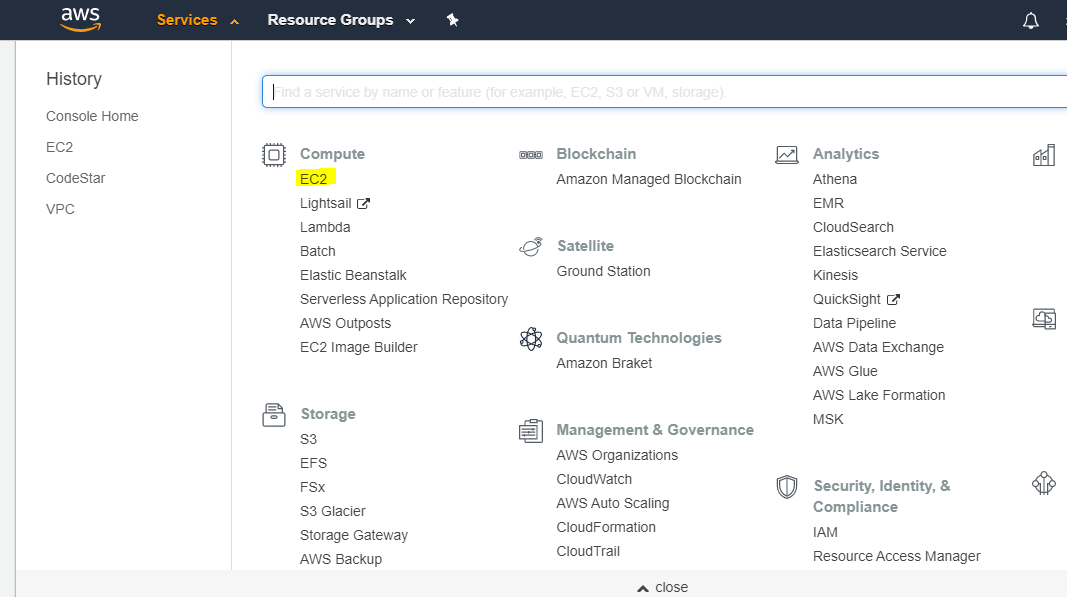
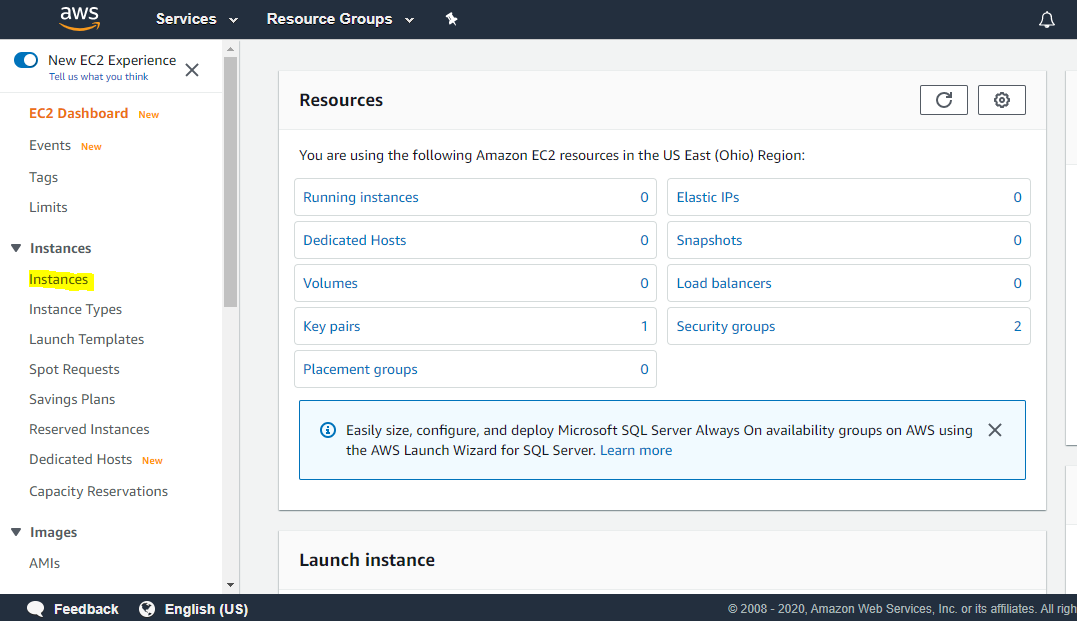
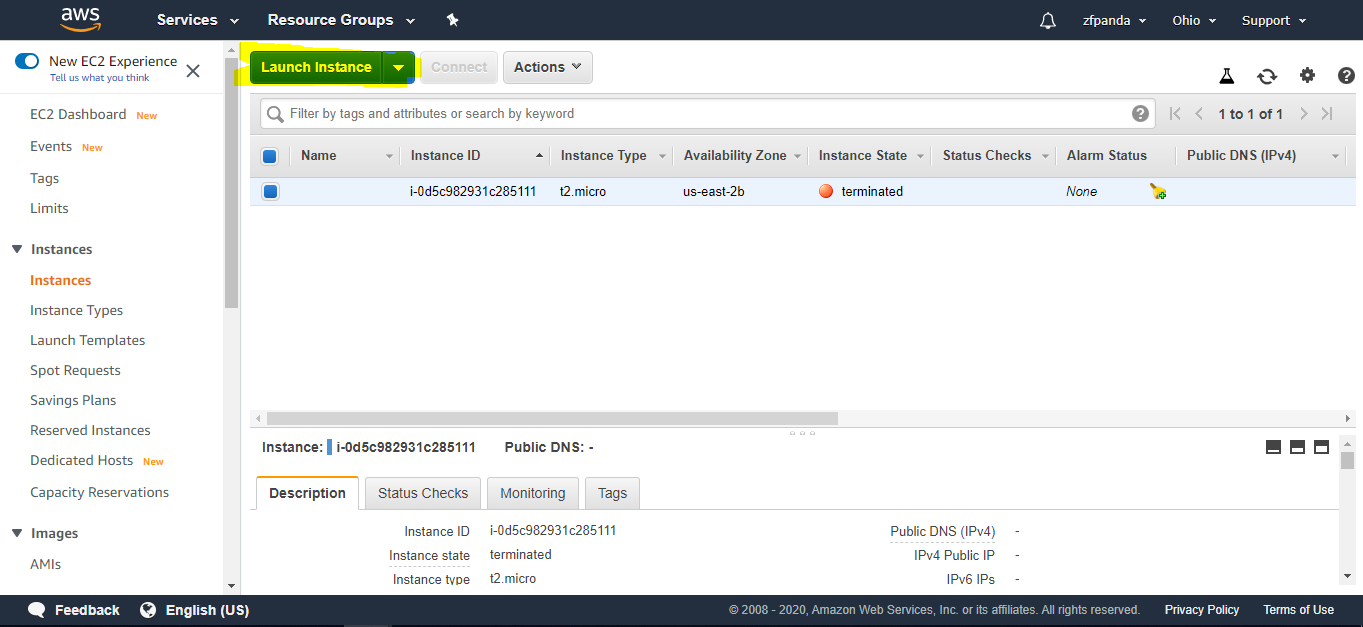
Select EC2 (cloud server)

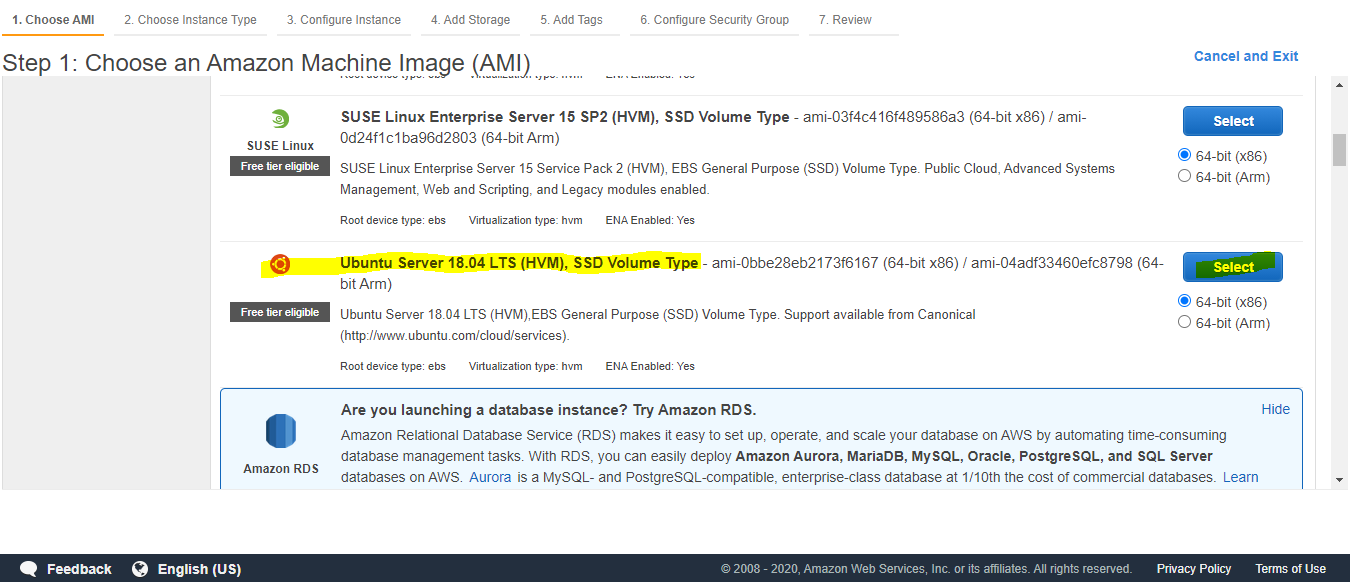


Select Instance( cloud server)

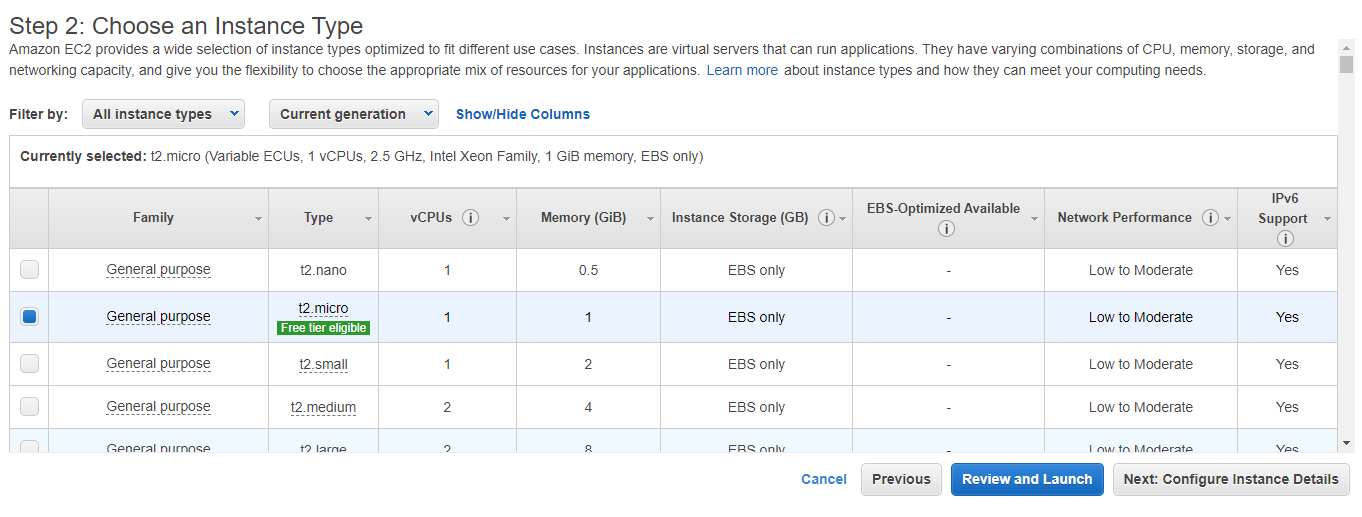


Launch Instance: create cloud server 

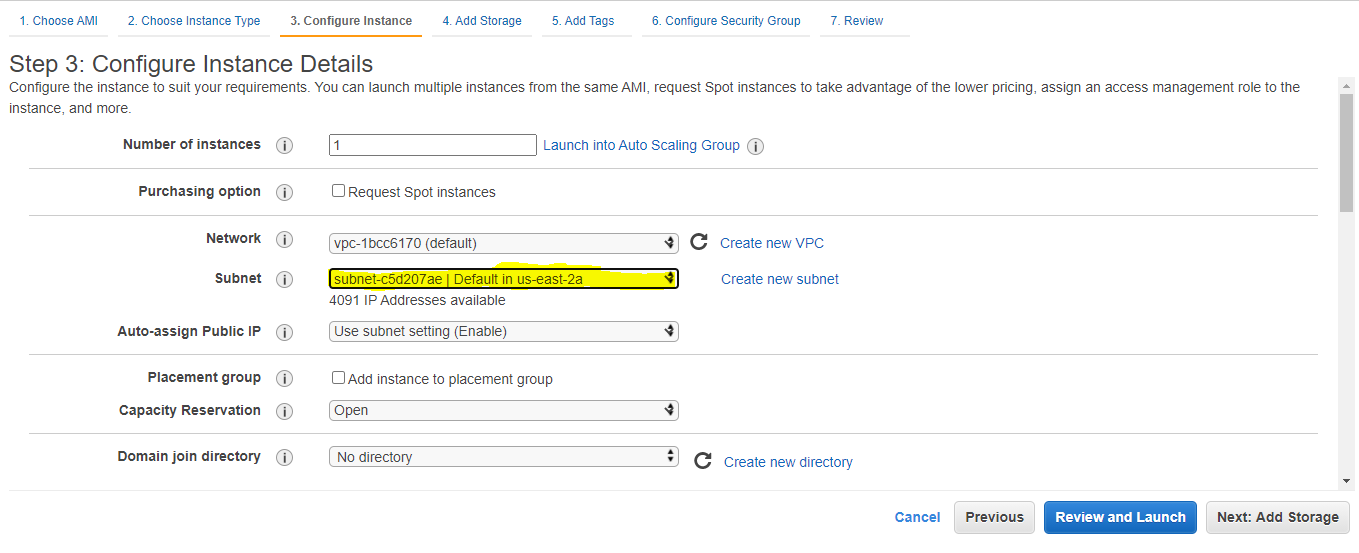
Select Ubuntu Server



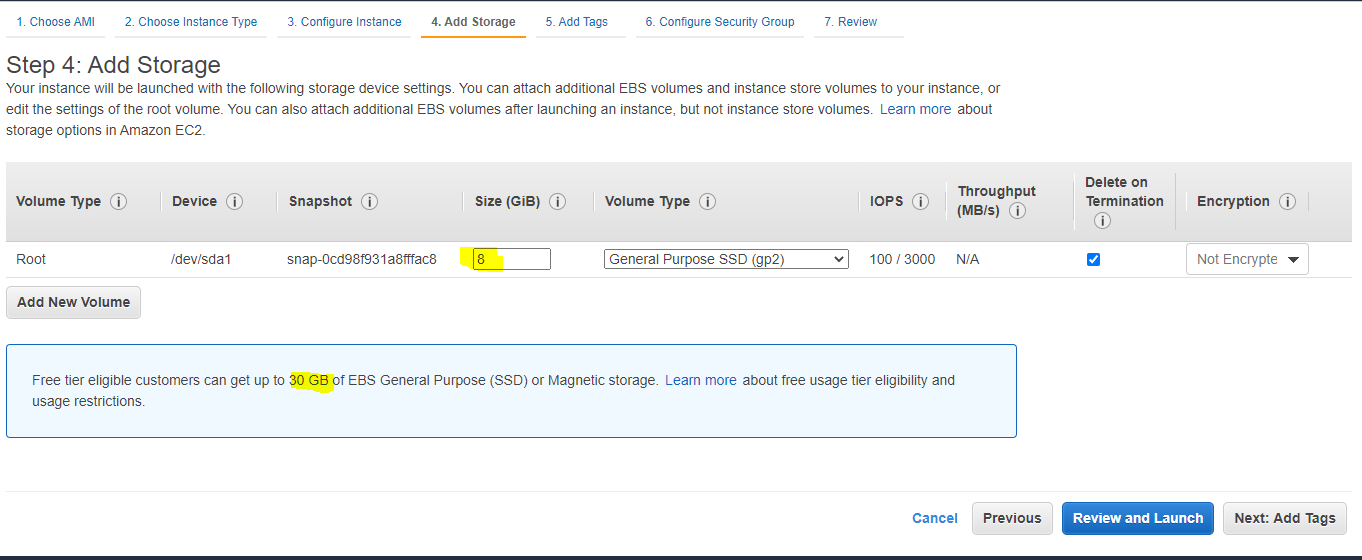
The Free tier eligible( can not allocate the vCpu and Memory)

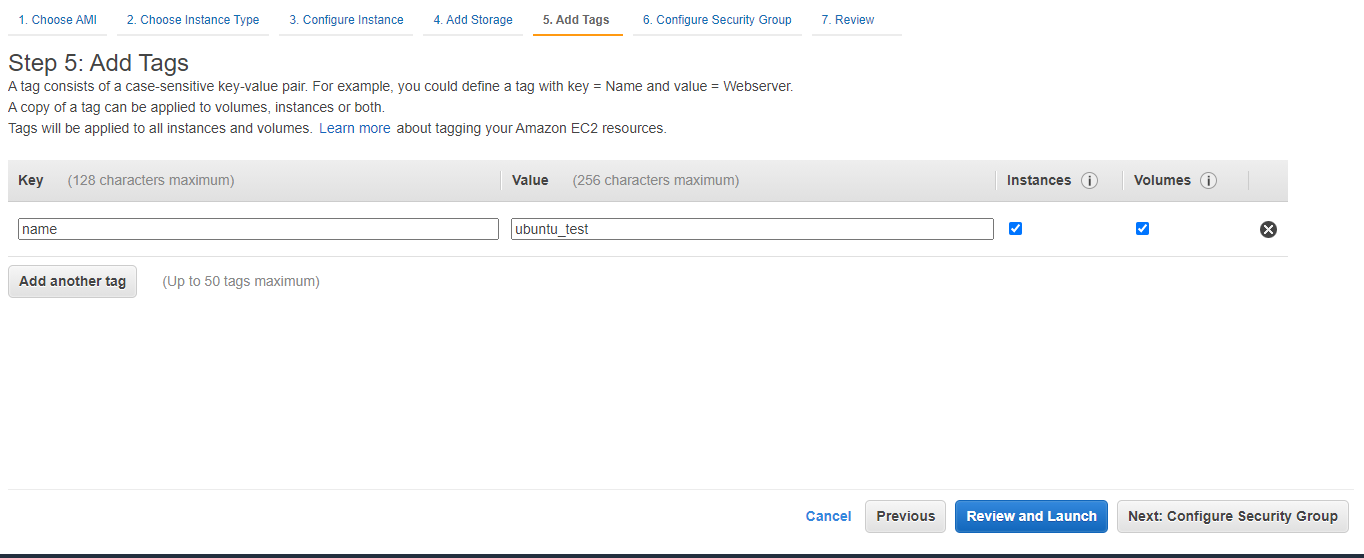


Take note of subnet, may need to create a virtual hard disk. If hard disk is created in another region, then it will not be connected. The rest fields no need to change.



default size is 8 GB, total free size is 30GB

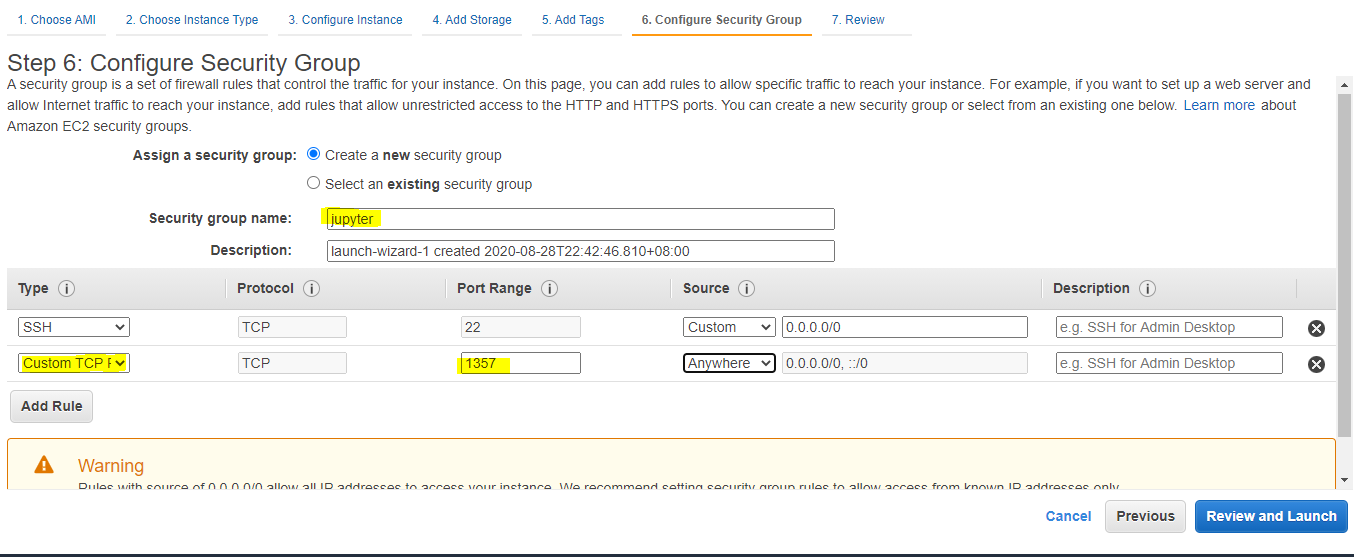


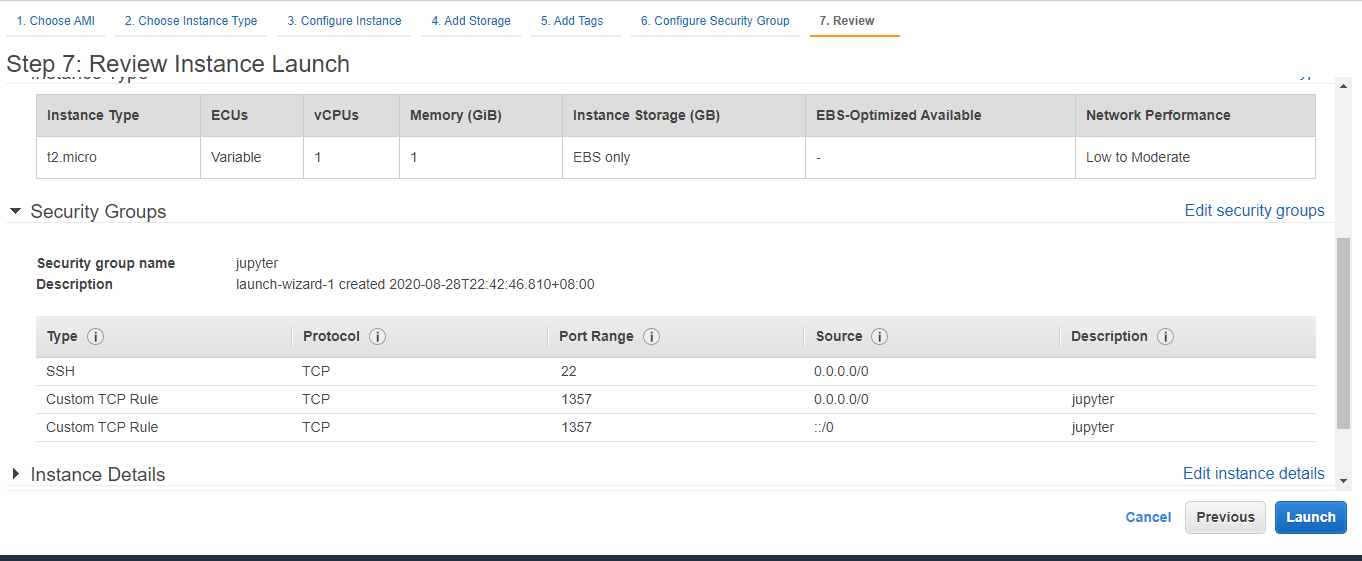


The purpose of Tag is to manage Virtual Machine (ubuntu, linux…). When there are many VMs, the tag will help to differentiate the VMs. It is optional if there is few VMs.

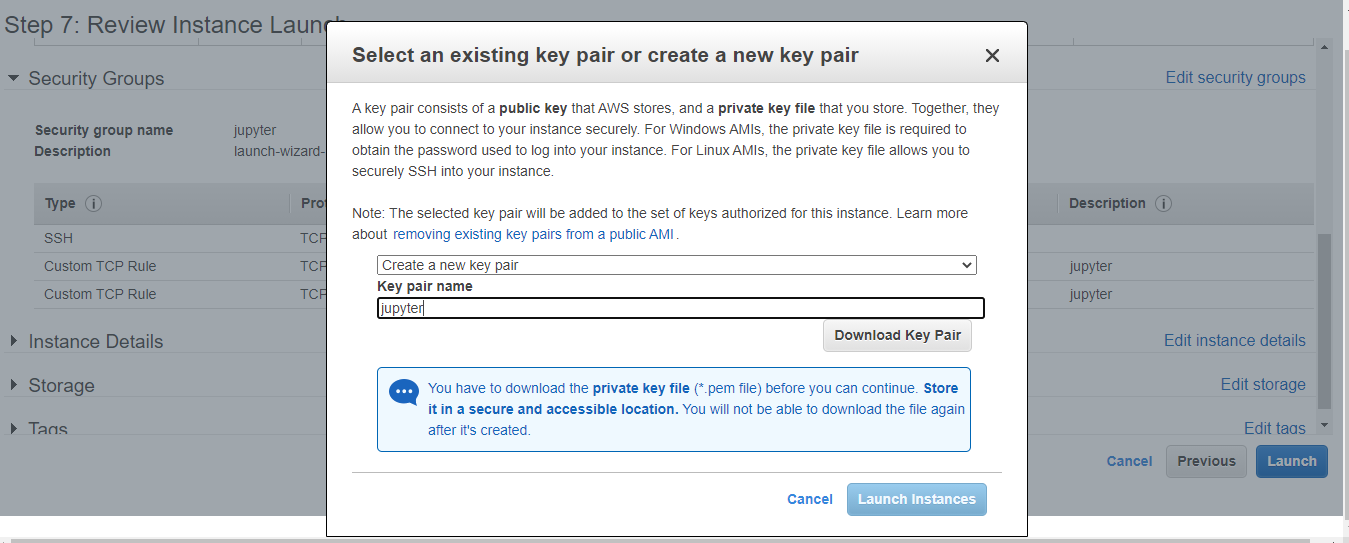
SSH will turn on port 22. 0.0.0.0/0 means any ip address can visit port 22.

We add a rule, custom TCP, we use port 1357 for test, later need to tune jupyter port to match this port(1357) as the default port for jupyter is 8888. The default port 8888 can be easily scanned by others.

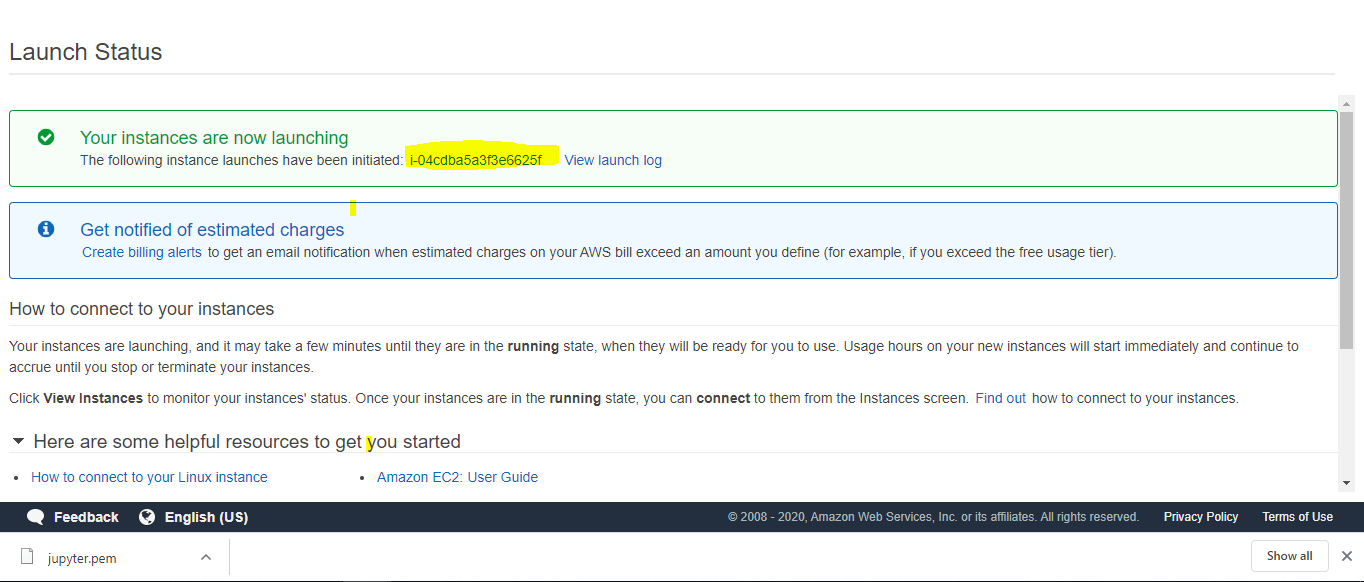


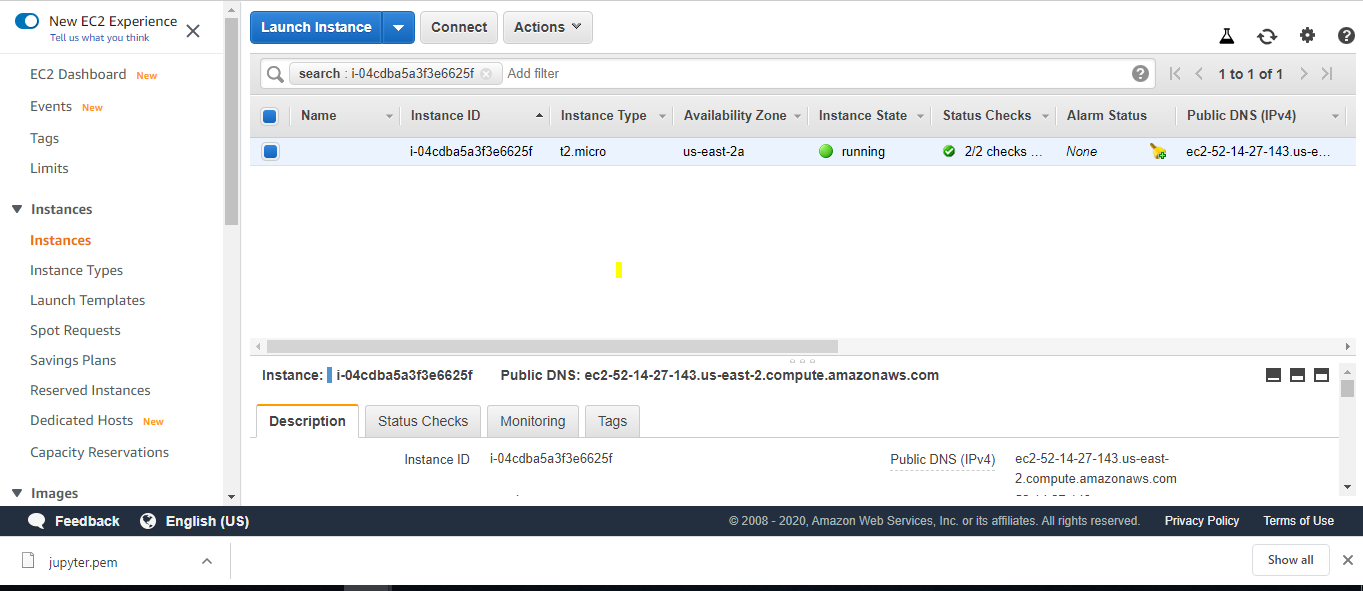


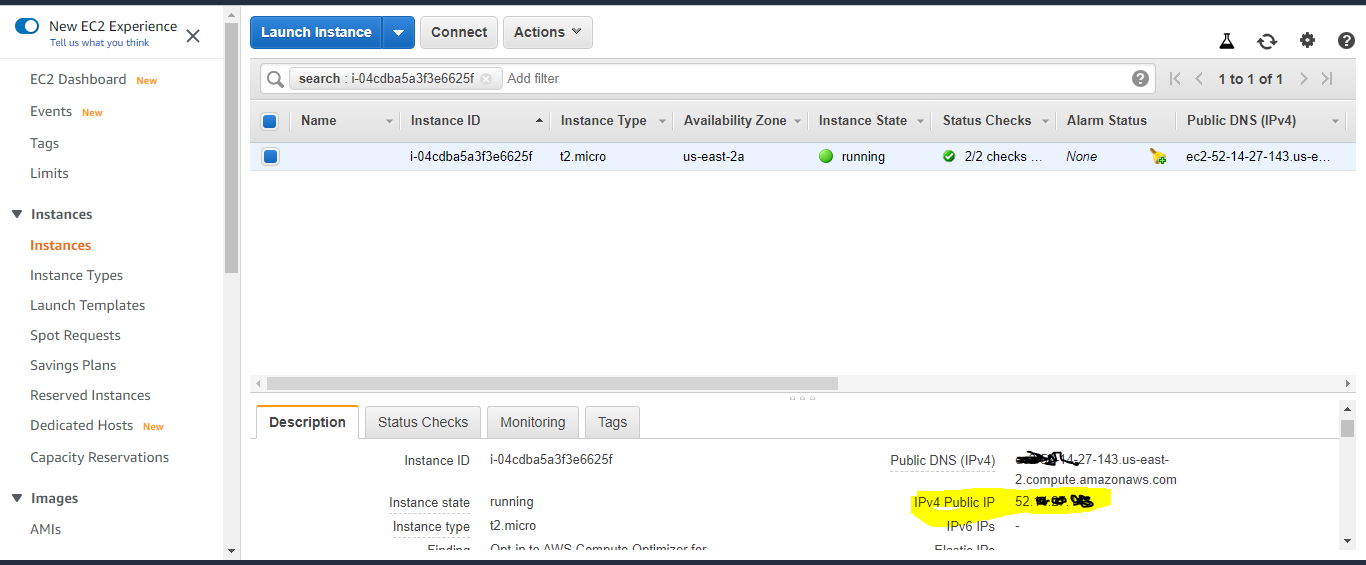
When login by SSH, it cannot use normal password, need to use key pair. download the key pair and keep it. Then Launch instance.



When click instance ID (highlight part), will link to instance page, show details of this instance





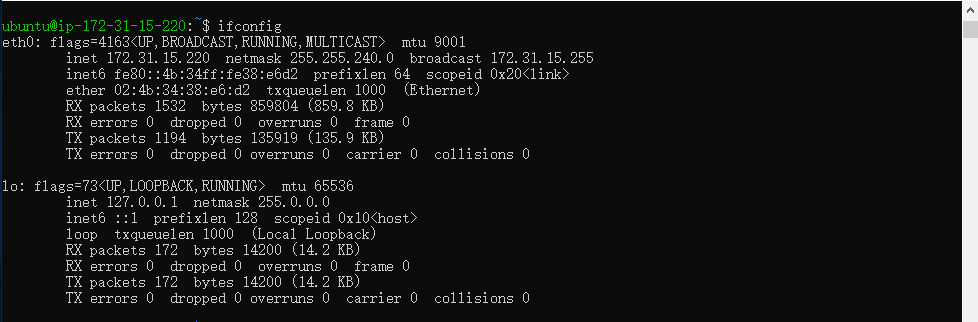


Now can use SSH to remote connect Ubuntu:

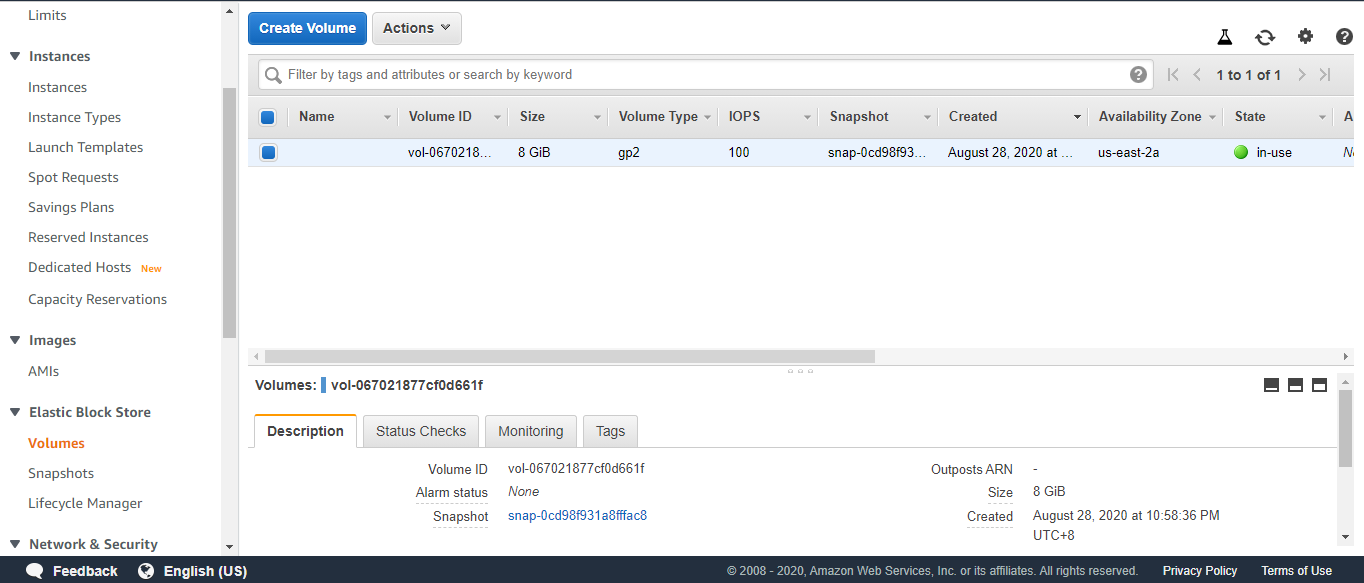
Put the pem file under the current path C:Users\XXXX\

ssh -i jupyter.pem [ubuntu@52.xx.xx.](mailto:ubuntu@52.xx.xx.143)xx (key in your own public IP address)

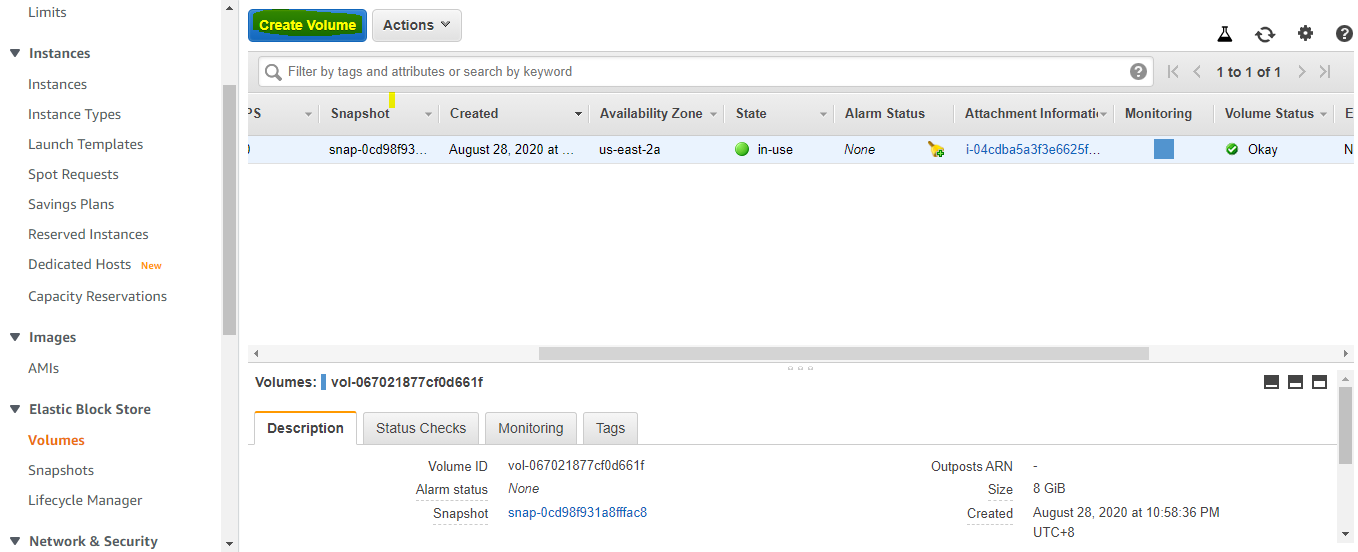




Now select Volumes ( create hard disk)

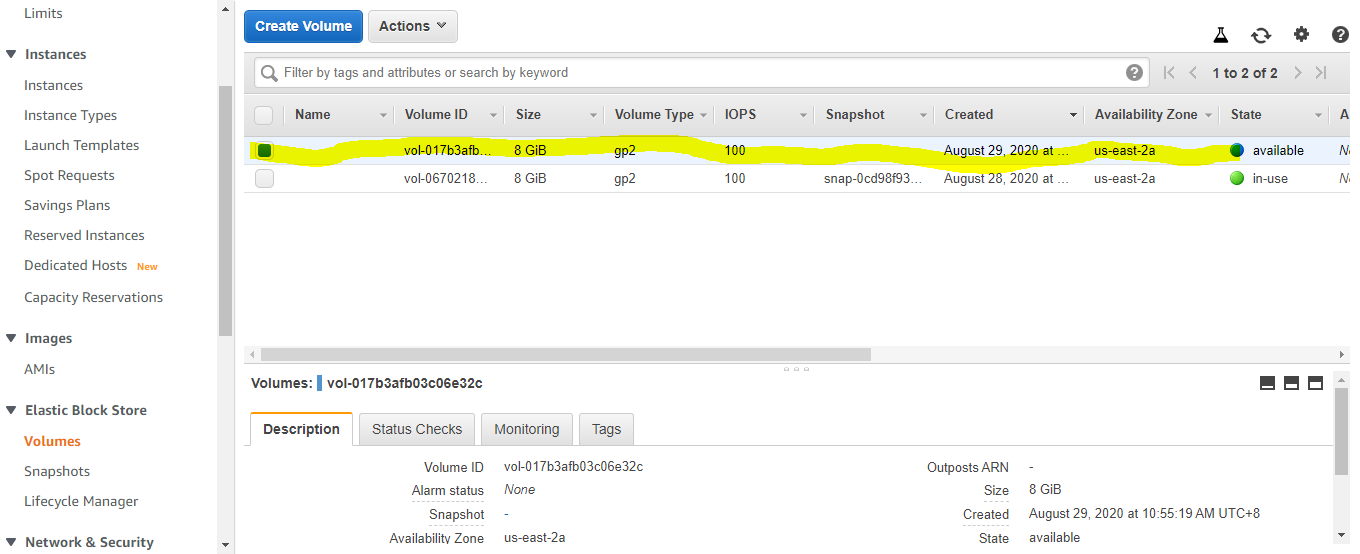


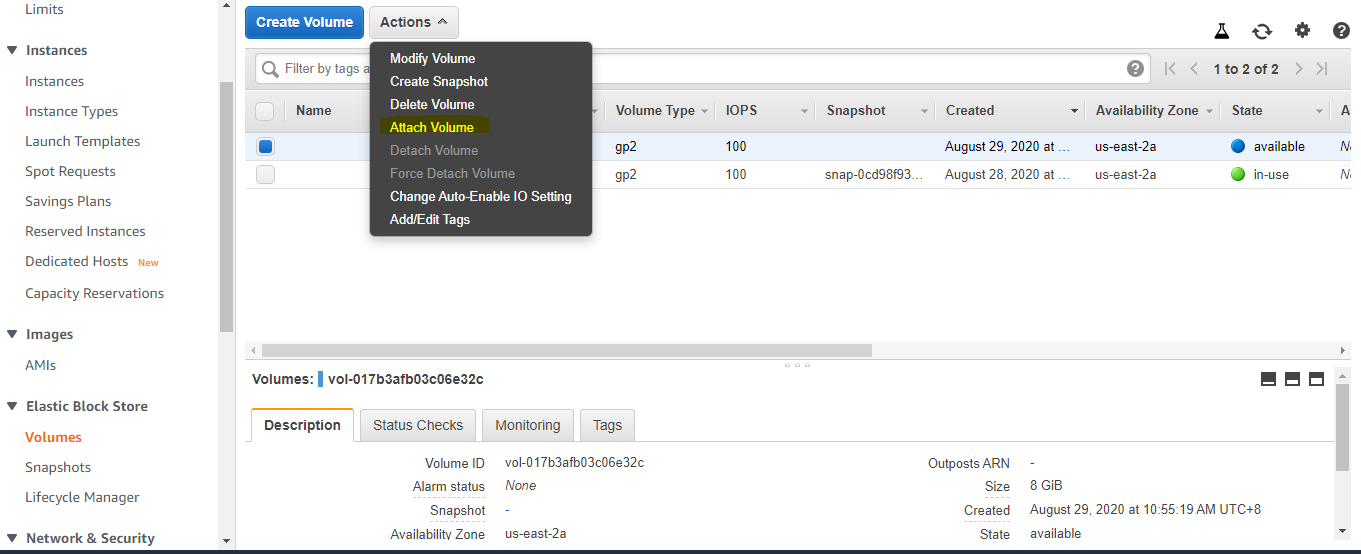
Create volume

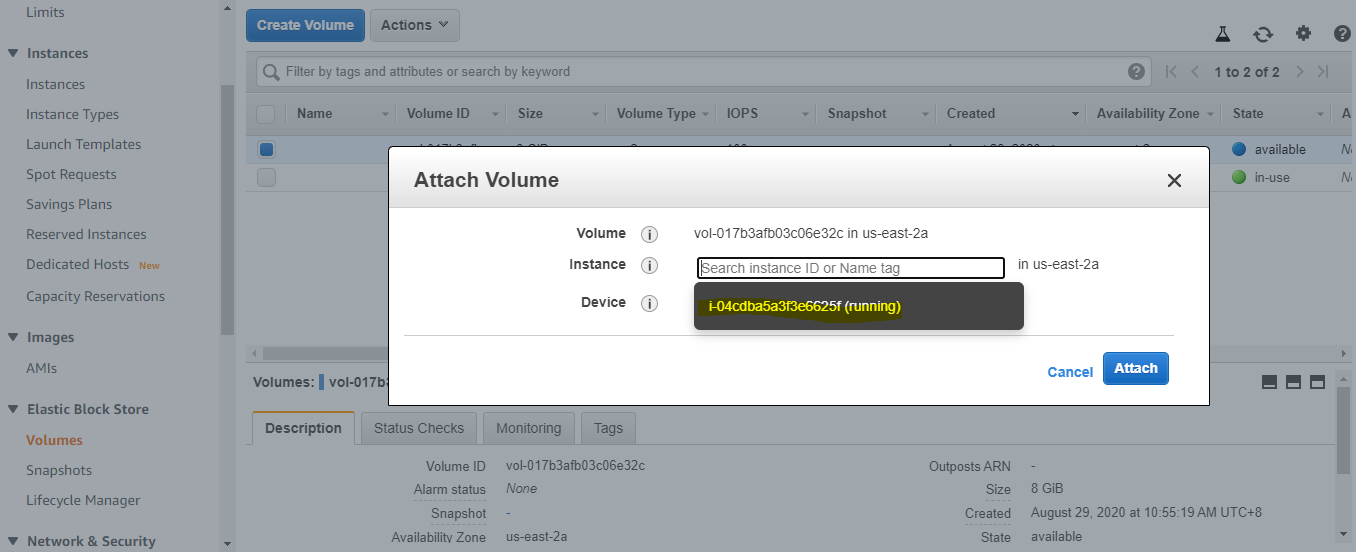


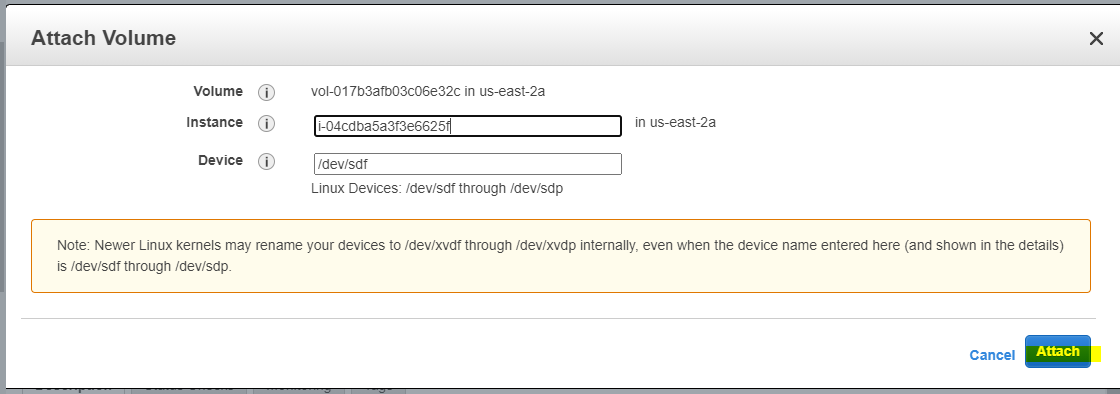


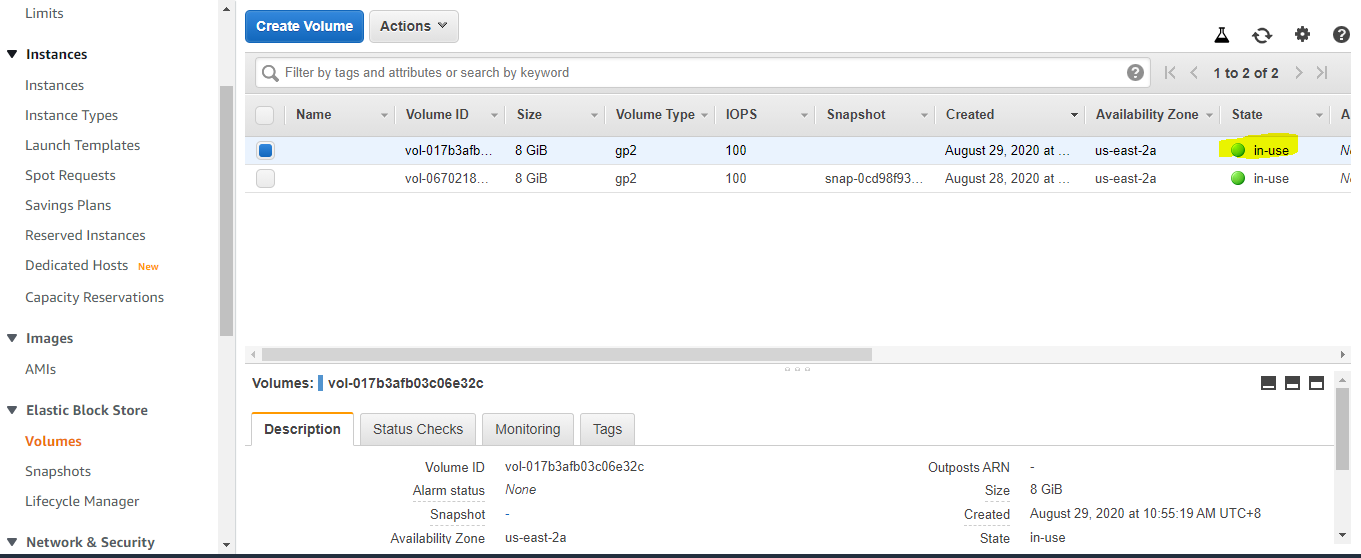
Tick the created volume, under action, select attach Volume





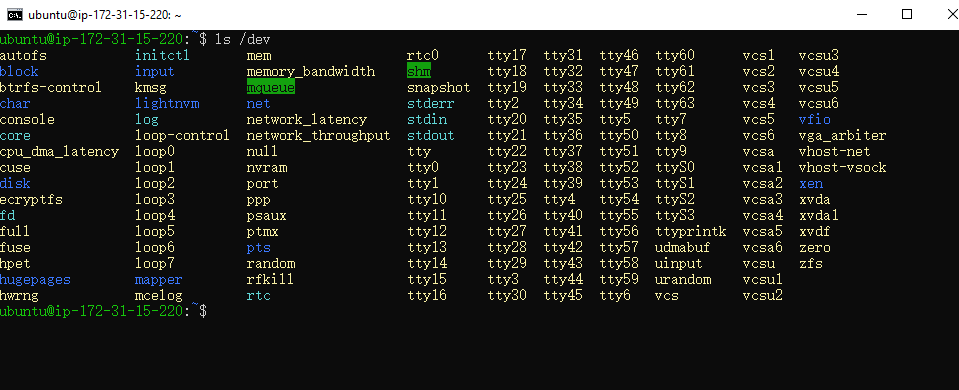






The state of volume become “in-use”, Now the Volume(virtual hard disk is attached to the cloud server(ubuntu)

Now in the cmd, ls /dev, will shows devices. And just now the created hard disk “xvdf” is displayed, this is similar like thumb drive, need mount to the system

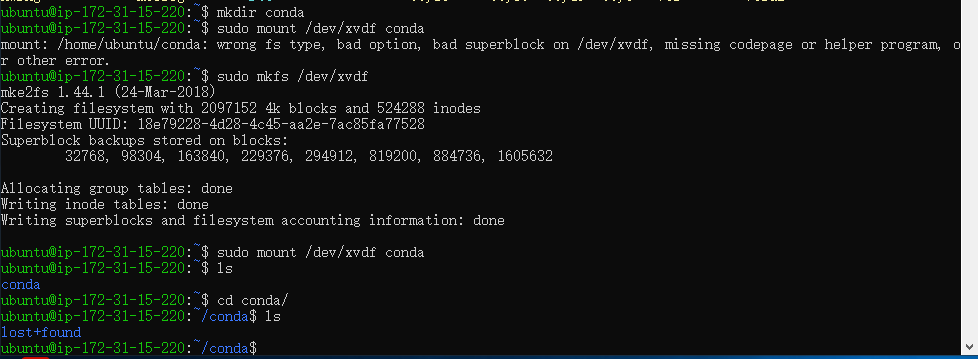


mkdir conda

sudo mount /dev/xvdf conda

sudo mkfs /dev/xvdf --this is to format “xvdf” device before use

sudo mount /dev/xvdf conda



Use wget https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86\_64.sh

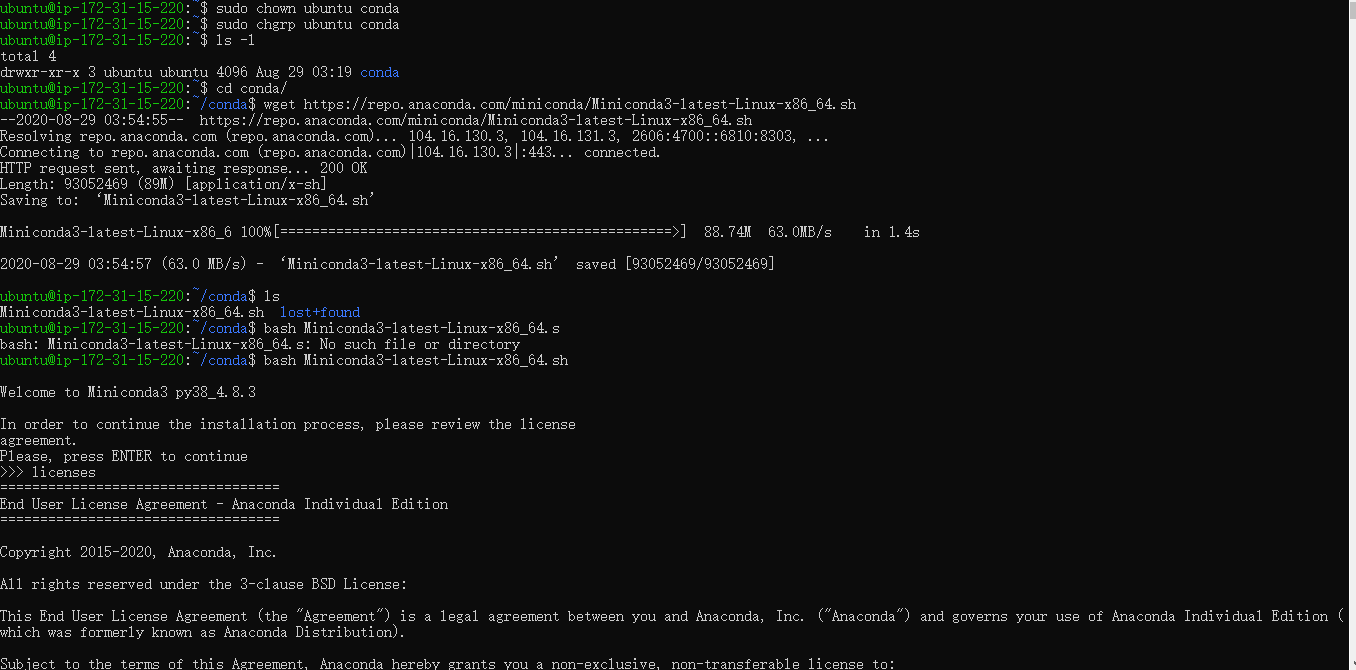


sudo chown ubuntu conda

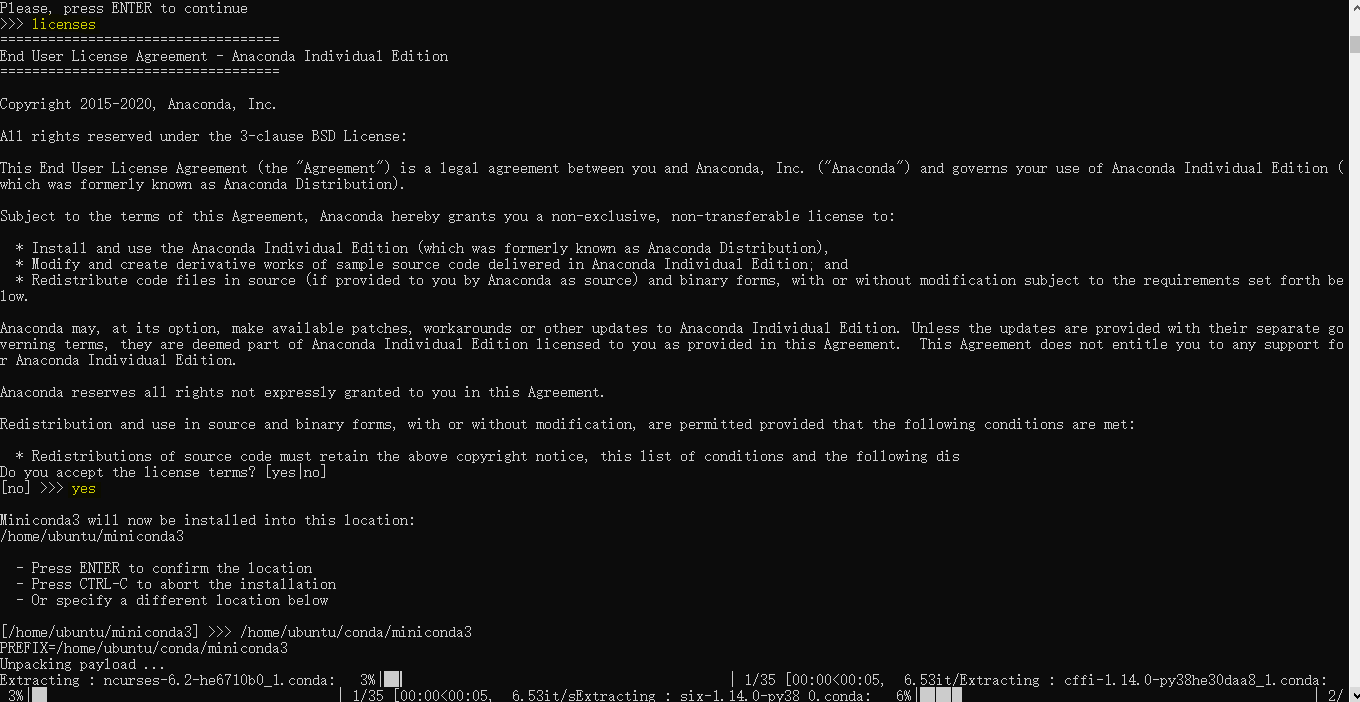
sudo chgrp ubuntu conda

Use wget <https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86_64.sh>

Using bash to install conda



Check licenses, then type yes



It will prompt message to ask where to install the miniconda:

/home/ubuntu/conda/miniconda3

Do you wish the installer to initialize Miniconda3 in your /home/ubuntu/.bashrc[yes|no]

Yes

Once the miniconda3 is installed, we need to run the “initialize shell script”. There are 2 ways. First is just quit conda and login in again. 2rd is to run “ubuntu@ip-172-31-15-220:-$ ~conda$ . ~/.bashrc”

Once it successfully installed,

conda upgrade conda

now create virtual environment

conda create -n jupyter python=3.8 -y

Once the environment:

conda activate jupyter

the (jupyter) means it is in jupyter environment now

pip install jupyter

mkdir jupyter\_hom

cd jupyter\_home/

jupyter notebook

y

ubuntu@ip-172-31-15-220:-$ ~conda/jupyter\_home$ python

>>> from IPython.lib import passwd

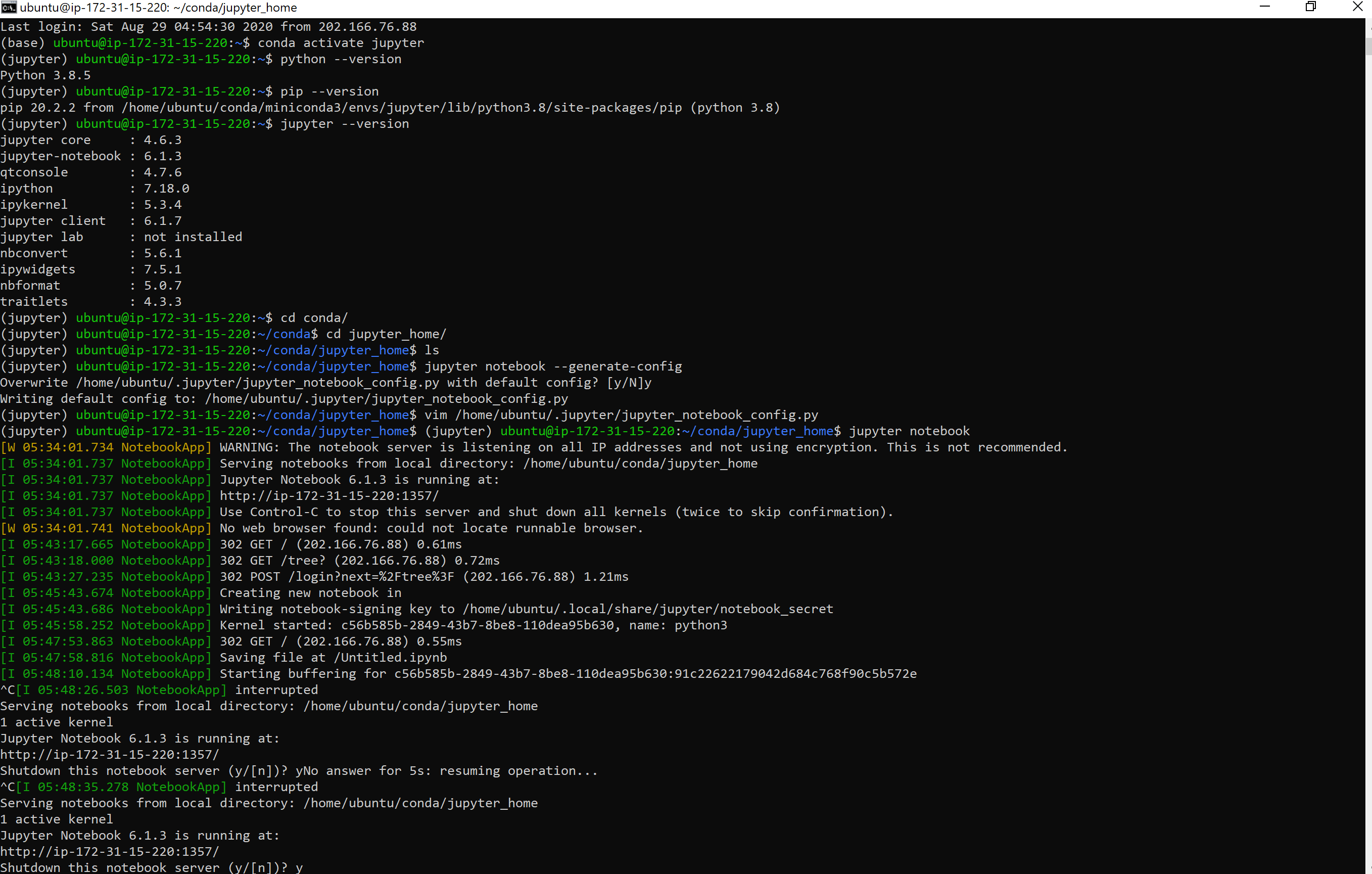
>>>passwd()

Enter password:

Verify password:

'sha1:665bf9279840:c7b38e4315e0d2e740594a081e563746f5123e5e' this will used to config jupyter config file

Jupyter notebook –generate-config



vim jupyter\_notebook\_config.py

(press i enter edit mode; press esc, :wq then save and quit)

c.NotebookApp.allow\_remote\_access = True

c.NotebookApp.ip = "\*"

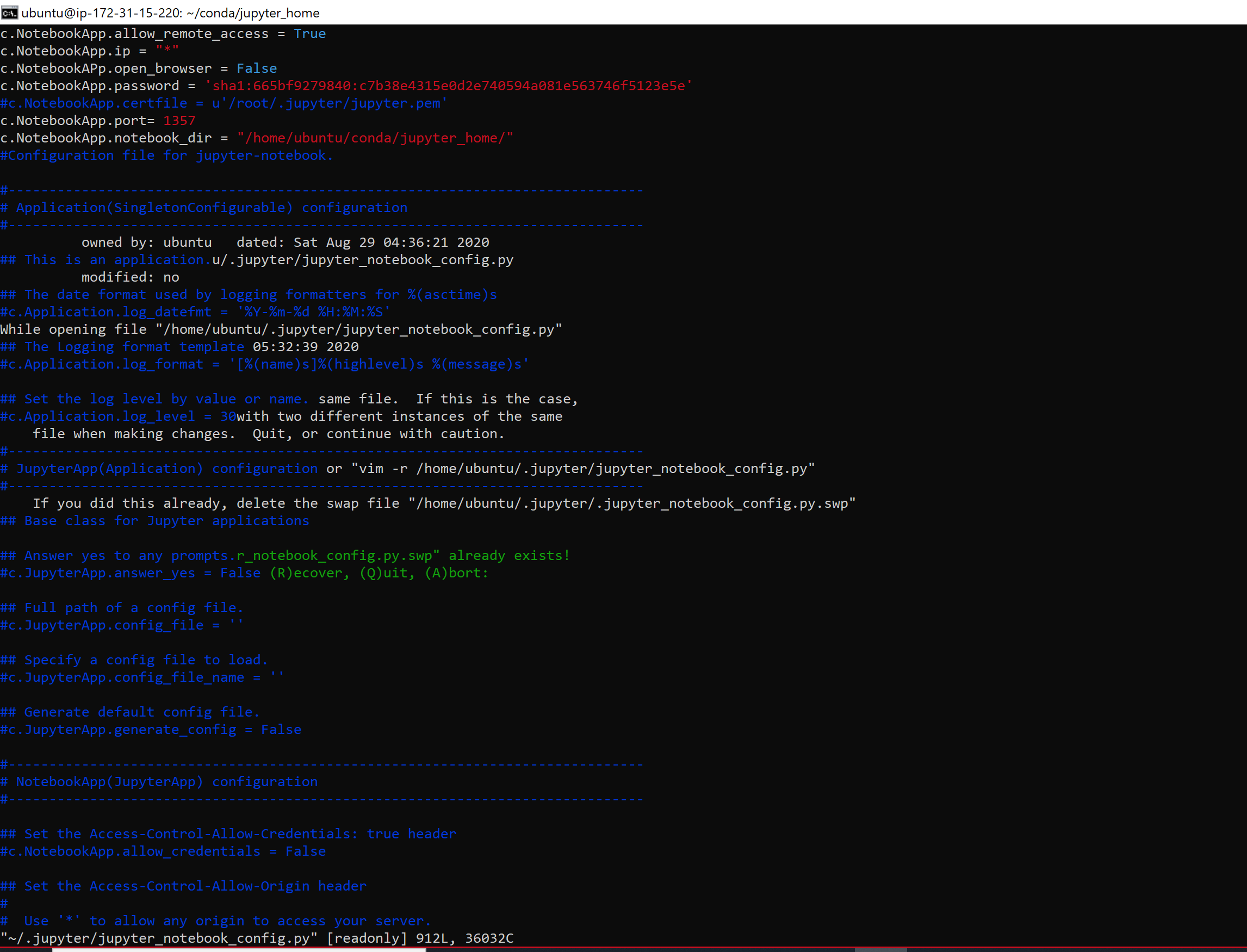
c.NotebookAPp.open\_browser = False

c.NotebookApp.password = 'sha1:665bf9279840:c7b38e4315e0d2e740594a081e563746f5123e5e'

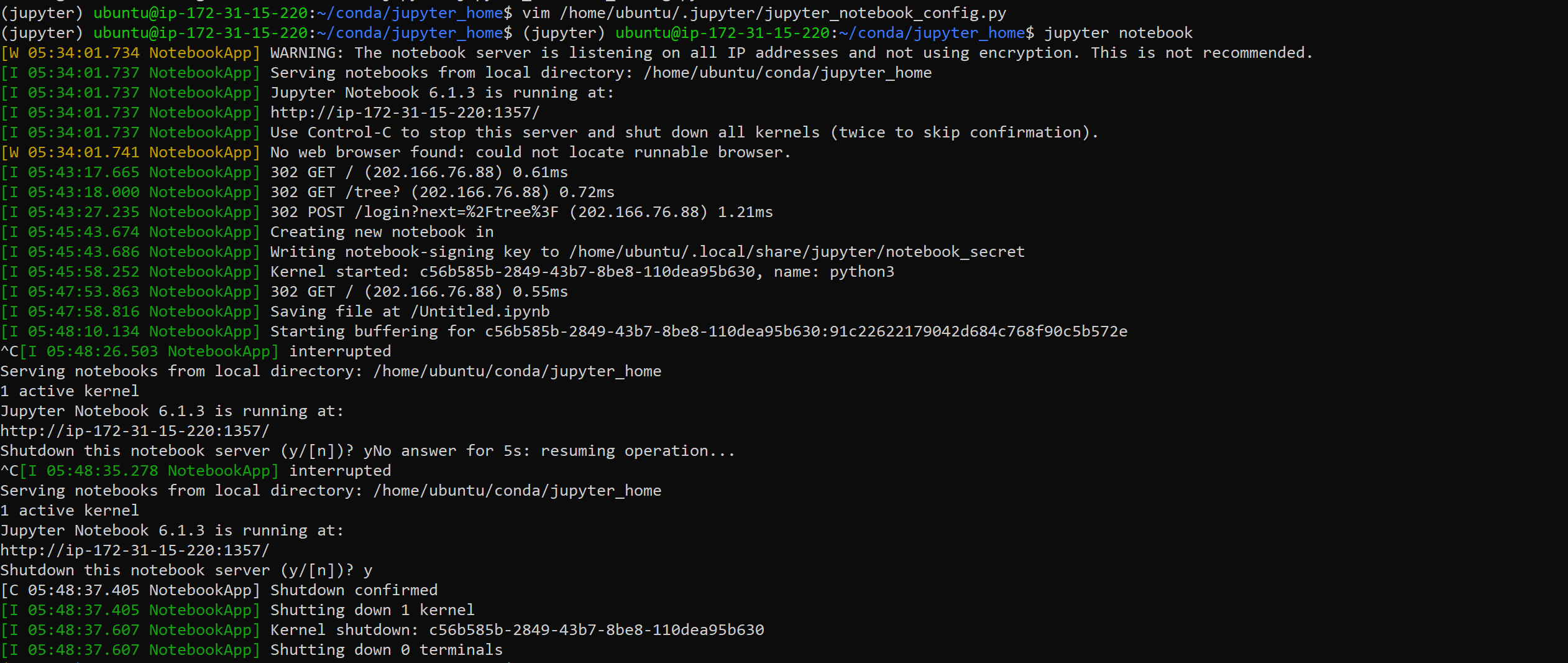
#c.NotebookApp.certfile = u'/root/.jupyter/jupyter.pem'

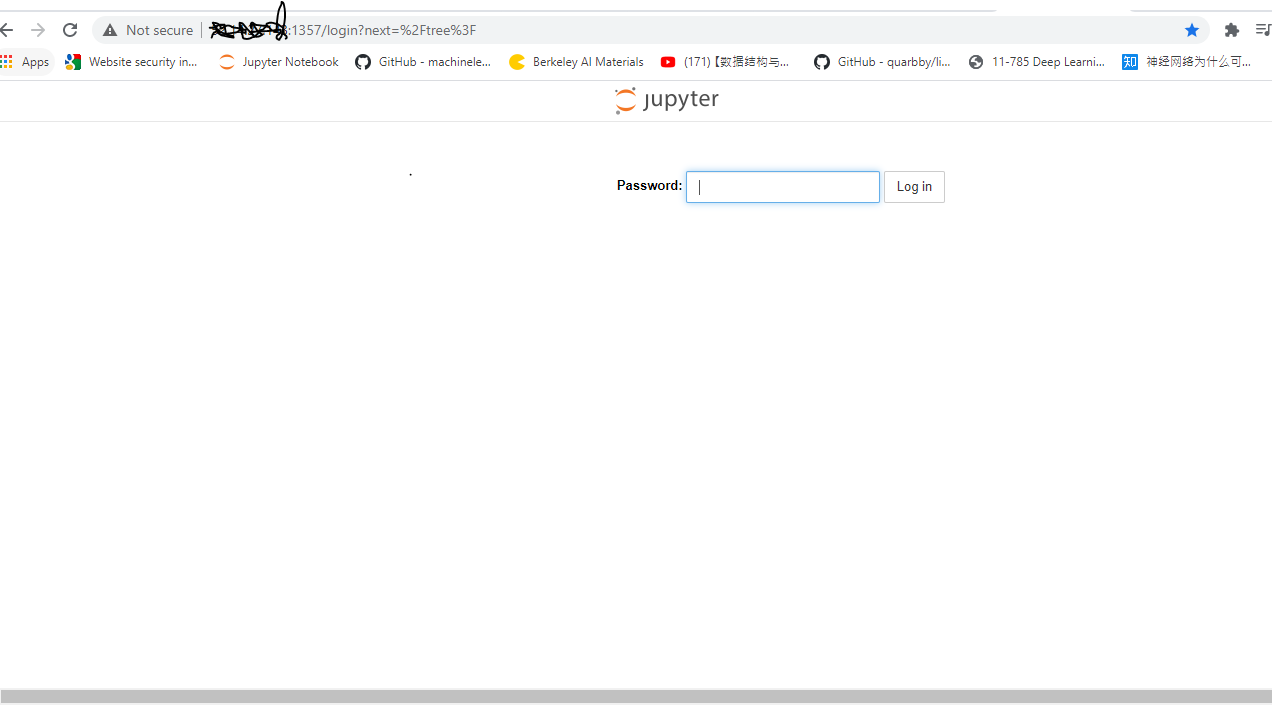
c.NotebookApp.port= 1357

c.NotebookApp.notebook\_dir = "/home/ubuntu/conda/jupyter\_home/"

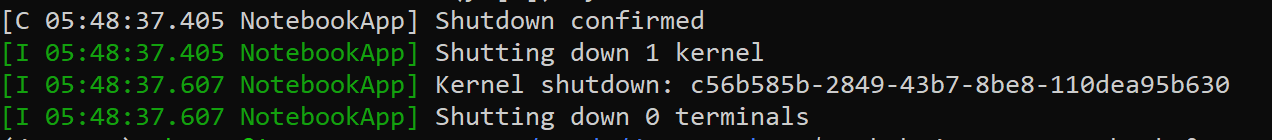


Run the jupyter notebook



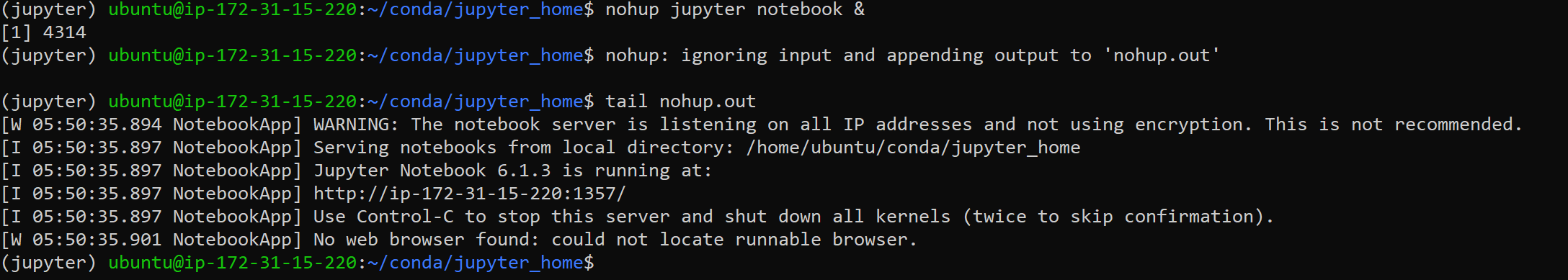


Once the terminal is close, the url can not run or support. To able to run the jupyter even when terminal is closed.



Run nohup jupyter notebook &

Nohup helps to run the command in a terminal will never shutdown.



To kill the jupyter notebook

Run: pkill jupyter