firestore
Began at 2019-09-24 (13:07:35)

Google App Engine incident #19011
We've received a report of issues with multiple Cloud products.
Began at 2019-09-24 (13:07:00)

Google Cloud DNS incident #19001
We've received a report of issues with multiple Cloud products.
Began at 2019-09-24 (13:07:00)

Cloud Filestore incident #19002
We've received a report of issues with multiple Cloud products.
Began at 2019-09-24 (13:07:00)

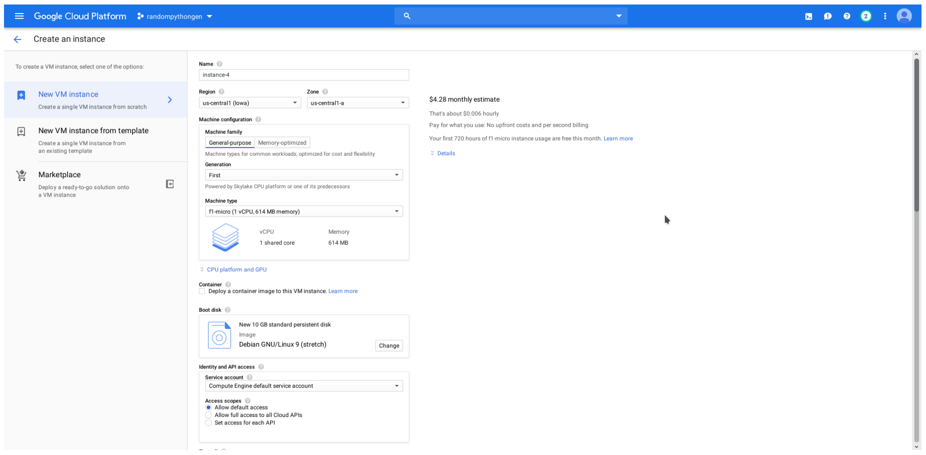
Google Compute Engine incident #19006
We've received a report of issues with multiple Cloud products.
Began at 2019-09-24 (13:07:00)

Cloud Developer Tools incident #19002
We've received a report of issues with multiple Cloud products.
Began at 2019-09-24 (13:00:00)

Google Cloud Composer incident #19002
We've received a report of issues with multiple Cloud products.
Began at 2019-09-24 (13:00:00)

All times are US/Pacific
Data provided by status.cloud.google.com

Click create a new instance and setup your instances as follows.



To create a VM instance, select one of the options:

New VM instance — Create a single VM instance from scratch

New VM instance from template — Create a single VM instance from an existing template

Marketplace — Deploy a ready-to-go solution onto a VM instance

Name: instance-4

Region: us-central1 (Iowa)

Zone: us-central1-a

Machine configuration

Machine family: General-purpose | Memory-optimized

Machine types for common workloads, optimized for cost and flexibility

Generation: First

Powered by Skylake CPU platform or one of its predecessors

Machine type: f1-micro (1 vCPU, 614 MB memory)

vCPU: 1 shared core

Memory: 614 MB

Container: Deploy a container image to this VM instance. Learn more

Boot disk: New 10 GB standard persistent disk

Image: Debian GNU/Linux 9 (stretch)

Identity and API access

Service account: Compute Engine default service account

Access scopes:
- [✓ Allow default access]
- Allow full access to all Cloud APIs
- Set access for each API

$4.28 monthly estimate

That's about $0.006 hourly

Pay for what you use: No upfront costs and per second billing

Your first 720 hours of f1-micro instance usage are free this month. Learn more

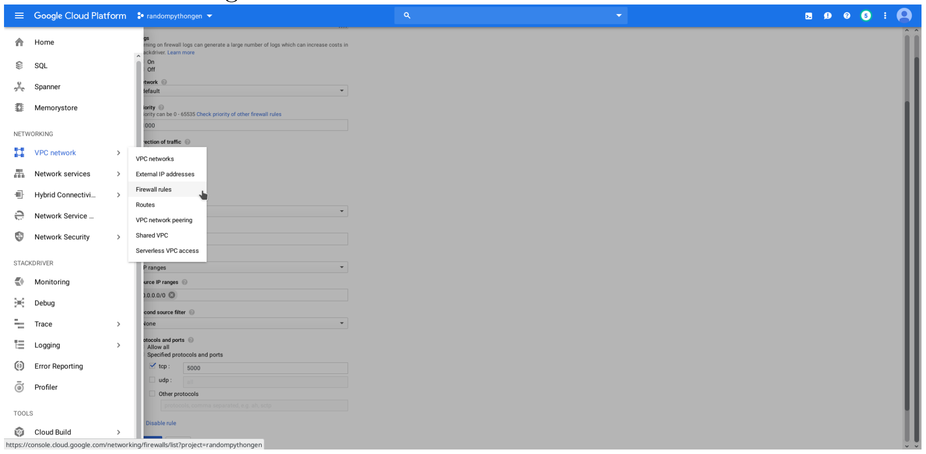
Change the machine type to f1-micro.

For machine type use DebianGNU/Linux 9 (stretch).

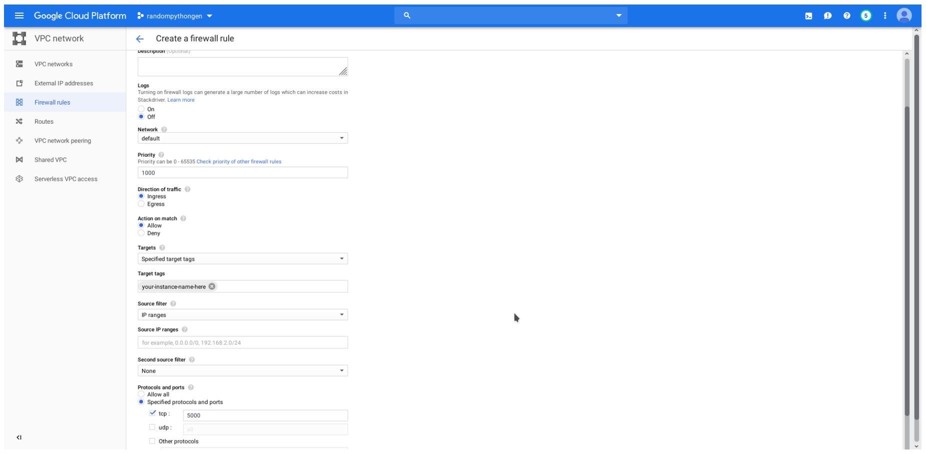
Allow HTTP(S) traffic.

Click the arrows to expand the settings menu. I recommend setting a static IP address although this is not strictly necessary for testing purposes. Click create and wait for it to allocate resources and start.

Next we need to open up port 5000 to allow inbound traffic for TCP. Go to networking --> VPC Network --> Firewall Rules as follows.



Apply the following rules, naming the rule as you see fit.



Description (Optional)

Logs
Turning on firewall logs can generate a large number of logs which can increase costs in Stackdriver. Learn more
On [✓ Off]

Network: default

Priority
Priority can be 0 - 65535 Check priority of other firewall rules
1000

Direction of traffic: [✓ Ingress] Egress

Action on match: [✓ Allow] Deny

Targets: Specified target tags

Target tags: your-instance-name-here

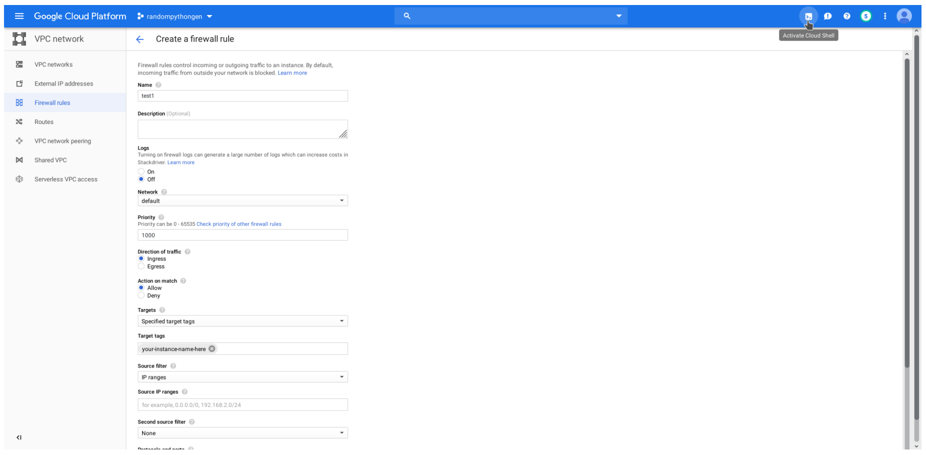
Source filter: IP ranges

Source IP ranges: for example, 0.0.0.0/0, 192.168.2.0/24

Second source filter: None

Protocols and ports
Allow all [✓ Specified protocols and ports]
[✓ tcp :] 5000
udp : all
Other protocols

The instance is now up and running. Let's go to the google cloud console now (or your preferred method of connecting to the instance).



Create a firewall rule

Firewall rules control incoming or outgoing traffic to an instance. By default, incoming traffic from outside your network is blocked. Learn more

Name: test1

Description (Optional)

Logs
Turning on firewall logs can generate a large number of logs which can increase costs in Stackdriver. Learn more
On [✓ Off]

Network: default

Priority
Priority can be 0 - 65535 Check priority of other firewall rules
1000

Direction of traffic: [✓ Ingress] Egress

Action on match: [✓ Allow] Deny

Targets: Specified target tags

Target tags: your-instance-name-here

Source filter: IP ranges

Source IP ranges: for example, 0.0.0.0/0, 192.168.2.0/24

Second source filter: None

Install the Requirements

Enter the following lines into your console.

```
sudo apt install git
```

‘wget -c http://repo1.maven.org/maven2/org/eclipse/jetty/

jetty-distribution/9.3.12.v20160915/jetty-distribution-9.3.12.v20160915.tar.gz‘

```
tar xzf jetty-distribution-9.3.12.v20160915.tar.gz

mv jetty-distribution-9.3.12.v20160915 jetty9

sudo mv jetty9 /opt

sudo apt install maven

git clone https://github.com/katyfelkner/cs4263-rng

cd cs4263-rng/JavaInstance/

mvn jetty:run
```

And you should be up and running!

Running the Java VM

Jetty Helloworld Webapp

Run with:

```
$ mvn jetty:run
```

or:

```
$ mvn jetty:run -Djetty.http.port=9999
```

in case your system already use 8080 (default port).

Timing

Generated using the GetRequest.java

Central Region: App Java, App Python, Instance Java, Instance Python

```
10.204.6.71: US-Central-1_App_Java @http://java-random-number.appspot.com/demo
325 223831
```

```
10.204.6.71: US-Central-1_App_Python @http://randompythongen.appspot.com/randomNumber/
149 83806
```

```
10.204.6.71: US-Central-1_Instance_Java @http://104.199.77.112:5000/
255 978982
```

```
10.204.6.71: US-Central-1_Instance_Python @http://35.238.165.97:5000/
112 119348
```

EU-WEST-1 Region: App Java, App Python, Instance Java, Instance Python

```
10.204.6.71: EU-WEST-1_App_Java @https://composed-anvil-254017.appspot.com/demo
575 855859
```

```
10.204.6.71: EU-WEST-1_App_Python @https://cs4263-test.appspot.com
291 974314
```

```
10.204.6.71: EU-WEST-1_Instance_Java @http://34.77.239.200:5000/
251 154908
```

```
10.204.6.71: EU-WEST-1_Instance_Python @http://35.238.165.97:5000/
107 929849
```