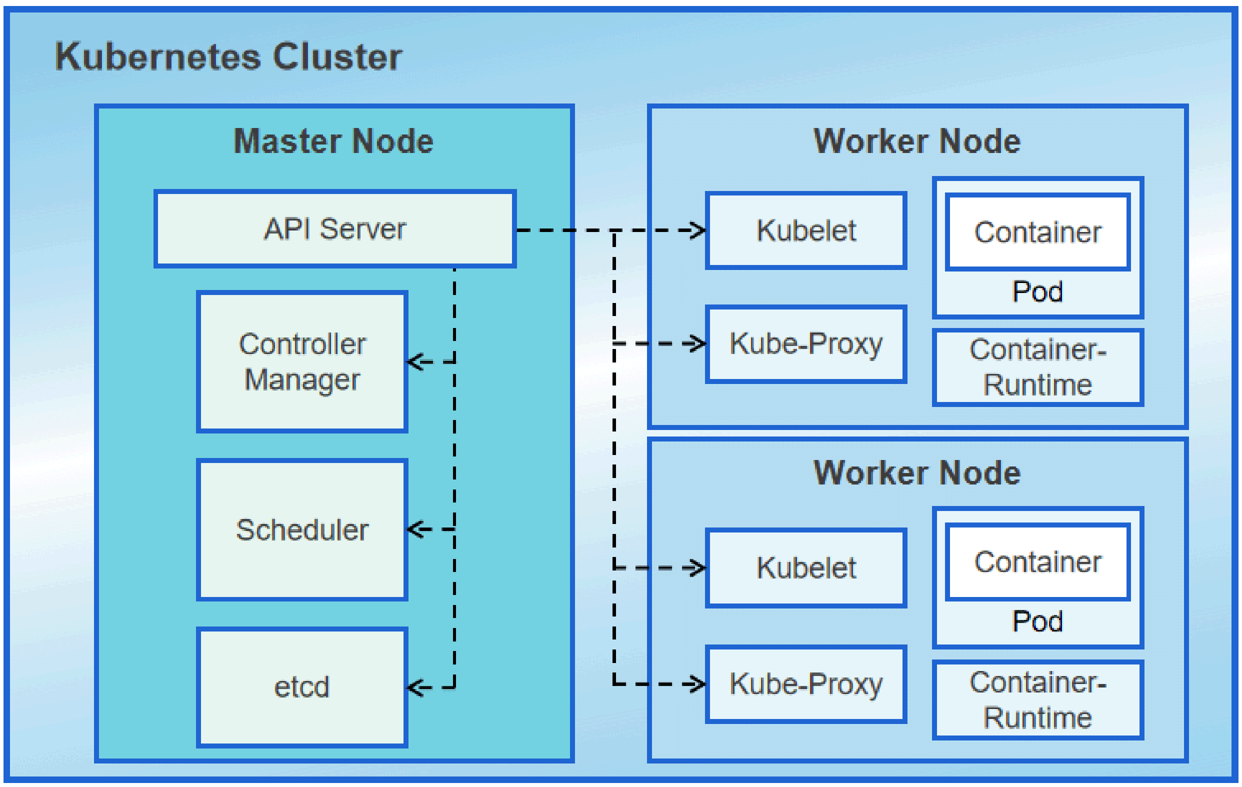
Adv. Devops Experiment no. 3

Name: Rohan Lalchandani
Class: D15A   Roll no.: 25

Aim: To understand the Kubernetes Cluster Architecture, install and Spin Up a Kubernetes Cluster on Linux Machines/Cloud Platforms.

Theory:

Kubernetes cluster architecture is designed to manage containerized applications efficiently. It consists of two main components: the **Control Plane** and the **Worker Nodes**.



1. Control Plane

The control plane is responsible for managing the overall state of the cluster. It makes global decisions about the cluster, such as scheduling applications, and detects/responds to cluster events (like a container failure).

- **API Server**: The entry point for all administrative tasks in the cluster. It exposes the Kubernetes API, acting as the front end of the control plane.
- **etcd**: A distributed key-value store that stores all the cluster's data. It is a highly reliable store for all cluster state and configuration.
- **Controller Manager**: Monitors the cluster state and performs routine tasks like handling node failures, maintaining the correct number of replicas for pods, and balancing the load across the cluster.
- **Scheduler**: Assigns newly created pods to nodes based on resource availability, affinity rules, and other policies.

2. Worker Nodes

Worker nodes run the containerized applications (pods). Each node has a set of components that communicate with the control plane to receive and execute tasks.

- **Kubelet**: The agent that runs on every node in the cluster. It ensures that containers are running in pods and communicates with the API server to get the desired state of the node.
- **Container Runtime**: The software responsible for running containers, such as Docker, containerd, or CRI-O. It manages the lifecycle of containers on the node.
- **Kube Proxy**: Ensures networking rules are applied to allow communication between the different components in the cluster, handling services and load balancing.

3. Cluster Networking

Kubernetes provides an abstracted networking layer for communication between pods. Key features include:

- **Pod-to-Pod Communication**: Every pod in a cluster can communicate with every other pod without using NAT (Network Address Translation).

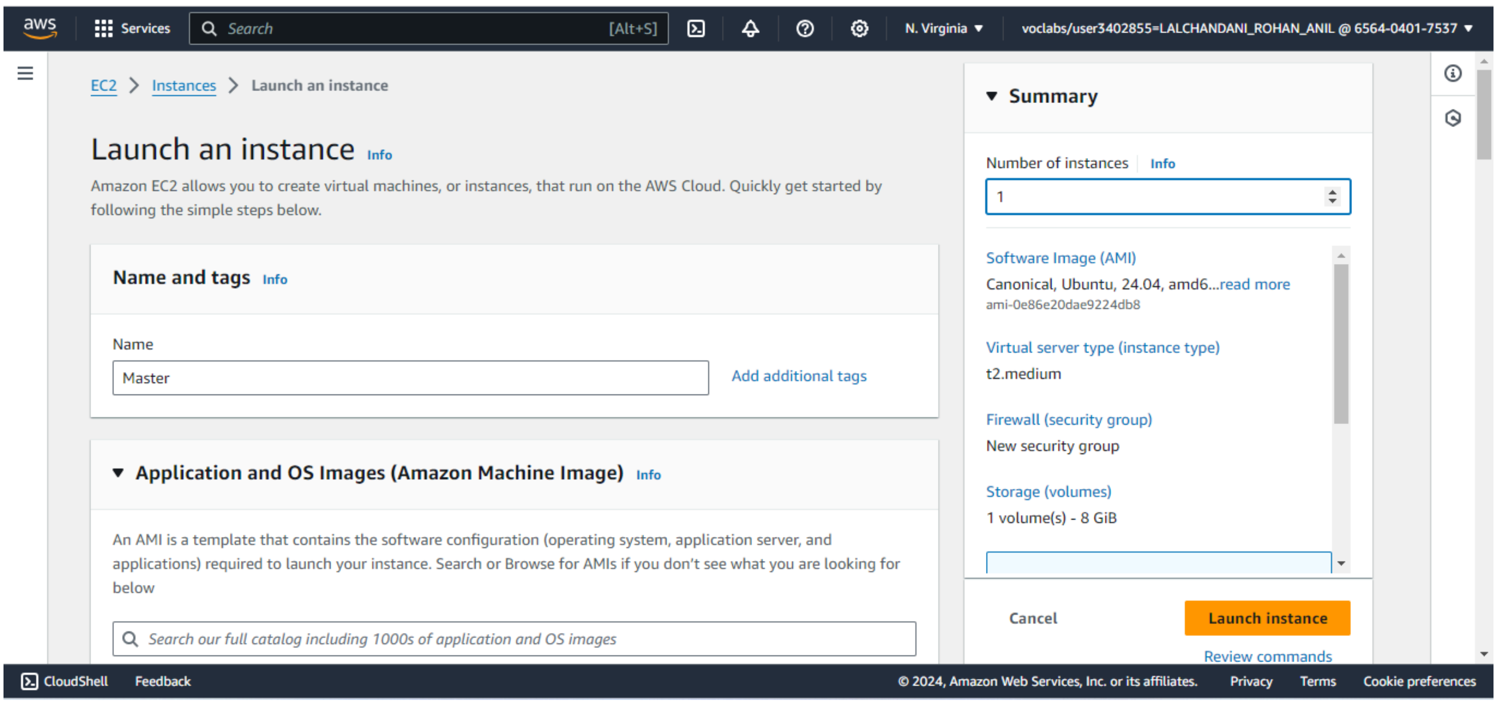
- **Service Abstraction**: Services provide a stable endpoint to access a group of pods. This decouples the frontend from backend pods, offering load balancing and failure recovery.

4. Storage

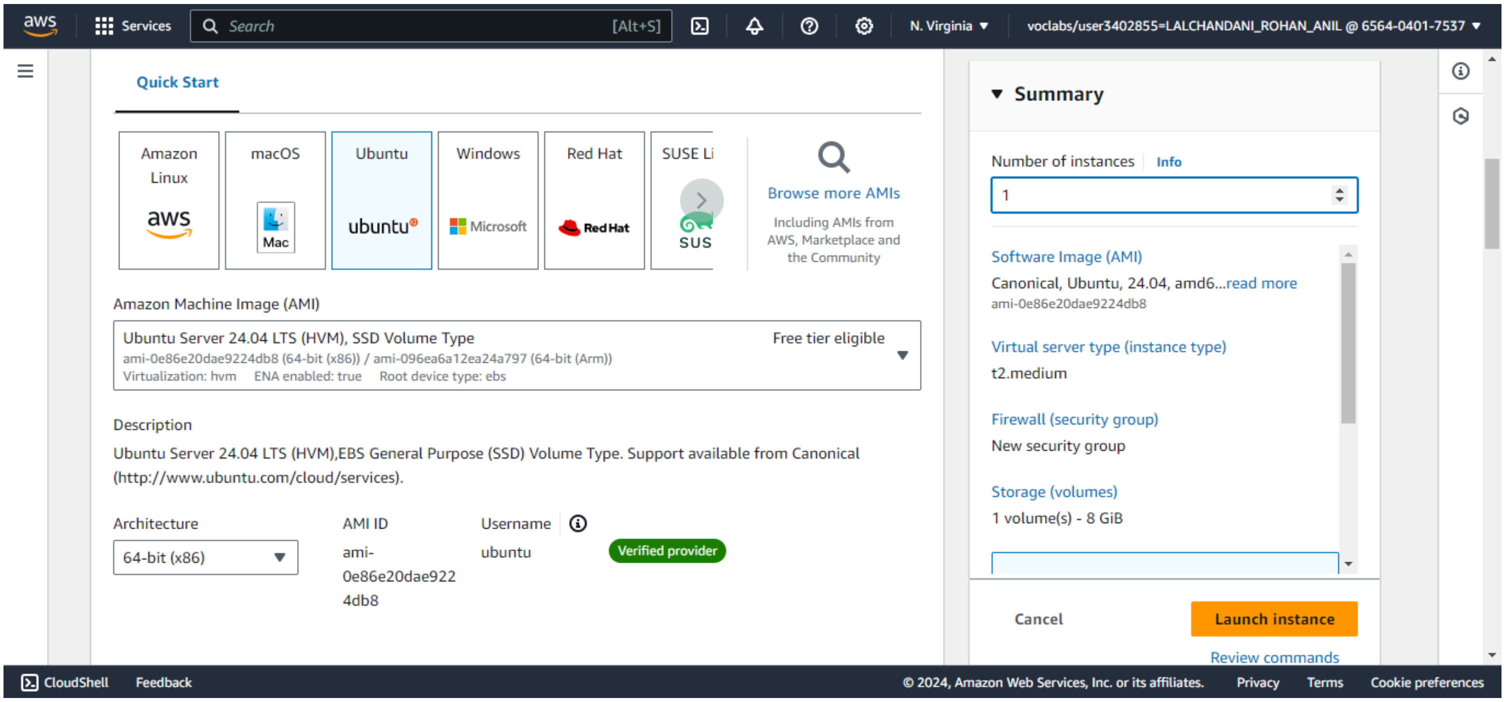
Kubernetes uses persistent storage systems like persistent volumes (PVs) and persistent volume claims (PVCs) to manage data storage needs.

Implementation:

1. Create three EC2 Ubuntu Instances - Master, Worker 1 and Worker 2.



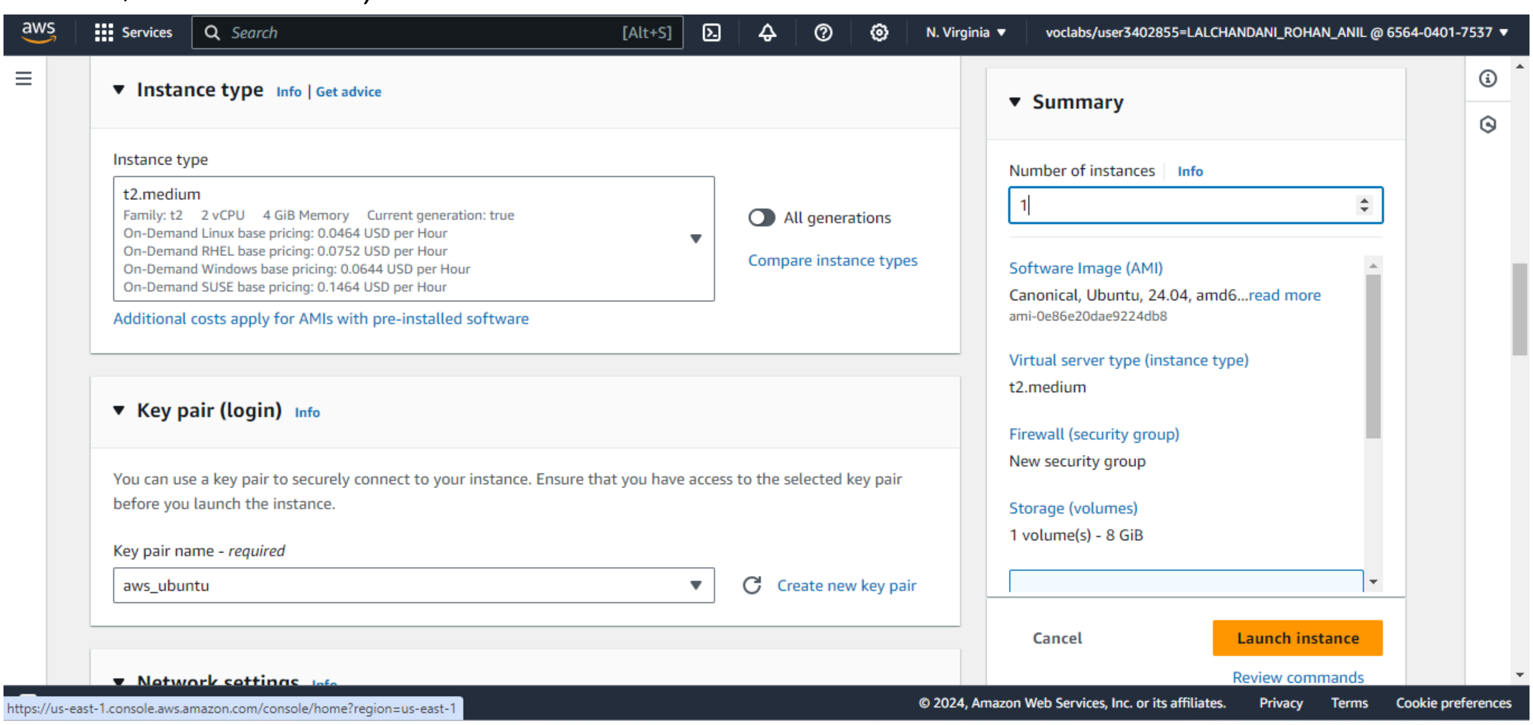
2. Select the following AMI image.



3. Select t2.medium in instance type.

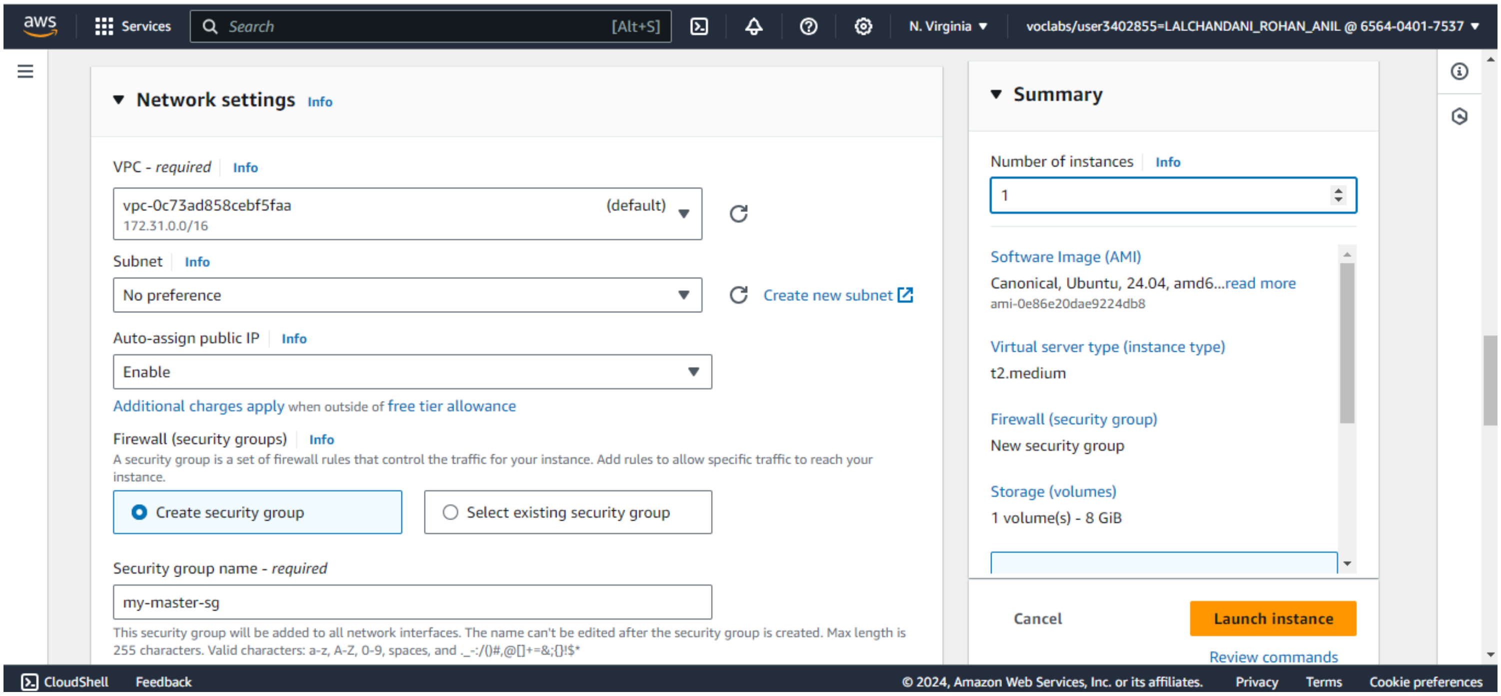
Why?

t2.medium has 2 vCPUs and 4 GB of RAM, which allows it to run more pods and handle larger or more resource-intensive containers compared to **t2.micro** (1 vCPU, 1 GB of RAM).

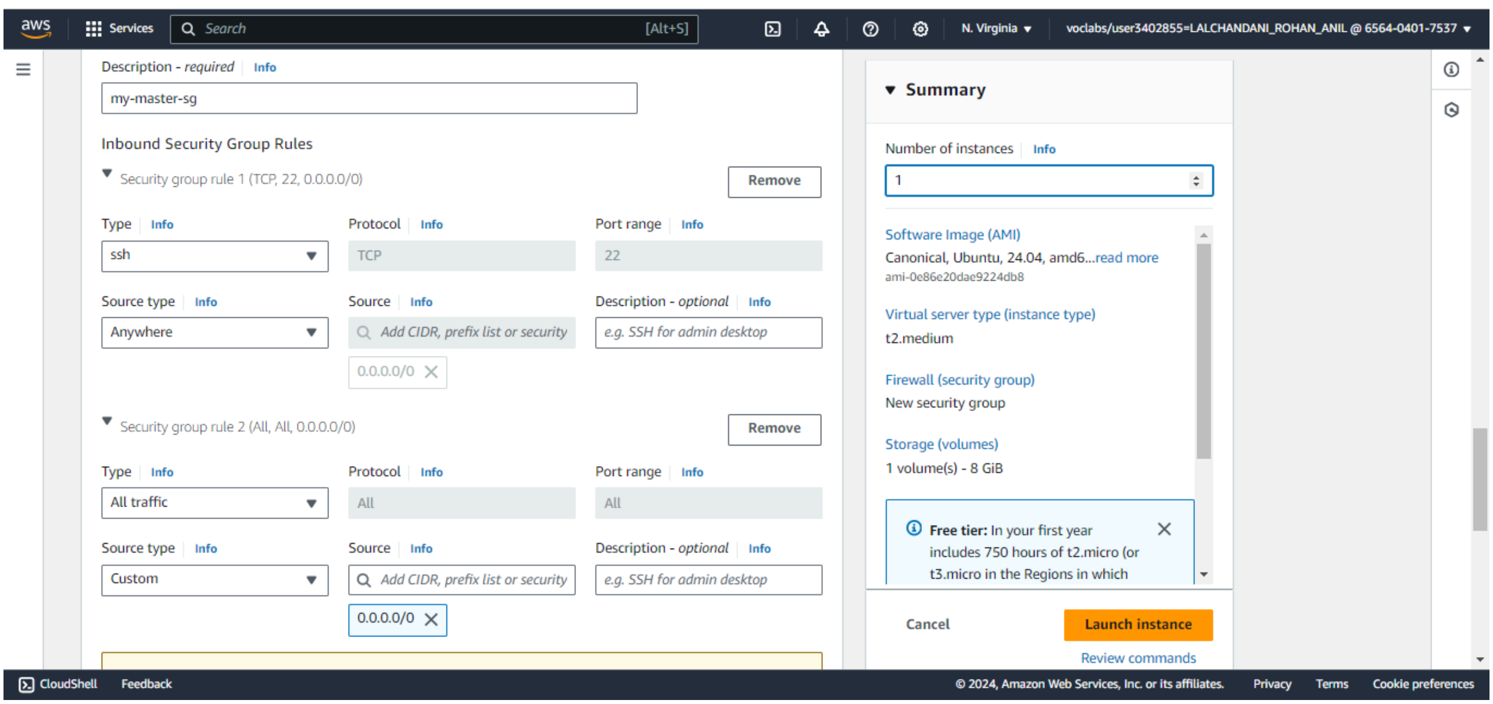


4. Create security group

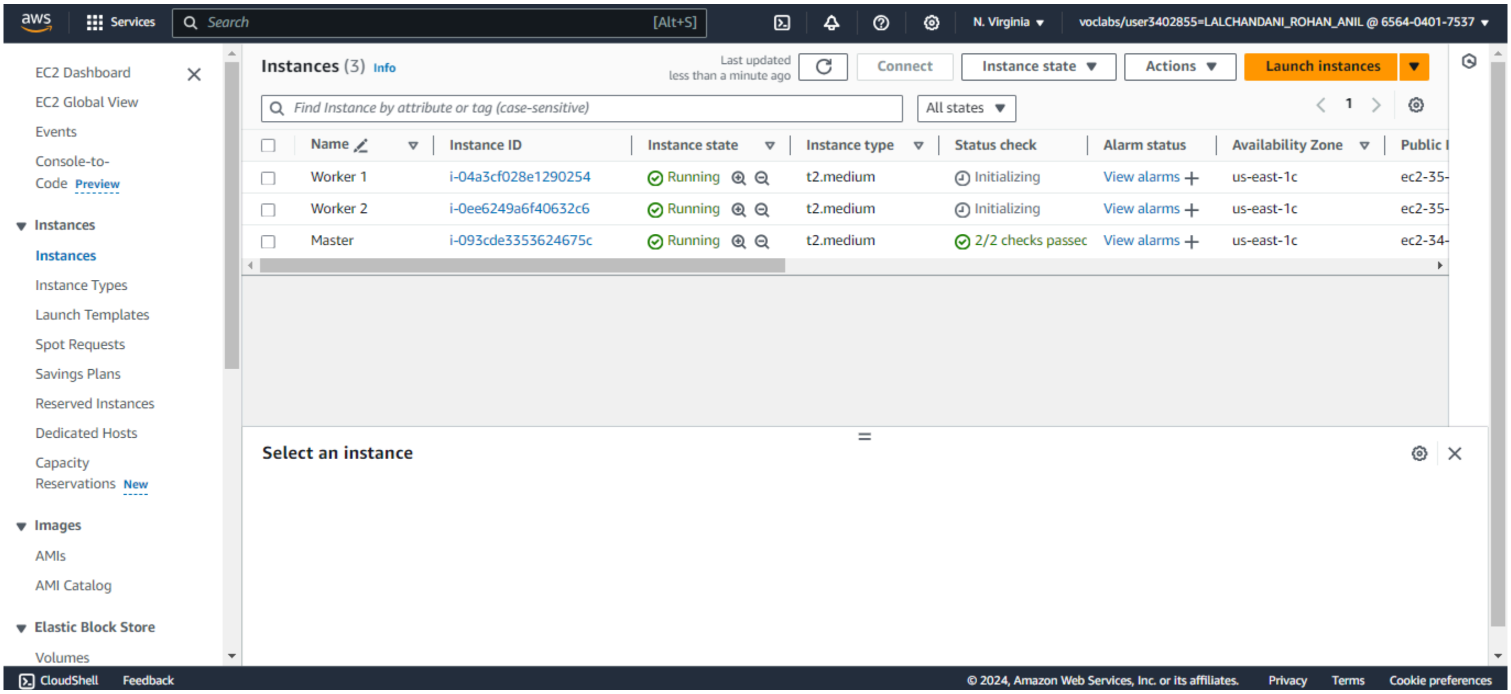
In it type the group name and description as my-master-sg



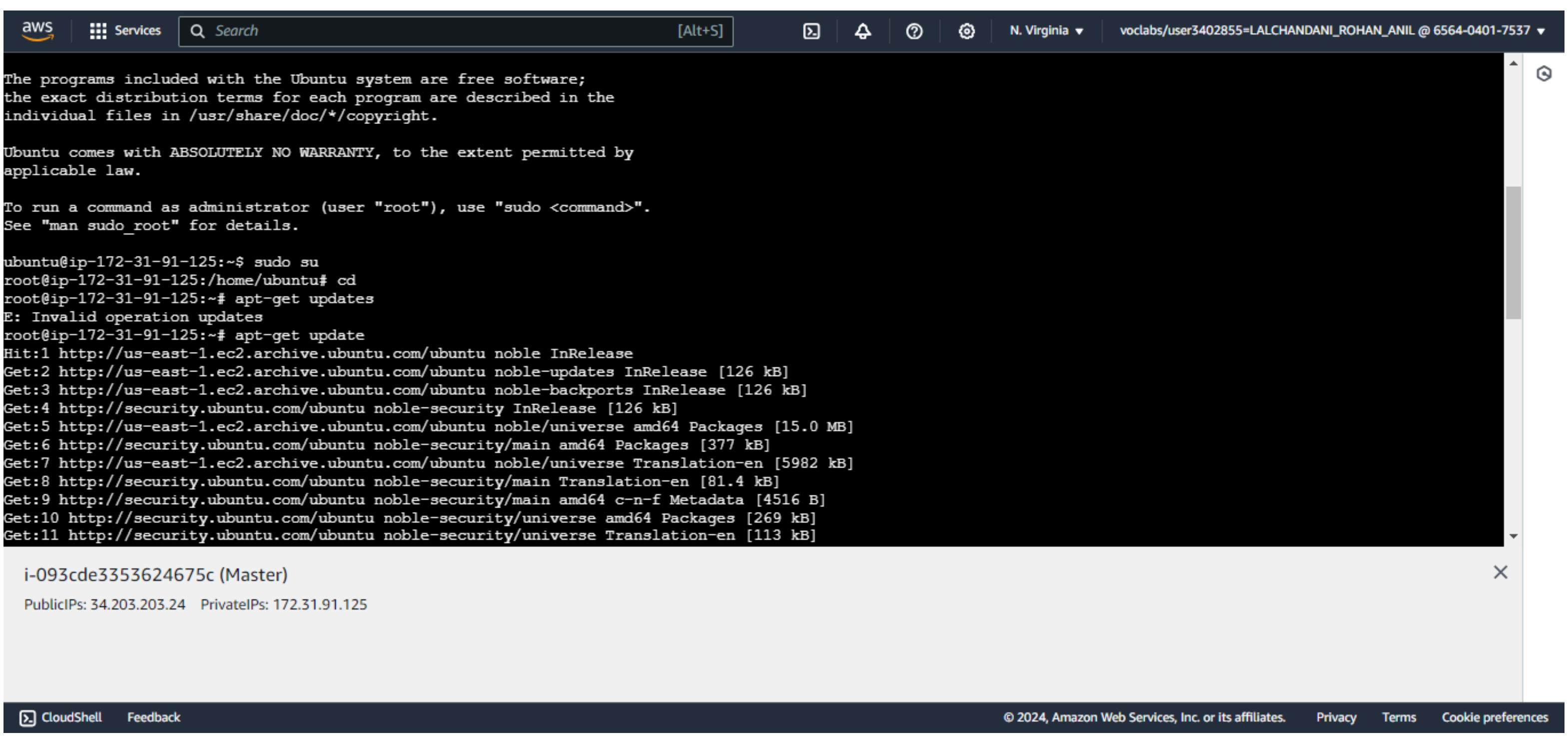
5. Edit the inbound rules and add a new rule to accept "All traffic" as shown below.



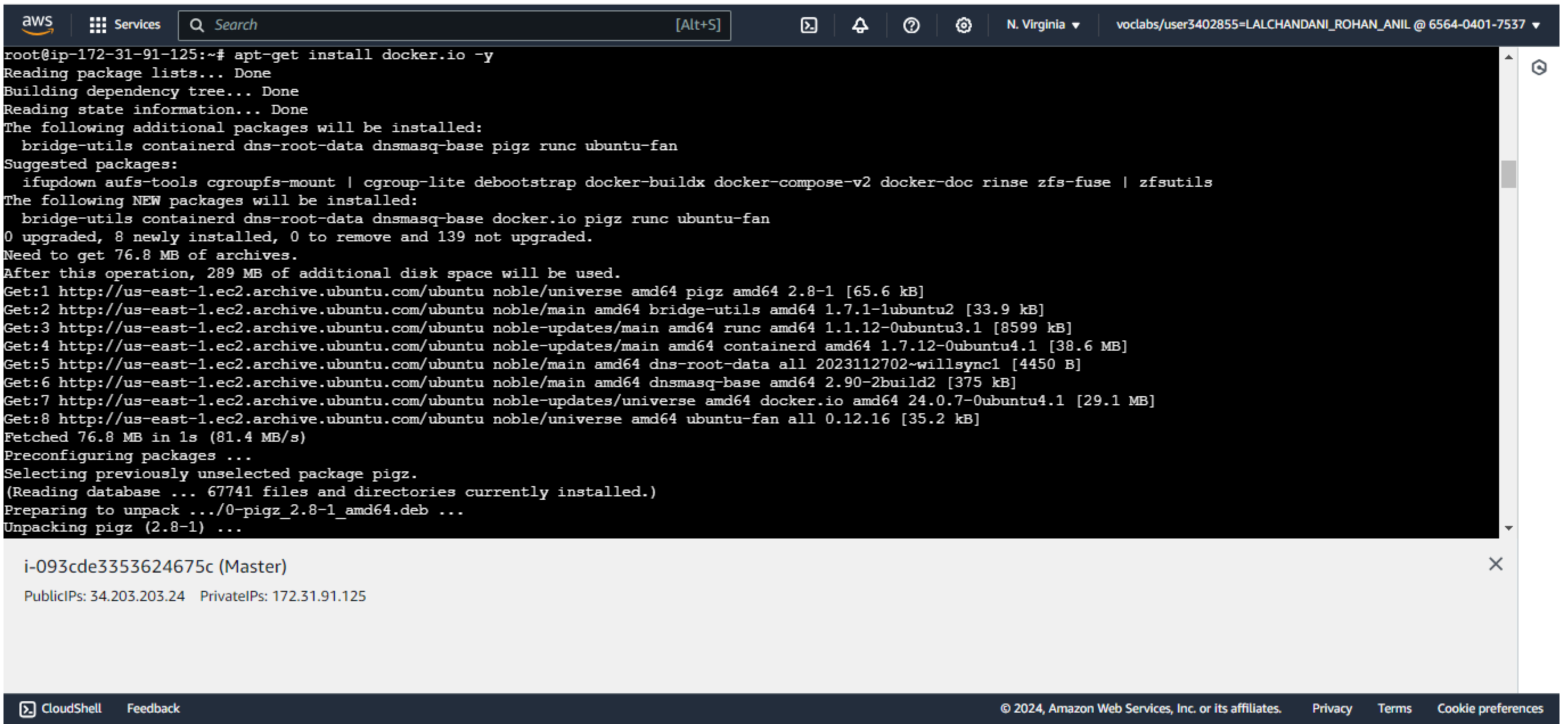
6. Create Worker 1 and Worker 2 also with same settings and select the existing security group of “my-master-sg” in them. Connect all the instances.



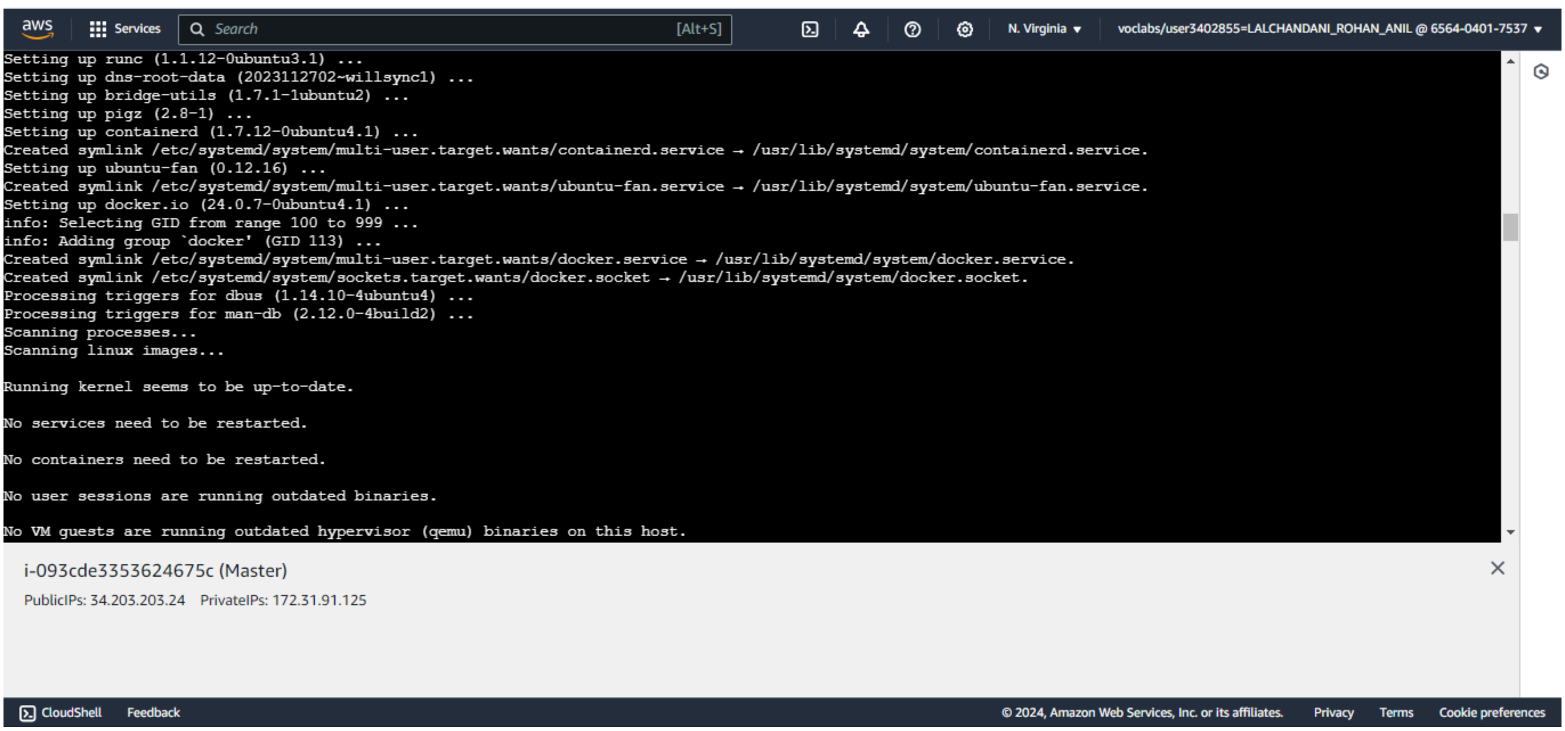
7. Now run “sudo su” to execute commands as root user.
“cd” to change directory
then “apt-get update”



8. Install docker by “apt-get install docker.io -y”

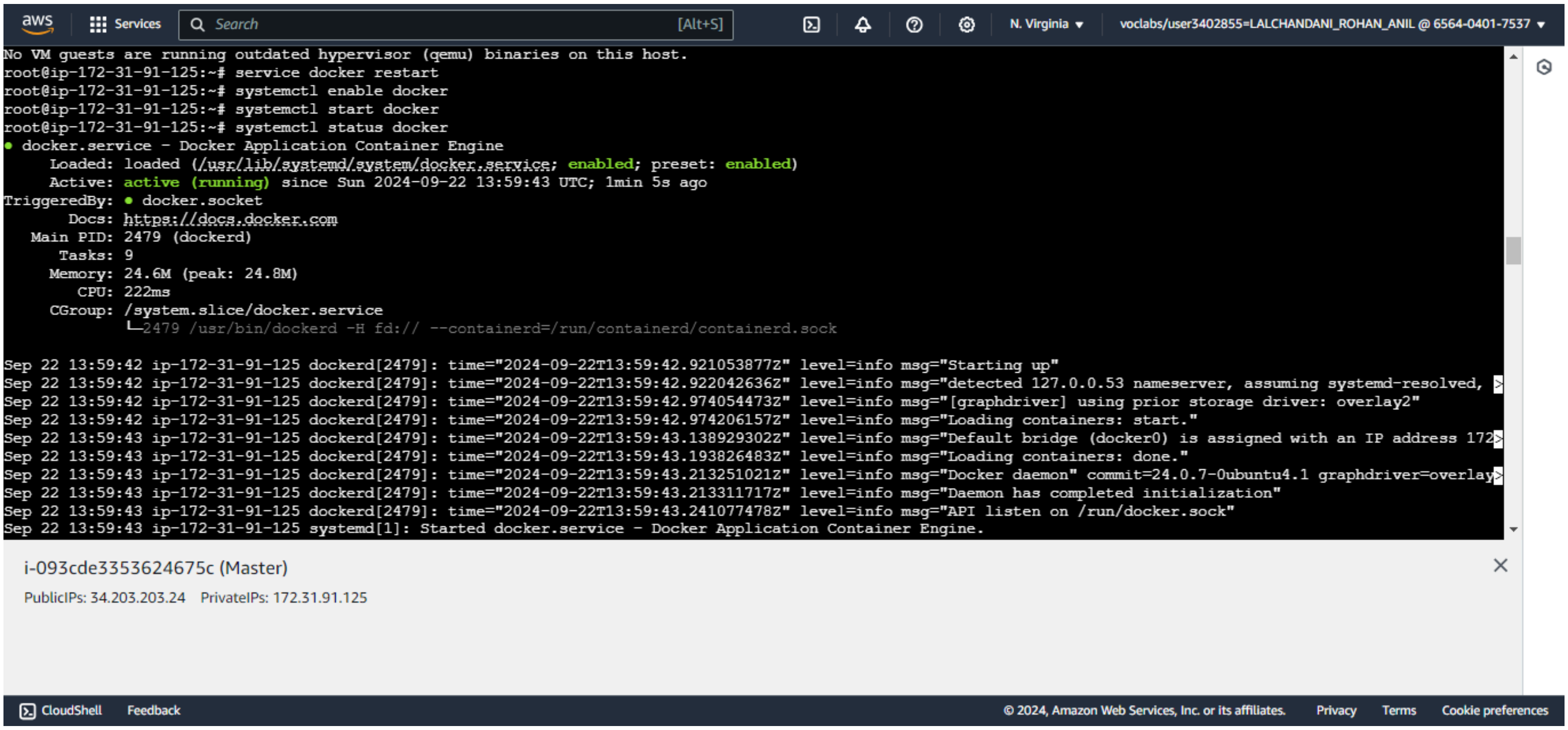



```
root@ip-172-31-91-125:~# apt-get install docker.io -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  bridge-utils containerd dns-root-data dnsmasq-base pigz runc ubuntu-fan
Suggested packages:
  ifupdown aufs-tools cgroupfs-mount | cgroup-lite debootstrap docker-buildx docker-compose-v2 docker-doc rinse zfs-fuse | zfsutils
The following NEW packages will be installed:
  bridge-utils containerd dns-root-data dnsmasq-base docker.io pigz runc ubuntu-fan
0 upgraded, 8 newly installed, 0 to remove and 139 not upgraded.
Need to get 76.8 MB of archives.
After this operation, 289 MB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 pigz amd64 2.8-1 [65.6 kB]
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 bridge-utils amd64 1.7.1-1ubuntu2 [33.9 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 runc amd64 1.1.12-0ubuntu3.1 [8599 kB]
Get:4 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 containerd amd64 1.7.12-0ubuntu4.1 [38.6 MB]
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 dns-root-data all 2023112702~willsync1 [4450 B]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 dnsmasq-base amd64 2.90-2build2 [375 kB]
Get:7 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 docker.io amd64 24.0.7-0ubuntu4.1 [29.1 MB]
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 ubuntu-fan all 0.12.16 [35.2 kB]
Fetched 76.8 MB in 1s (81.4 MB/s)
Preconfiguring packages ...
Selecting previously unselected package pigz.
(Reading database ... 67741 files and directories currently installed.)
Preparing to unpack .../0-pigz_2.8-1_amd64.deb ...
Unpacking pigz (2.8-1) ...
```




```
Setting up runc (1.1.12-0ubuntu3.1) ...
Setting up dns-root-data (2023112702~willsync1) ...
Setting up bridge-utils (1.7.1-1ubuntu2) ...
Setting up pigz (2.8-1) ...
Setting up containerd (1.7.12-0ubuntu4.1) ...
Created symlink /etc/systemd/system/multi-user.target.wants/containerd.service → /usr/lib/systemd/system/containerd.service.
Setting up ubuntu-fan (0.12.16) ...
Created symlink /etc/systemd/system/multi-user.target.wants/ubuntu-fan.service → /usr/lib/systemd/system/ubuntu-fan.service.
Setting up docker.io (24.0.7-0ubuntu4.1) ...
info: Selecting GID from range 100 to 999 ...
info: Adding group `docker' (GID 113) ...
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /usr/lib/systemd/system/docker.service.
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket → /usr/lib/systemd/system/docker.socket.
Processing triggers for dbus (1.14.10-4ubuntu4) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
```

9. enable, start and check status of docker if it’s active.



10. Now we have to install kubernetes, for that we will refer the documentation at https://kubernetes.io/docs/setup/production-environment/tools/kubeadm/install-kubeadm/

Got to the “Debian-based distributions” and follow the instructions below to install kubeadm

sudo apt-get install -y apt-transport-https ca-certificates curl gpg
sudo mkdir -p -m 755 /etc/apt/keyrings
curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.31/deb/Release.key | sudo gpg --dearmor -o /etc/apt/keyrings/kubernetes-apt-keyring.gpg
echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.31/deb/ /' | sudo tee /etc/apt/sources.list.d/kubernetes.list
sudo apt-get update
sudo apt-get install -y kubelet kubeadm kubectl
sudo apt-mark hold kubelet kubeadm kubectl
sudo systemctl enable --now kubelet
systemctl restart kubelet

aws Services Search [Alt+S] N. Virginia voclabs/user3402855=LALCHANDANI_ROHAN_ANIL @ 6564-0401-7537

```
root@ip-172-31-91-125:~# sudo apt-get install -y apt-transport-https ca-certificates curl gpg
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
ca-certificates is already the newest version (20240203).
ca-certificates set to manually installed.
gpg is already the newest version (2.4.4-2ubuntu17).
gpg set to manually installed.
The following NEW packages will be installed:
  apt-transport-https
The following packages will be upgraded:
  curl libcurl3t64-gnutls libcurl4t64
3 upgraded, 1 newly installed, 0 to remove and 136 not upgraded.
Need to get 904 kB of archives.
After this operation, 38.9 kB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 apt-transport-https all 2.7.14build2 [3974 B]
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 curl amd64 8.5.0-2ubuntu10.4 [227 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 libcurl4t64 amd64 8.5.0-2ubuntu10.4 [341 kB]
Get:4 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 libcurl3t64-gnutls amd64 8.5.0-2ubuntu10.4 [333 kB]
Fetched 904 kB in 0s (24.7 MB/s)
Selecting previously unselected package apt-transport-https.
(Reading database ... 68108 files and directories currently installed.)
Preparing to unpack .../apt-transport-https_2.7.14build2_all.deb ...
Unpacking apt-transport-https (2.7.14build2) ...
Preparing to unpack .../curl_8.5.0-2ubuntu10.4_amd64.deb ...
Unpacking curl (8.5.0-2ubuntu10.4) over (8.5.0-2ubuntu10.1) ...
Preparing to unpack .../libcurl4t64_8.5.0-2ubuntu10.4_amd64.deb ...
```

i-093cde3353624675c (Master)

PublicIPs: 34.203.203.24 PrivateIPs: 172.31.91.125

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

aws Services Search [Alt+S] N. Virginia voclabs/user3402855=LALCHANDANI_ROHAN_ANIL @ 6564-0401-7537

```
Preparing to unpack .../libcurl4t64_8.5.0-2ubuntu10.4_amd64.deb ...
Unpacking libcurl4t64:amd64 (8.5.0-2ubuntu10.4) over (8.5.0-2ubuntu10.1) ...
Preparing to unpack .../libcurl3t64-gnutls_8.5.0-2ubuntu10.4_amd64.deb ...
Unpacking libcurl3t64-gnutls:amd64 (8.5.0-2ubuntu10.4) over (8.5.0-2ubuntu10.1) ...
Setting up apt-transport-https (2.7.14build2) ...
Setting up libcurl4t64:amd64 (8.5.0-2ubuntu10.4) ...
Setting up libcurl3t64-gnutls:amd64 (8.5.0-2ubuntu10.4) ...
Setting up curl (8.5.0-2ubuntu10.4) ...
Processing triggers for man-db (2.12.0-4build2) ...
Processing triggers for libc-bin (2.39-0ubuntu8.2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@ip-172-31-91-125:~# sudo mkdir -p -m 755 /etc/apt/keyrings
root@ip-172-31-91-125:~# curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.31/deb/Release.key | sudo gpg --dearmor -o /etc/apt/keyrings/kubernetes-apt-keyring.gpg
root@ip-172-31-91-125:~# echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.31/deb/ /' | sudo tee /etc/apt/so
urces.list.d/kubernetes.list
deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.31/deb/ /
```

i-093cde3353624675c (Master)

PublicIPs: 34.203.203.24 PrivateIPs: 172.31.91.125

aws Services Search [Alt+S] N. Virginia voclabs/user3402855=LALCHANDANI_ROHAN_ANIL @ 6564-0401-7537

```
deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.31/deb/ /
root@ip-172-31-91-125:~# sudo apt-get update
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu noble-security InRelease
Get:5 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb  InRelease [1186 B]
Get:6 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb  Packages [4865 B]
Fetched 132 kB in 0s (288 kB/s)
Reading package lists... Done
root@ip-172-31-91-125:~# sudo apt-get install -y kubelet kubeadm kubectl
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  conntrack cri-tools kubernetes-cni
The following NEW packages will be installed:
  conntrack cri-tools kubeadm kubectl kubelet kubernetes-cni
0 upgraded, 6 newly installed, 0 to remove and 136 not upgraded.
Need to get 87.4 MB of archives.
After this operation, 314 MB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 conntrack amd64 1:1.4.8-1ubuntu1 [37.9 kB]
Get:2 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb  cri-tools 1.31.1-1.1 [15.7 MB]
Get:3 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb  kubeadm 1.31.1-1.1 [11.4 MB]
Get:4 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb  kubectl 1.31.1-1.1 [11.2 MB]
Get:5 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb  kubernetes-cni 1.5.1-1.1 [33.9 MB]
Get:6 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb  kubelet 1.31.1-1.1 [15.2 MB]
```

i-093cde3353624675c (Master)

PublicIPs: 34.203.203.24 PrivateIPs: 172.31.91.125

CloudShell Feedback

aws Services Search [Alt+S] N. Virginia voclabs/user3402855=LALCHANDANI_ROHAN_ANIL @ 6564-0401-7537

```
Selecting previously unselected package conntrack.
(Reading database ... 68112 files and directories currently installed.)
Preparing to unpack .../0-conntrack_1%3a1.4.8-1ubuntu1_amd64.deb ...
Unpacking conntrack (1:1.4.8-1ubuntu1) ...
Selecting previously unselected package cri-tools.
Preparing to unpack .../1-cri-tools_1.31.1-1.1_amd64.deb ...
Unpacking cri-tools (1.31.1-1.1) ...
Selecting previously unselected package kubeadm.
Preparing to unpack .../2-kubeadm_1.31.1-1.1_amd64.deb ...
Unpacking kubeadm (1.31.1-1.1) ...
Selecting previously unselected package kubectl.
Preparing to unpack .../3-kubectl_1.31.1-1.1_amd64.deb ...
Unpacking kubectl (1.31.1-1.1) ...
Selecting previously unselected package kubernetes-cni.
Preparing to unpack .../4-kubernetes-cni_1.5.1-1.1_amd64.deb ...
Unpacking kubernetes-cni (1.5.1-1.1) ...
Selecting previously unselected package kubelet.
Preparing to unpack .../5-kubelet_1.31.1-1.1_amd64.deb ...
Unpacking kubelet (1.31.1-1.1) ...
Setting up conntrack (1:1.4.8-1ubuntu1) ...
Setting up kubectl (1.31.1-1.1) ...
Setting up cri-tools (1.31.1-1.1) ...
Setting up kubernetes-cni (1.5.1-1.1) ...
Setting up kubeadm (1.31.1-1.1) ...
Setting up kubelet (1.31.1-1.1) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
```

i-093cde3353624675c (Master)

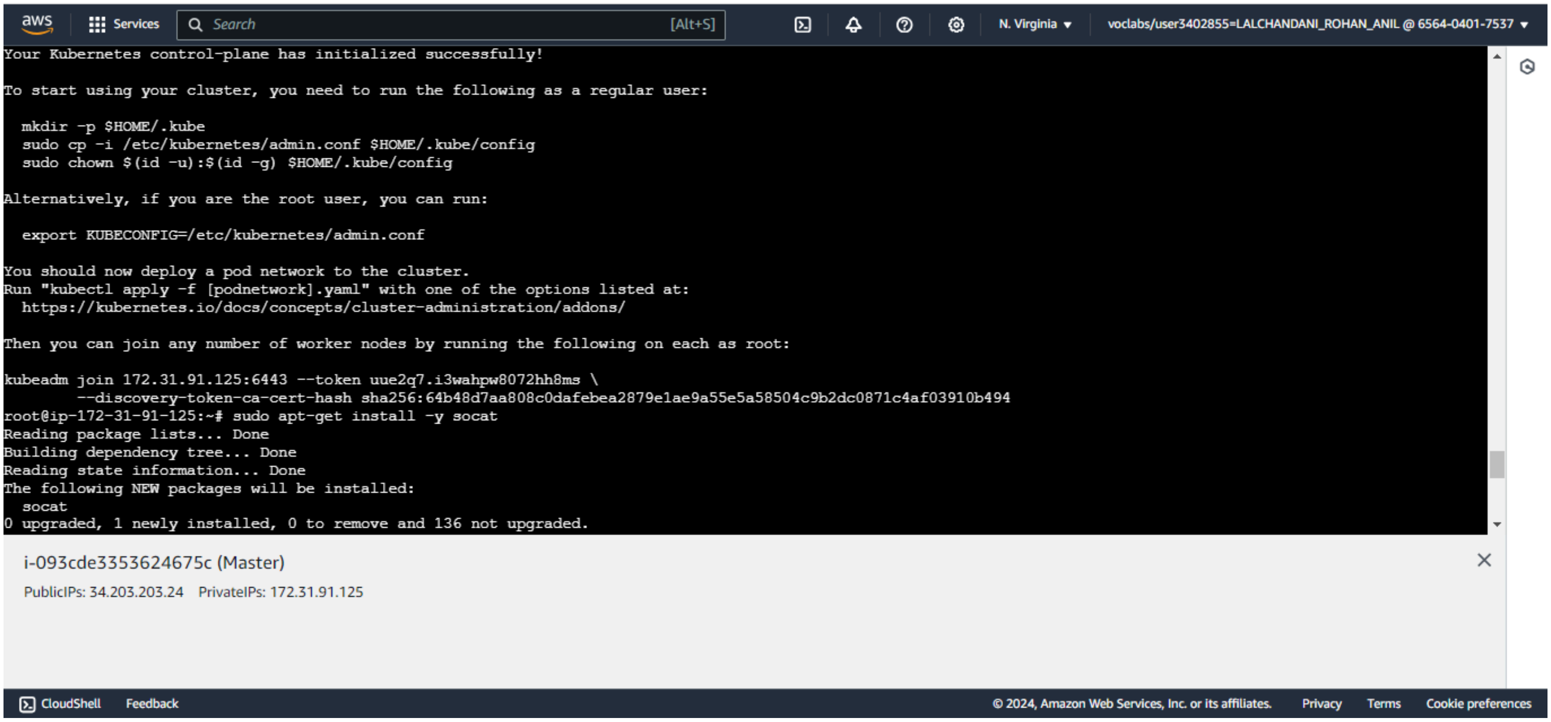
PublicIPs: 34.203.203.24 PrivateIPs: 172.31.91.125

CloudShell Feedback

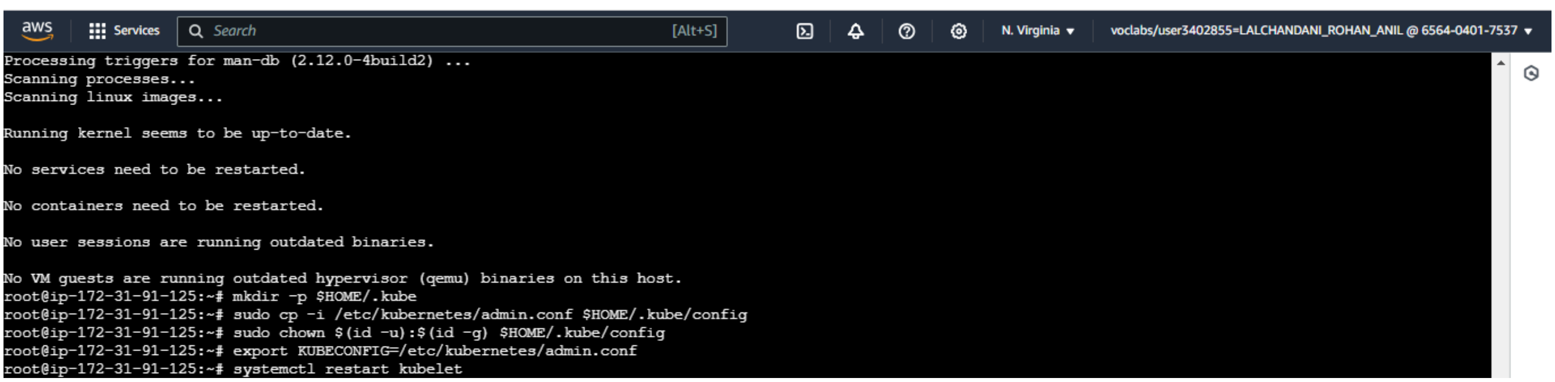
aws Services Search [Alt+S] N. Virginia voclabs/user3402855=LALCHANDANI_ROHAN_ANIL @ 6564-0401-7537

```
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@ip-172-31-91-125:~# sudo apt-mark hold kubelet kubeadm kubectl
kubelet set on hold.
kubeadm set on hold.
kubectl set on hold.
root@ip-172-31-91-125:~# sudo systemctl enable --now kubelet
```

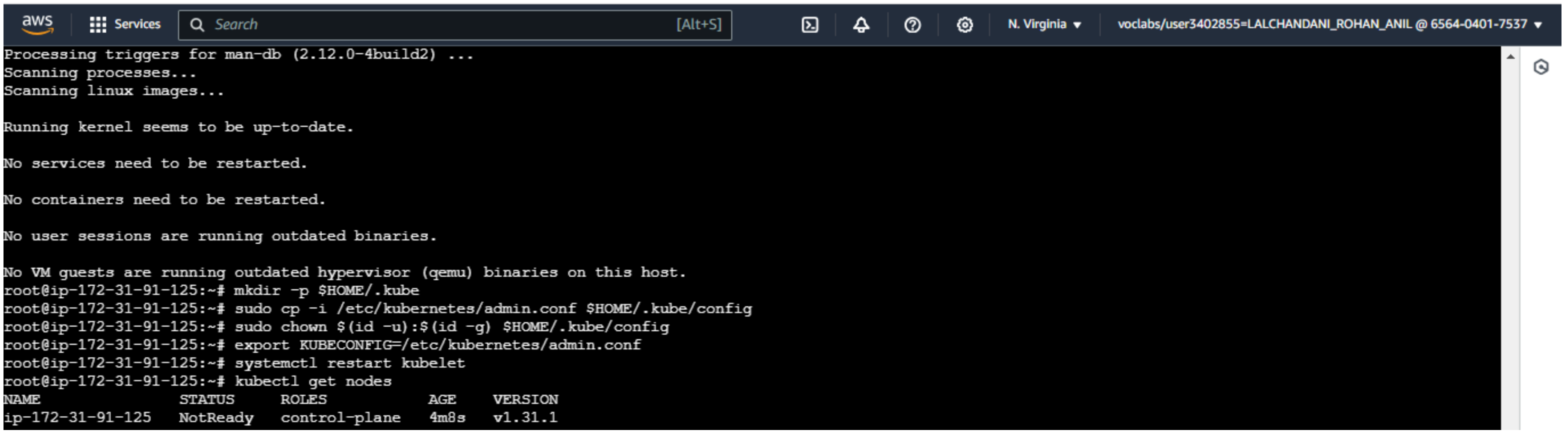
Now only in **Master Node**, execute the “kubeadm init” to initialize the cluster.



11. Now copy the 3 commands from the regular user and 1 command from root user and one by one execute them.
Execute the same set of steps from 1 to 11 in Worker 1 and Worker 2 except the kubeadm init command.

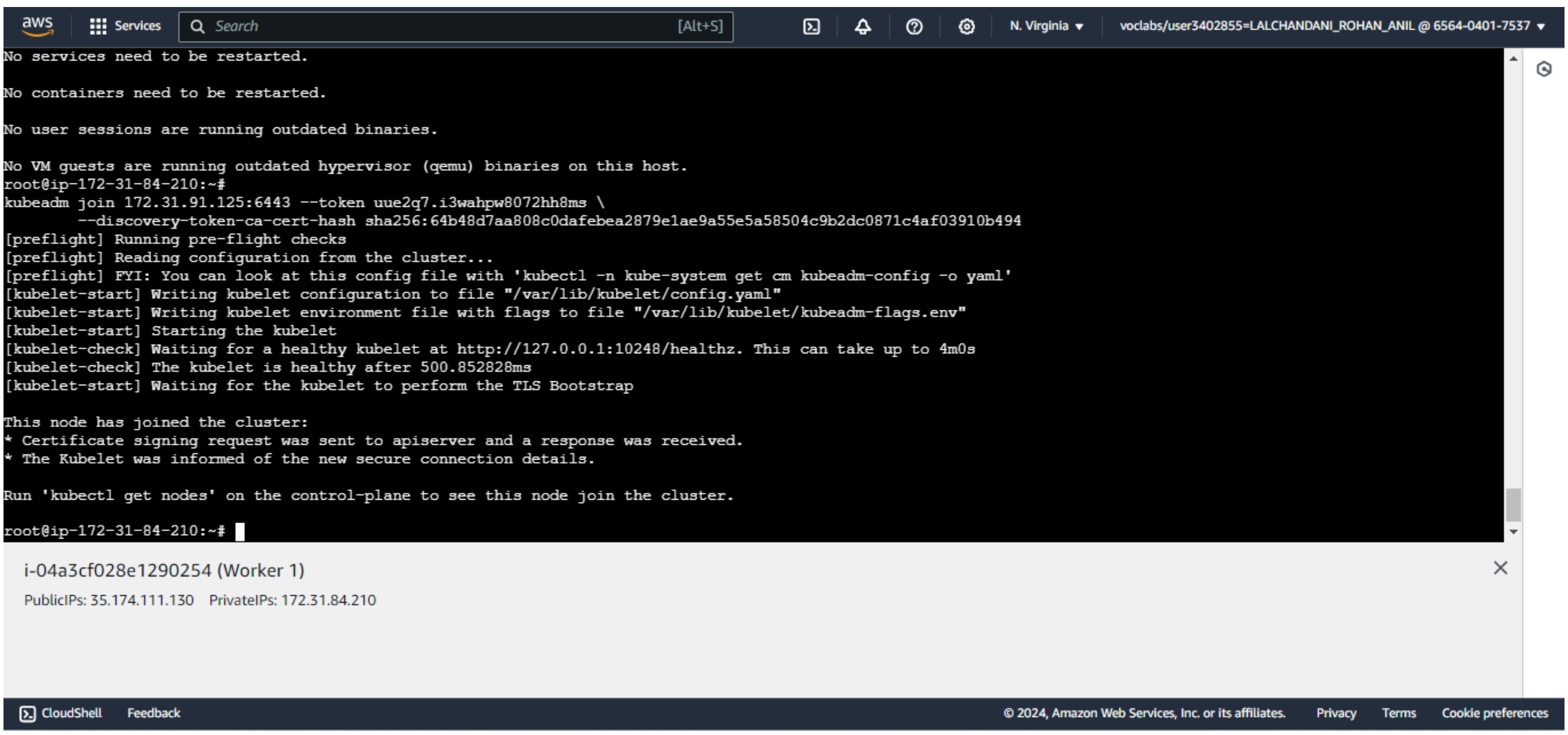


12. Execute “kubectl get nodes” in master to get a list of nodes present. Initially there will be no nodes present.

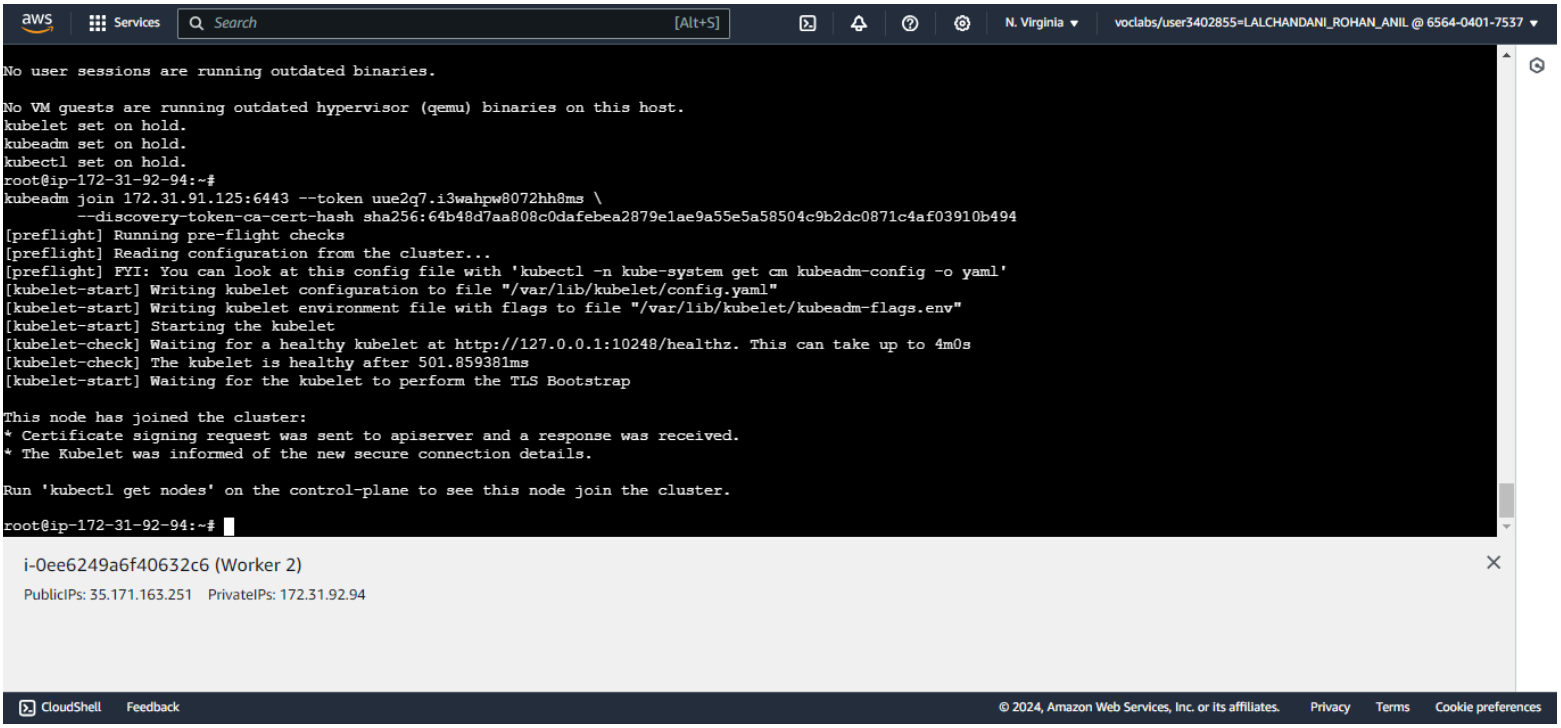



```
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@ip-172-31-91-125:~# mkdir -p $HOME/.kube
root@ip-172-31-91-125:~# sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
root@ip-172-31-91-125:~# sudo chown $(id -u):$(id -g) $HOME/.kube/config
root@ip-172-31-91-125:~# export KUBECONFIG=/etc/kubernetes/admin.conf
root@ip-172-31-91-125:~# systemctl restart kubelet
root@ip-172-31-91-125:~# kubectl get nodes
NAME               STATUS     ROLES           AGE    VERSION
ip-172-31-91-125   NotReady   control-plane   4m8s   v1.31.1
```

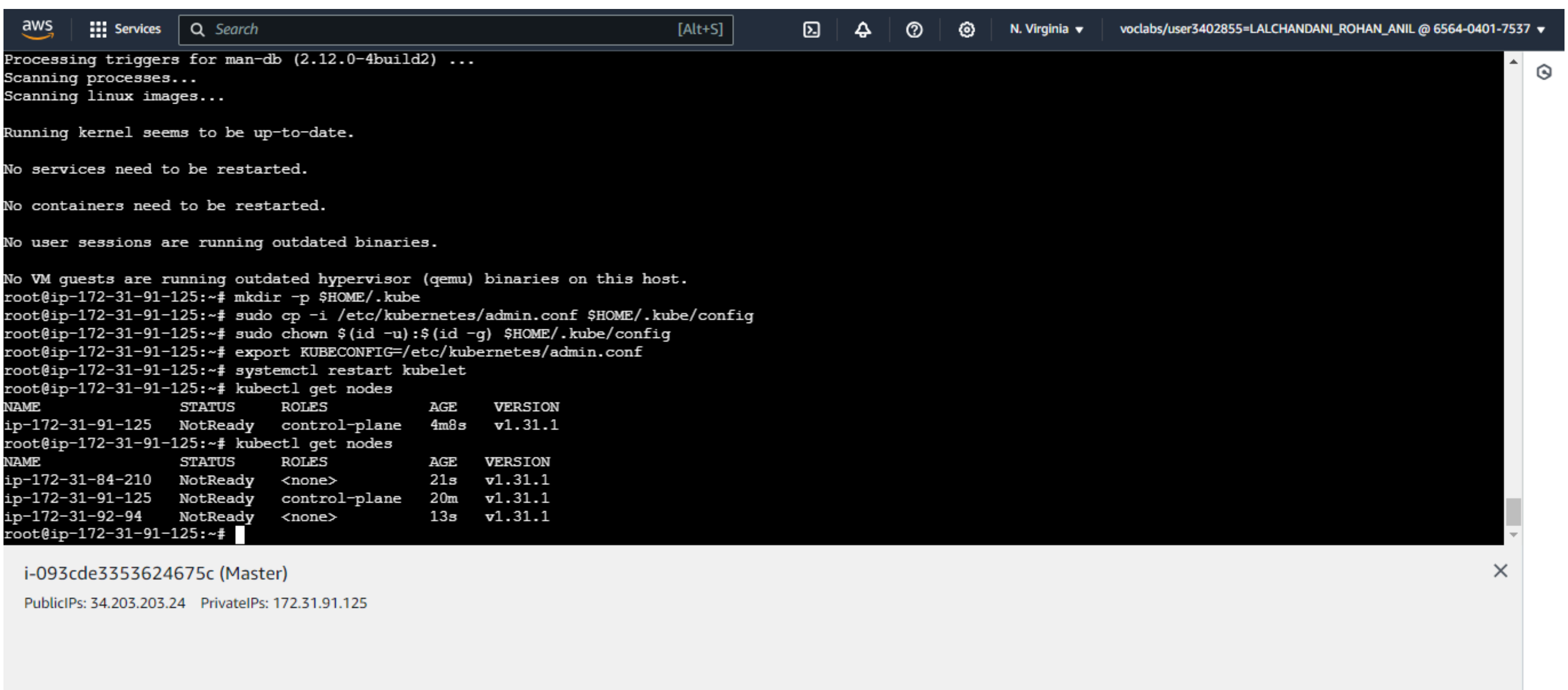
Copy the join command from Master and execute it in Worker 1 and Worker 2 to join them to the cluster.




```
No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@ip-172-31-84-210:~#
kubeadm join 172.31.91.125:6443 --token uue2q7.i3wahpw8072hh8ms \
        --discovery-token-ca-cert-hash sha256:64b48d7aa808c0dafebea2879e1ae9a55e5a58504c9b2dc0871c4af03910b494
[preflight] Running pre-flight checks
[preflight] Reading configuration from the cluster...
[preflight] FYI: You can look at this config file with 'kubectl -n kube-system get cm kubeadm-config -o yaml'
[kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/config.yaml"
[kubelet-start] Writing kubelet environment file with flags to file "/var/lib/kubelet/kubeadm-flags.env"
[kubelet-start] Starting the kubelet
[kubelet-check] Waiting for a healthy kubelet at http://127.0.0.1:10248/healthz. This can take up to 4m0s
[kubelet-check] The kubelet is healthy after 500.852828ms
[kubelet-start] Waiting for the kubelet to perform the TLS Bootstrap

This node has joined the cluster:
* Certificate signing request was sent to apiserver and a response was received.
* The Kubelet was informed of the new secure connection details.

Run 'kubectl get nodes' on the control-plane to see this node join the cluster.

root@ip-172-31-84-210:~#
```

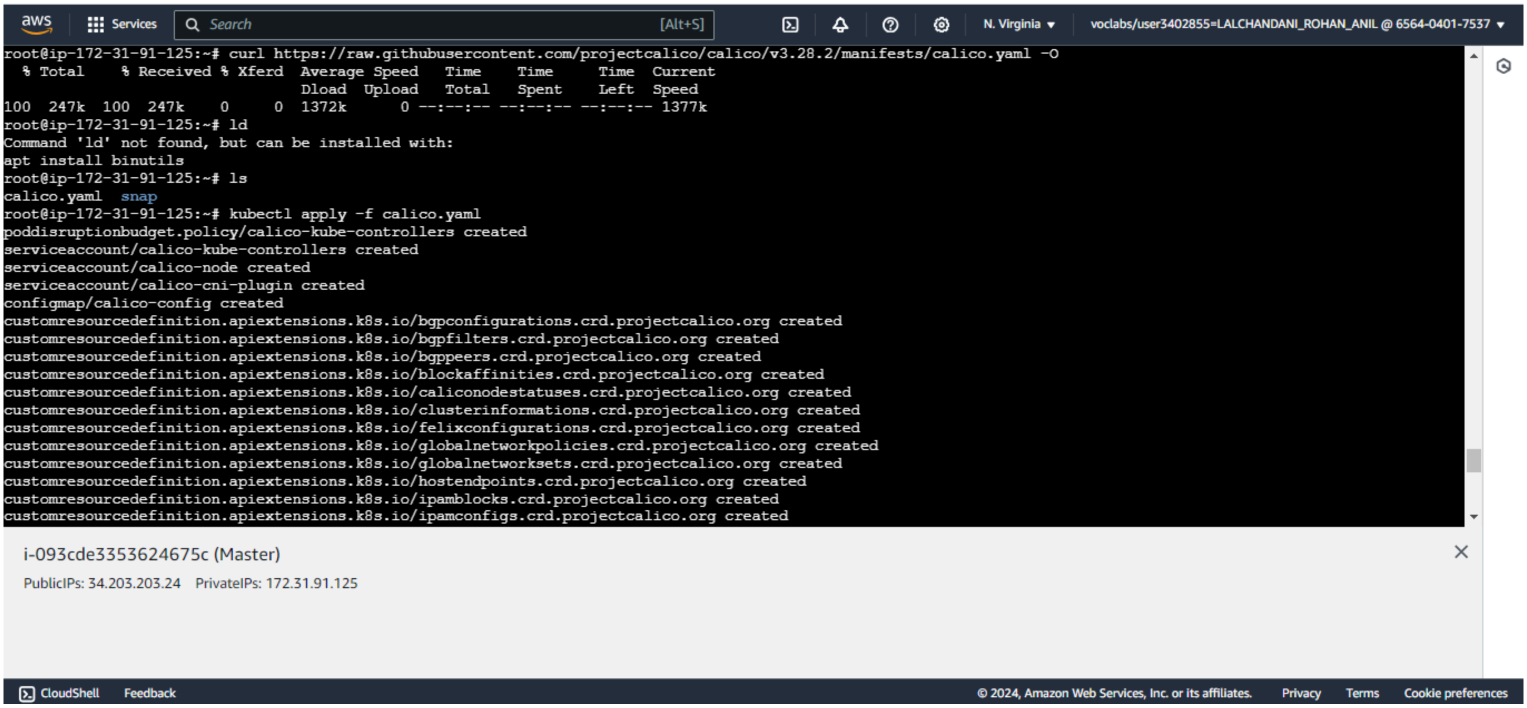


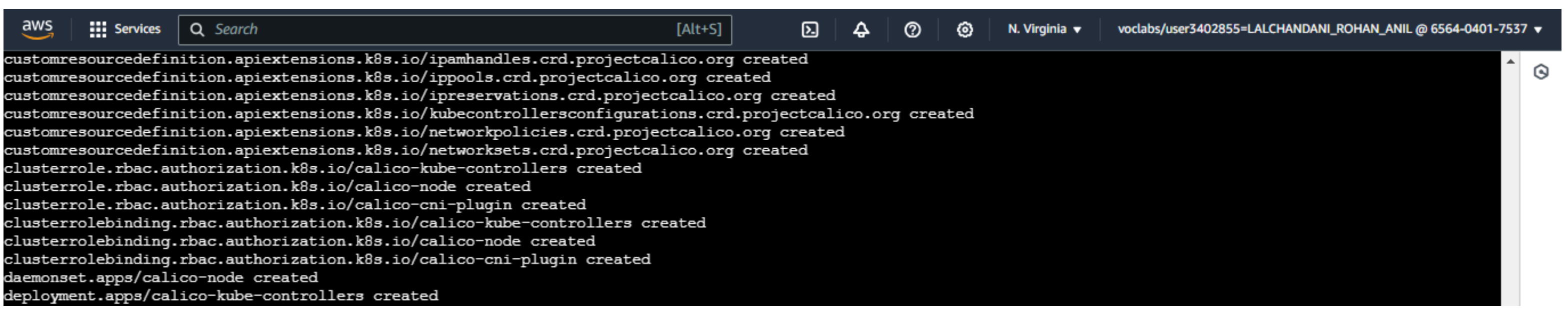
Again execute kubectl get nodes to see if they have joined. You can see the status of nodes to be Not Ready, we have to make them Ready.



Now, we have to manifest “calico.yaml” file to make the status ready for that execute:

curl https://raw.githubusercontent.com/projectcalico/calico/v3.28.2/manifests/calico.yaml -O
ls
kubectl apply -f calico.yaml





Now the status becomes ready.
EXTRA: Renaming the role names:

kubectl label nodes <node ip address> node-role.kubernetes.io/<nodename>=<nodename>

Replace node ip address with your worker 1 or 2 ip address.
Type the name of the node in nodename.