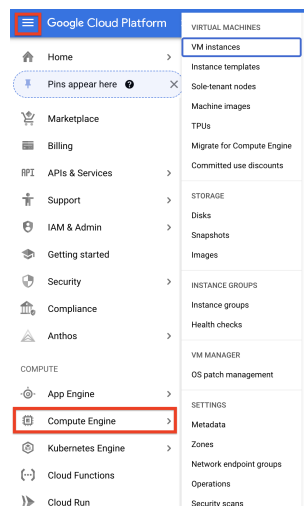


## 1. Overview

In this short tutorial, students will setup and install a **SEED VM** on the **Google Cloud Platform**.

## 2. Create a VM Instance

- (1) Log in to Google Cloud Console <https://console.cloud.google.com/>
- (2) Click on **Navigation Menu** and **select Compute Engine**, then **VM instances**.



- (3) If you are creating the instance for the first time, you will be prompted to create a project. Each instance belongs to a Google Cloud Project, which can have one or more instances. Enter your project name (e.g., **My CS458 Labs**), select Billing Account, and click on the **Create** button.
  - Or, if you already have a project, select it and continue.

**New Project**

**Project name \***  
My CS458 LABS ?

Project ID: my-cs458-labs. It cannot be changed later. [EDIT](#)

**Billing account \***  
Ethical Hacking and Penetration Test ▼

Any charges for this project will be billed to the account you select here.

**Organization \***  
iit.edu ?

Select an organization to attach it to a project. This selection can't be changed later.

**Location \***  
iit.edu [BROWSE](#)

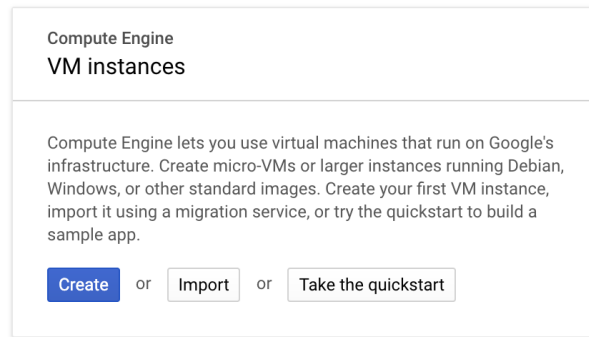
Parent organization or folder

[CREATE](#) [CANCEL](#)

- Your selected project name will appear in the top bar of the screen.

<sup>1</sup>Credit: Google Cloud & SEEDLabs

(4) Click on **Create instance** button



(5) Machine configuration: Use the following parameters.

- **e2-small** (2 vCPU and 2GB of memory) is sufficient for most SEED labs. You can easily change the machine configuration later after the machine is created.

(6) Boot disk

- On **Boot disk** section, you can choose an operating system image and boot disk type.
- Choose the **Ubuntu 20.04 LTS** operating system. For the disk size, **20GB** is more desirable, but **10GB** is sufficient (you may have to do some cleanup if the disk becomes full).

- Click on the **Select** button to confirm all the options

(7) Click on the **Create** button. And wait for the VM creation. This can take some minutes.

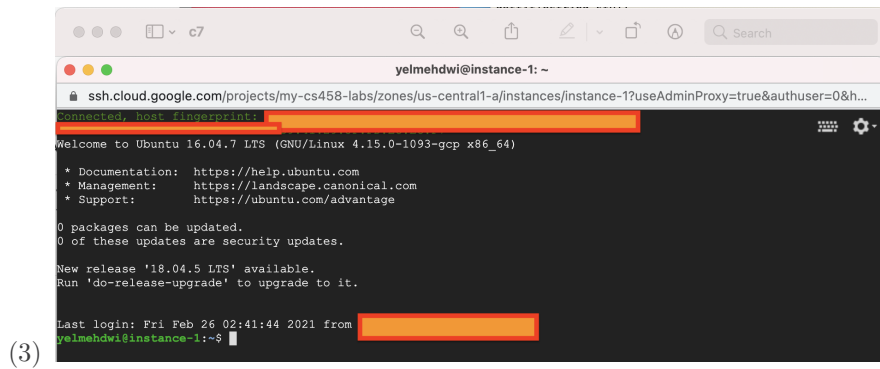
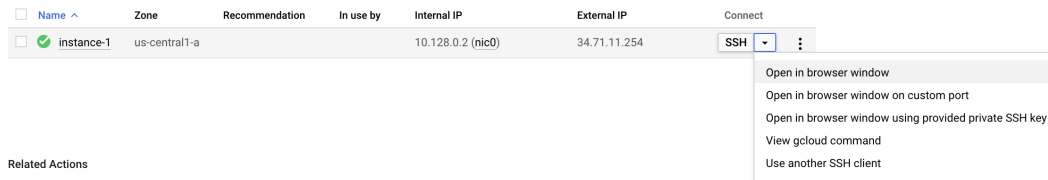
- (8) The new virtual machine will show up in your VM instances list as green. This means that it is ready to use.
- (9) Later, we need to set up firewall rules, so you can connect to the VM from the outside. There are two essential rules that we need to set, one for **SSH** (which is usually already set by the cloud), and the other is for **VNC**.

### 3. INSTALL SOFTWARE AND CONFIGURE SYSTEM

When the **Ubuntu 20.04 VM** is built, a default username with the root privilege will be created in the system. The actual name of the user is typically chosen by the cloud operator. Most cloud platforms will provide a method for you to **SSH** into this account.

**3.1. SSH from the browser.** Using the **SSH** from the browser window lets you use **SSH** to connect to a Compute Engine virtual machine (VM) instance from within the **Google Cloud Console**. You do not need to install web browser extensions or additional software to use this feature. **SSH** from the browser is an alternative to other methods of connecting to an instance.

- (1) In the Cloud Console, go to the VM instances page
- (2) In the **SSH** option, you can open an **SSH** web console to connect to the machine. Click **Open in browser window**.



- (4) Download **src-cloud.zip** from <https://seed.nyc3.cdn.digitaloceanspaces.com/src-cloud.zip> or using the following command (if copy-and-paste does not work for your **SSH** client, you may have to type the command; make sure you type the URL correctly):

```
$curl -o src-cloud.zip https://seed.nyc3.cdn.digitaloceanspaces.com/src-cloud.zip
```

```
yelmehdwi@instance-1:~$ curl -o src-cloud.zip https://seed.nyc3.cdn.digitaloceanspaces.com/src-cloud.zip
% Total % Received % Xferd Average Speed Time Time Time Current
Dload Upload Total Spent Left Speed
100 670k 100 670k 0 0 1136k 0 --:--:-- --:--:-- --:--:-- 1138k
yelmehdwi@instance-1:~$
```

- (5) In order to unzip the file, we first need to install the unzip program using the following command. After that, unzip the file.

```
sudo apt update
sudo apt -y install unzip
unzip src-cloud.zip
```

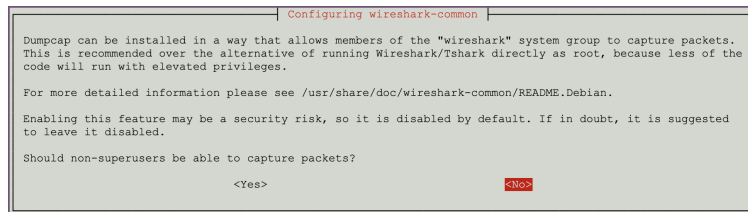
- (6) After unzipping the file, you will see a `src-cloud` folder. Enter this folder, and run the following command to install software and configure the system.

```
./install.sh
```

```
yelmehdwi@instance-1:~$ ls
src-cloud  src-cloud.zip
yelmehdwi@instance-1:~$
yelmehdwi@instance-1:~$ cd src-cloud/
yelmehdwi@instance-1:~/src-cloud$
```

- (7) **Note:** This shell script will download and install all the software needed for the `SEED labs`. The whole process will take a few minutes. Please don't leave, because you will be asked twice to make choices:

- During the installation of Wireshark, you will be asked whether non-superuser should be able to capture packets. Select `No`.



- During the installation of `xfce4`, you will be asked to choose a default display manager. Choose `LightDM`.

- (8) After the script finishes, a new account called `seed` is created. We will use this account for all the SEED labs, instead of the default one created by the cloud. We intentionally did not set a password for this account, so nobody can directly log into this account. We can switch to the seed account using the following command (if you do not use `sudo`, the OS will ask you to type the password, making it impossible to log in):

```
$sudo su seed
```

- On the cloud VM: We need to make sure that we are in the seed account. If you are still in the default account, do the following, and you will be in the seed account:

- (9) Do the following to create a folder to store your work.

```
seed@instance-1: /home/yelmehdwi/Labs/Lab02
ssh.cloud.google.com/projects/cs458-labs/zones/us-central1-a/instances/instance-1?useAdminProxy=tr...
seed@instance-1:/home/yelmehdwi$ sudo mkdir Labs
seed@instance-1:/home/yelmehdwi$ cd Labs/
seed@instance-1:/home/yelmehdwi/Labs$ sudo mkdir Lab02
seed@instance-1:/home/yelmehdwi/Labs$ cd Lab02
seed@instance-1:/home/yelmehdwi/Labs/Lab02$
```

## 4. Access the VM Using VNC

For most labs, being able to `SSH` into the VM should be sufficient. Some labs do need to access `GUI` applications on the VM, such as `Firefox` and `Wireshark`. If your network bandwidth is not too bad, being able to get a graphical desktop of the remote VM is always more desirable than getting a text terminal via `SSH`. We will use `VNC` (Virtual Network Computing) to get the remote desktop.

### 4.1. VNC vs. VPN.

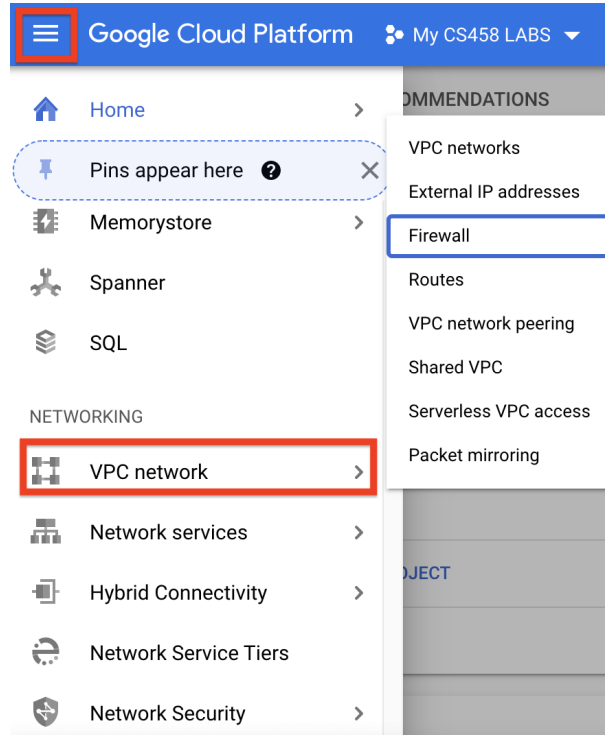
- `Virtual Network Computing (VNC)` provides remote control of a computer at some other location allowing users to operate that computer as if they were sitting in front of it.
- `Virtual Private Network (VPN)` simply connects you to a remote network, but does not provide a desktop for you to use.

In this section, we will cover how can we remote desktop to your VM instance. For this purpose, we are going to use `VNC server` and `VNC client` for remote desktop connection. `VNC client` will connect to the `VNC server` using `TCP/IP` protocol at port `5901`.

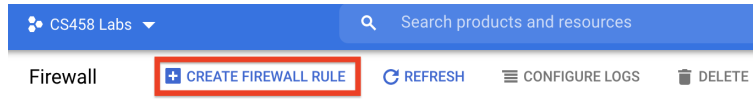
To start with, we need to create a `firewall rule` to allow `TCP/IP` connections to our VM instance.

**4.2. Creating firewall rules.** `Firewall rules` are defined at the network level, and only apply to the network where they are created; however, the name you choose for each of them must be unique to the project. We will explain how to create a `firewall rule` via console.

- (1) Log in to `Google Cloud Console` via <https://console.cloud.google.com/>
- (2) `Firewall rules` are available under the `VPC network` in the `networking` section on the left side menu.



- (3) Click **Create Firewall Rule**.



- (4) Enter a **Name** for the firewall rule. This name must be unique for the project and only in lowercase, numbers, hyphens and no space is allowed. See Figure 1.
- (5) Specify the **Description**. This is optional but good to enter something meaningful, so you remember in future
- (6) Specify the **Network** for the firewall rule.
- If you haven't created any **VPC** then you will see only default and leave it as it is. However, if you have multiple **VPC** then select the network where you want to apply the firewall rules.
- (7) Specify the **Priority** of the rule. It starts with **1000**.
- The lower the number, the higher the priority.
  - In most cases, you want to keep all critical services (**HTTP**, **HTTPS**, etc.) with priority **1000**.
- (8) For the **Direction of traffic**, choose **ingress** (incoming) or **egress** (outgoing).
- (9) For the **Action on match**, choose **allow** or **deny**.
- (10) Specify the **Targets** of the rule where you want to apply the rules.
- If you want the rule to apply to all instances in the network, choose **All instances in the network**.
  - If you want the rule to apply to select instances by network (target) tags, choose **Specified target tags**, then type the tags to which the rule should apply into the **Target tags** field.
  - If you want the rule to apply to select instances by associated service account, choose **Specified service account**, indicate whether the service account is in the current project or another one under **Service account scope**, and choose or type the service account name in the **Target service account** field.
- (11) For an **ingress** rule, specify the **Source filter**. **Source filter** is a source which will be validated to either allow or deny. You can filter by **IP ranges**, **subnetworks**, **source tags**, and **service accounts**.
- Choose **IP ranges** and type the **CIDR blocks** into the **Source IP ranges** field to define the source for incoming traffic by **IP address ranges**. Use **0.0.0.0/0** for a source from any network.
- (12) For an **egress** rule, specify the **Destination filter**
- Choose **IP ranges** and type the **CIDR blocks** into the **Destination IP ranges** field to define the destination for outgoing traffic by **IP address ranges**. Use **0.0.0.0/0** to mean everywhere.
- (13) Define the **Protocols and ports** to which the rule applies
- Select **Allow all** or **Deny all**, depending on the action, to have the rule apply to all protocols and destination ports.
  - Define **specific protocols** and **destination ports**:
    - Select **tcp** to include the **TCP** protocol and destination ports. Enter all or a comma-delimited list of destination ports, such as **20-22**, **80**, **8080**.
    - Select **udp** to include the **UDP** protocol and destination ports. Enter all or a comma-delimited list of destination ports, such as **67-69**, **123**.
    - Select **Other protocols** to include protocols such as **icmp** or **sctp**
- (14) Click **Create**.

[←](#) 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 \*

default-allow-vnc

?

Lowercase letters, numbers, hyphens allowed

Description

remote vnc

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 \*

1000

CHECK PRIORITY OF OTHER FIREWALL RULES

?

Priority can be 0 - 65535

Direction of traffic

?

☒ Ingress

☐ Egress

Action on match

?

☒ Allow

☐ Deny

Targets

All instances in the network

▼

?

Source filter

IP ranges

▼

?

Source IP ranges \*

0.0.0.0/0

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 :

5901-5910

☐ udp :

all

☐ Other protocols

protocols, comma separated, e.g. ah, sctp

[✓ DISABLE RULE](#)

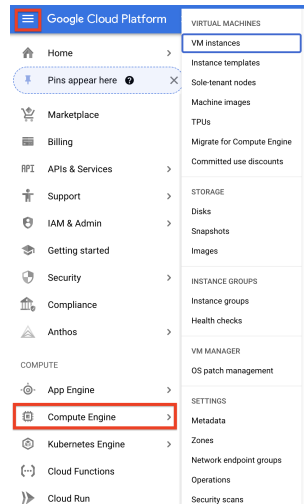
CREATE

CANCEL

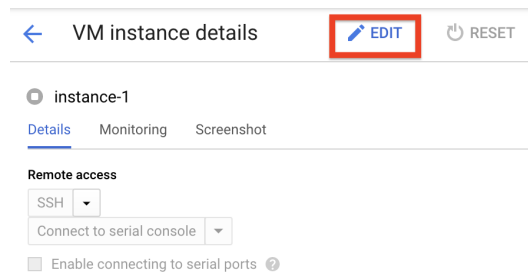
FIGURE 1. Create a firewall rule

4.3. **Apply the rule to VM instance.** Next, we need to apply the new rule to VM instance. To view all of the firewall rules that apply to a specific network interface of a VM instance:

- (1) Go to the **VM instances** page in the **Google Cloud Console** and find the instance to view.



- (2) Navigate to the **VM instance details** screen (click on the VM instance) and click **Edit**.



- (3) In the **Network tags** section, add the **Target tag** you have entered when created the rule. In this case, **default-allow-vcn**

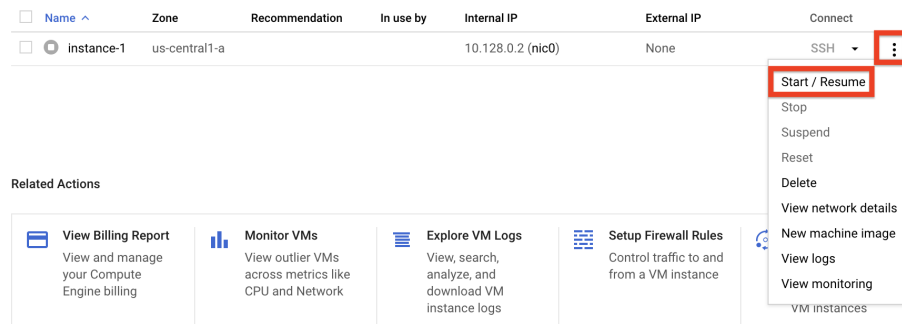


- (4) Click **Save** to make the changes take effect.



#### 4.4. To test The TCP connection to your VM instance.

- (1) Start the VM instance. In the instance's more actions menu, select **Start/Resume**.



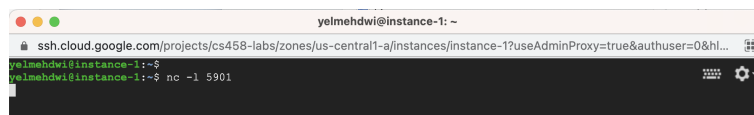
- (2) In the **SSH** option, you can open an **SSH web console** to connect to the machine.



- (3) Click **Open in browser window**.

- (4) To test the **TCP** connection, we will use **Netcat** (or **nc**). **Netcat** ( **nc** ) is a command-line utility/computer networking utility for reading from and writing to network connections using **TCP** or **UDP** protocols. It is one of the most powerful tools in the network and system administrators arsenal, and it as considered as a **Swiss army knife of networking tools**. Type

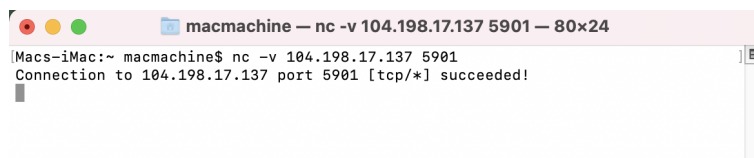
```
$ nc -l 5901
```



- (5) On your computer, open a terminal and type

```
$ nc -v 104.198.17.137 5901
```

- Your VM instance external IP (in my case, it was 104.198.17.137) may be different. Every time you start you VM, you'll get a new external IP address.



- If we got: **Connection to 104.198.17.137 port 5901 [tcp/\*] succeeded!**, then we manage successfully to connect via **TCP** to our **VM**.

4.5. **Install VNC Server and Viewer.** Next, you need to install the **VNC Client** on your local machine and **VNC Server** on the cloud VM.

4.5.1. *On the cloud VM.*

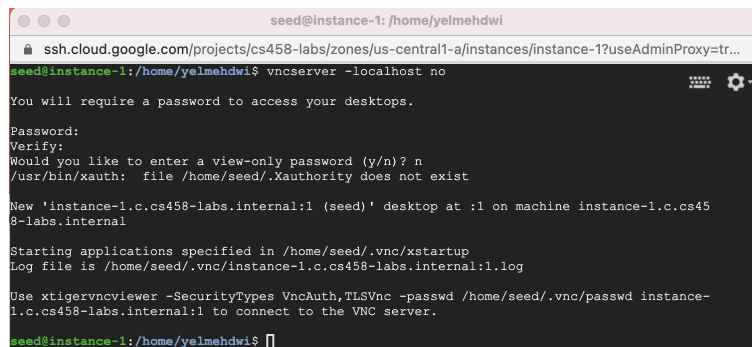
- (1) We need to make sure that we are in the seed account. If you are still in the default account, do the following, and you will be in the seed account:

```
$ sudo su seed
```

- (2) The pre installation script has already installed the **TigerVNC** server program on the VM. You need to start the server.

```
$ vncserver -localhost no
```

- By default, TigerVNC server only listens to **localhost/127.0.0.1**.
- The purpose of the **-localhost no** option means accepting access from the outside.
- When we first start the **vncserver**, we will be asked to provide a password. Make sure this password is strong enough.

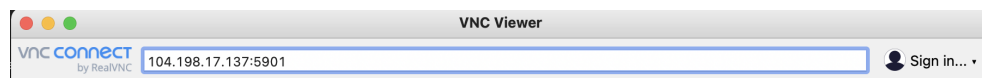


```
seed@instance-1: /home/yelmehdwi
ssh.cloud.google.com/projects/cs458-labs/zones/us-central1-a/instances/instance-1?useAdminProxy=tr...
seed@instance-1: /home/yelmehdwi$ vncserver -localhost no
You will require a password to access your desktops.
Password:
Verify:
Would you like to enter a view-only password (y/n)? n
/usr/bin/xauth: file /home/seed/.Xauthority does not exist
New 'instance-1.c.cs458-labs.internal:1 (seed)' desktop at :1 on machine instance-1.c.cs458-labs.internal
Starting applications specified in /home/seed/.vnc/xstartup
Log file is /home/seed/.vnc/instance-1.c.cs458-labs.internal:1.log
Use xtigervncviewer -SecurityTypes VncAuth,TLSVnc -passwd /home/seed/.vnc/passwd instance-1.c.cs458-labs.internal:1 to connect to the VNC server.
seed@instance-1: /home/yelmehdwi$
```

- Moreover, **VNC** communication itself is not encrypted, so you should not send anything personal.
- If you do want to secure it, you can run an **SSH tunnel** or **VPN tunnel** to protect the **VNC** communication.

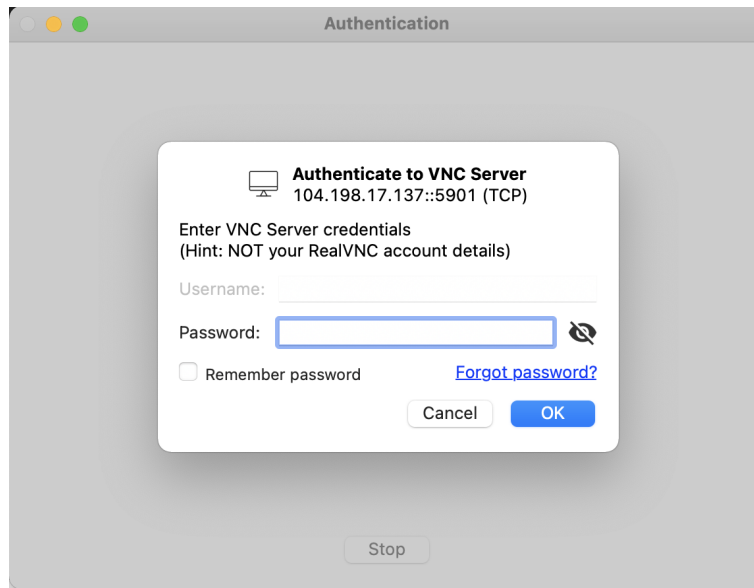
4.5.2. *On your computer.*

- (1) You need to have a **VNC viewer** installed on your computer, such as **TigerVNC** (<https://tigervnc.org/>), and **RealVNC** (<https://www.realvnc.com/en/connect/download/viewer/>).
- (2) If you prefer other **VNC viewers**, it is fine. Most of them are compatible with one another.
- (3) Some users have reported that **TigerVNC** have issues on **macOS**, but **RealVNC** has no problem.
- (4) Start your **VNC viewer** program, and type the **IP address** of the VM, along with the **port number**, such as **104.198.17.137:5901**.

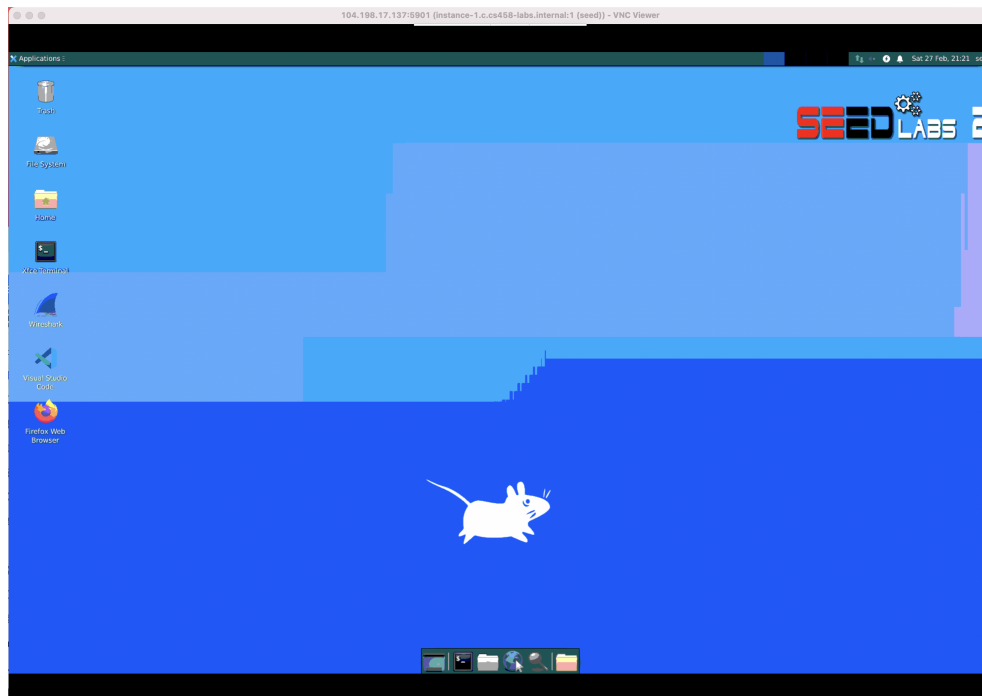


- Most cloud VMs have two **IP** addresses; make sure you use the **external IP address**, not the internal one.

- (5) You will be prompted for password, which is the one you typed when you first run the `VNC server`.



- (6) If everything is done correctly, you will see the desktop of your remote VM.

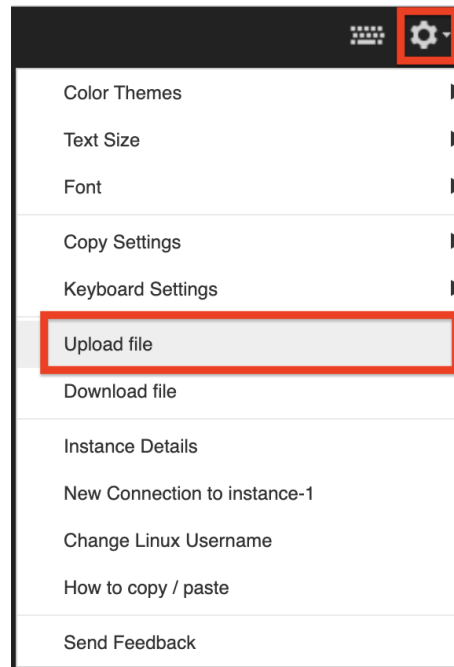


- (7) **Note:** To run `VNC`, you need to have reasonable bandwidth. If your `VNC` performance is bad, you should switch to `SSH`. You can get by with many of the `SEED labs` using just terminals. Most cloud platforms provide a default browser-based `SSH client`. `Google cloud's SSH client` even allows you to upload and download files, which is very convenient.

## 5. TRANSFERRING FILES USING SSH IN THE BROWSER

Sometime, you need to transfer files to Compute Engine instances. Different options are available depending on your workstation OS and the target instance OS. In this section, we will show you how to transfer files using **SSH in the browser**.

- (1) Follow **Steps 1** and **2** from **Section 3.1**
- (2) After the connection is established, click the **gear icon** in the upper right of the **SSH from the Browser window** and select **Upload file**. Alternatively, select **Download file** to download a file from the instance.



- (3) The transfer dialog opens. Specify which file you want to transfer.
  - If you uploaded a file, the file is in your user's **/home/\$USER** directory.
  - If you downloaded a file, the file is in the default download folder on your local workstation.

## 6. NOTES ON COST

Unless you have a special deal with cloud company, you will be charged for using the cloud VM. Please keep an eye on your bill, because sometimes, there are costs that you may not be aware of, such as bandwidth cost, storage cost, etc. Understanding where your expense is can help you reduce it. Moreover, to avoid wasting money, remember to suspend your VMs if you are not working on them. Although a suspended VM still incurs storage cost (usually very small), it does not incur any computing costs. You can easily resume them when you are ready to continue your work.