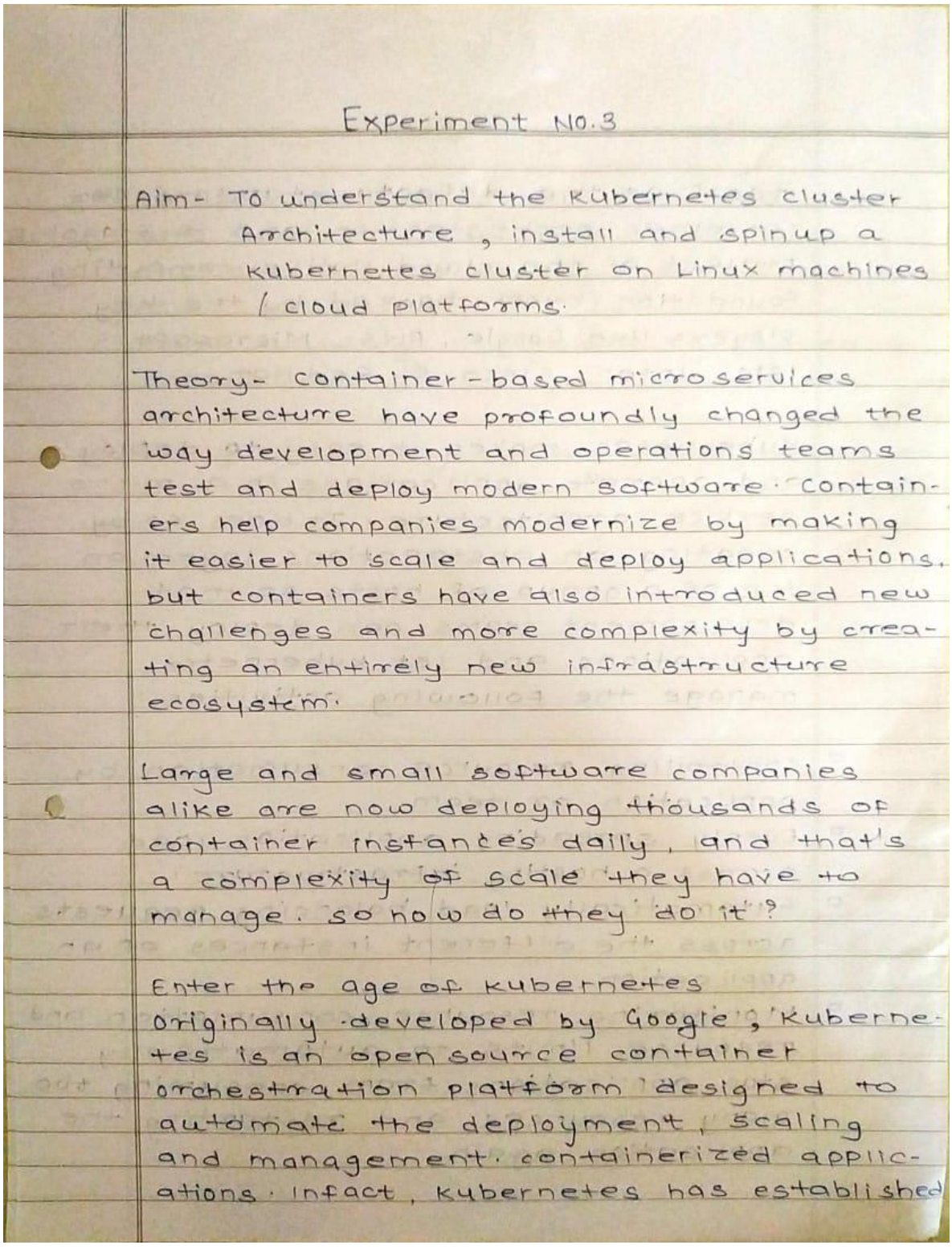
Adv.DevOps Exp 03

Name- Ishika Devare RollNo- 14



Experiment No.3

Aim- To understand the Kubernetes cluster Architecture, install and spinup a Kubernetes cluster on Linux machines / cloud platforms.

Theory- container-based microservices architecture have profoundly changed the way development and operations teams test and deploy modern software. Containers help companies modernize by making it easier to scale and deploy applications. but containers have also introduced new challenges and more complexity by creating an entirely new infrastructure ecosystem.

Large and small software companies alike are now deploying thousands of container instances daily, and that's a complexity of scale they have to manage. so how do they do it?

Enter the age of Kubernetes originally developed by Google, Kubernetes is an open source container orchestration platform designed to automate the deployment, scaling and management. containerized applications. infact, Kubernetes has established

itself as the defacto standard for container orchestration and is flagship project of the cloud Native computing foundation (CNCF), backed by the key players like Google, AWS, Microsoft, IBM, Intel, Cisco & Red Hat.

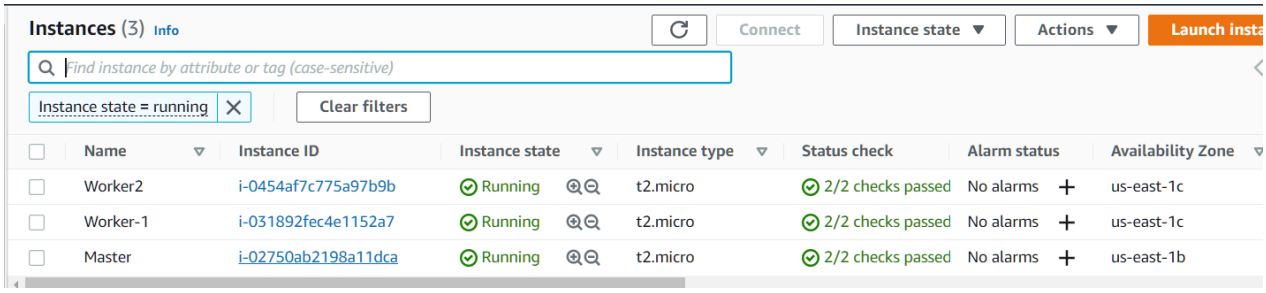
Kubernetes makes it easy to deploy and operate applications in a micro service architecture. It does so by creating an abstractions layer on top of a group of hosts so that development teams can deploy their applications and let kubernetes manage the following activities:

- Controlling resource consumption by application or team.
- Evenly spreading application load across a hosting infrastructure.
- Automatically load balancing requests across the different instances of an application.
- Monitoring resource consumption and resource limits to automatically stop applications from consuming too many resources and restarting the application again.

- Moving an application instance from one host to another if there is a shortage of resources in a host, or if the host dies.
- Automatically load balancing requests across the different instances of an application.
- Easily performing canary deployments and rollbacks.

Implementation:

Create 3 EC2 Ubuntu Instances of Ubuntu version 20.04 and keep all theinstances in the same security group on AWS.
(Name 1 as Master, the other 2 as worker-1 and worker-2)



Steps to Install Kubernetes on UbuntuStep 1:
Install Docker
1. Update the package list with the command:
$sudo apt-get update

Master

```
See "man sudo_root" for details.

ubuntu@ip-172-31-81-188:~$ sudo apt-get update
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal-updates InRelease [114 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal-backports InRelease [108 kB]
Get:4 http://security.ubuntu.com/ubuntu focal-security InRelease [114 kB]
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal/universe amd64 Packages [8628 kB]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal/universe Translation-en [5124 kB]
Get:7 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal/universe amd64 c-n-f Metadata [265 kB]
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal/multiverse amd64 Packages [144 kB]
Get:9 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal/multiverse Translation-en [104 kB]
Get:10 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal/multiverse amd64 c-n-f Metadata [9136 B]
```

Worker-1

```
ubuntu@ip-172-31-23-53:~$ sudo apt-get update
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal-updates InRelease [114 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal-backports InRelease [108 kB]
Get:4 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal/universe amd64 Packages [8628 kB]
Get:5 http://security.ubuntu.com/ubuntu focal-security InRelease [114 kB]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal/universe Translation-en [5124 kB]
Get:7 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal/universe amd64 c-n-f Metadata [265 kB]
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal/multiverse amd64 Packages [144 kB]
Get:9 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal/multiverse Translation-en [104 kB]
Get:10 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal/multiverse amd64 c-n-f Metadata [9136 B]
Get:11 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal-updates/main amd64 Packages [2080 kB]
Get:12 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal-updates/main Translation-en [370 kB]
```

Worker-2

```
ubuntu@ip-172-31-21-143:~$ sudo apt-get update
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal-updates InRelease [114 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal-backports InRelease [108 kB]
Get:4 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal/universe amd64 Packages [8628 kB]
Get:5 http://security.ubuntu.com/ubuntu focal-security InRelease [114 kB]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal/universe Translation-en [5124 kB]
Get:7 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal/universe amd64 c-n-f Metadata [265 kB]
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal/multiverse amd64 Packages [144 kB]
```

2. Next, **install Docker** with the command:
$sudo apt-get install docker.io

Master

```
ubuntu@ip-172-31-81-188:~$ sudo apt-get install docker.io
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  bridge-utils containerd dns-root-data dnsmasq-base libidn11 pigz runc ubuntu-fan
Suggested packages:
  ifupdown aufs-tools cgroupfs-mount | cgroup-lite debootstrap docker-doc rinse zfs-fuse | zfsutils
The following NEW packages will be installed:
  bridge-utils containerd dns-root-data dnsmasq-base docker.io libidn11 pigz runc ubuntu-fan
0 upgraded, 9 newly installed, 0 to remove and 62 not upgraded.
```

3. Repeat the process on each server that will act as a node.

Worker 1

```
ubuntu@ip-172-31-23-53:~$ sudo apt-get install docker.io
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  bridge-utils containerd dns-root-data dnsmasq-base libidn11 pigz runc ubuntu-fan
Suggested packages:
  ifupdown aufs-tools cgroupfs-mount | cgroup-lite debootstrap docker-doc rinse zfs-fuse | zfsutils
The following NEW packages will be installed:
  bridge-utils containerd dns-root-data dnsmasq-base docker.io libidn11 pigz runc ubuntu-fan
0 upgraded, 9 newly installed, 0 to remove and 62 not upgraded.
Need to get 69.2 MB of archives.
After this operation, 334 MB of additional disk space will be used.
Do you want to continue? [Y/n] Y
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal/universe amd64 pigz amd64 2.4-1 [57.4 kB]
```

Worker 2

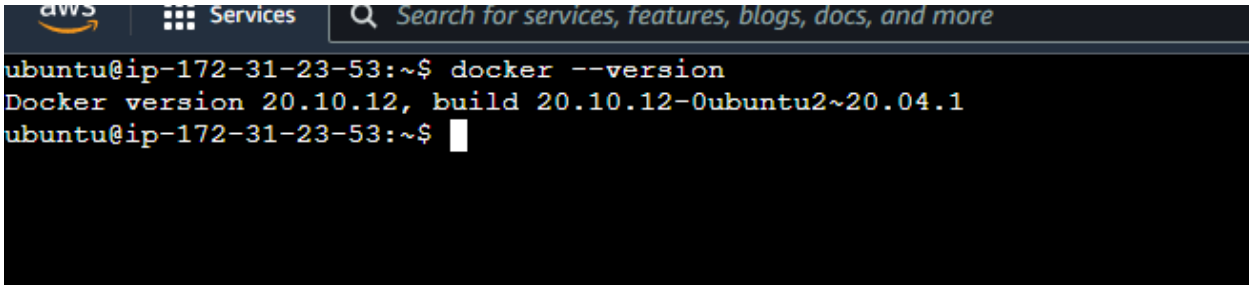
```
ubuntu@ip-172-31-21-143:~$ sudo apt-get install docker.io
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  bridge-utils containerd dns-root-data dnsmasq-base libidn11 pigz runc ubuntu-fan
Suggested packages:
  ifupdown aufs-tools cgroupfs-mount | cgroup-lite debootstrap docker-doc rinse zfs-fuse | zfsutils
The following NEW packages will be installed:
  bridge-utils containerd dns-root-data dnsmasq-base docker.io libidn11 pigz runc ubuntu-fan
0 upgraded, 9 newly installed, 0 to remove and 62 not upgraded.
Need to get 69.2 MB of archives.
After this operation, 334 MB of additional disk space will be used.
Do you want to continue? [Y/n] Y
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal/universe amd64 pigz amd64 2.4-1 [57.4 kB]
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal/main amd64 bridge-utils amd64 1.6-2ubuntu1
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal-updates/main amd64 runc amd64 1.1.0-0ubunt
Get:4 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal-updates/main amd64 containerd amd64 1.5.9-
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal/main amd64 dns-root-data all 2019052802 [5
```

4. Check the installation (and version) by entering the following:
$docker --version

Master

```
ubuntu@ip-172-31-81-188:~$ docker --version
Docker version 20.10.12, build 20.10.12-0ubuntu2~20.04.1
ubuntu@ip-172-31-81-188:~$
```

Worker1



Worker2

```
ubuntu@ip-172-31-21-143:~$ docker --version
Docker version 20.10.12, build 20.10.12-0ubuntu2~20.04.1
ubuntu@ip-172-31-21-143:~$
```

Step 2: Start and Enable Docker

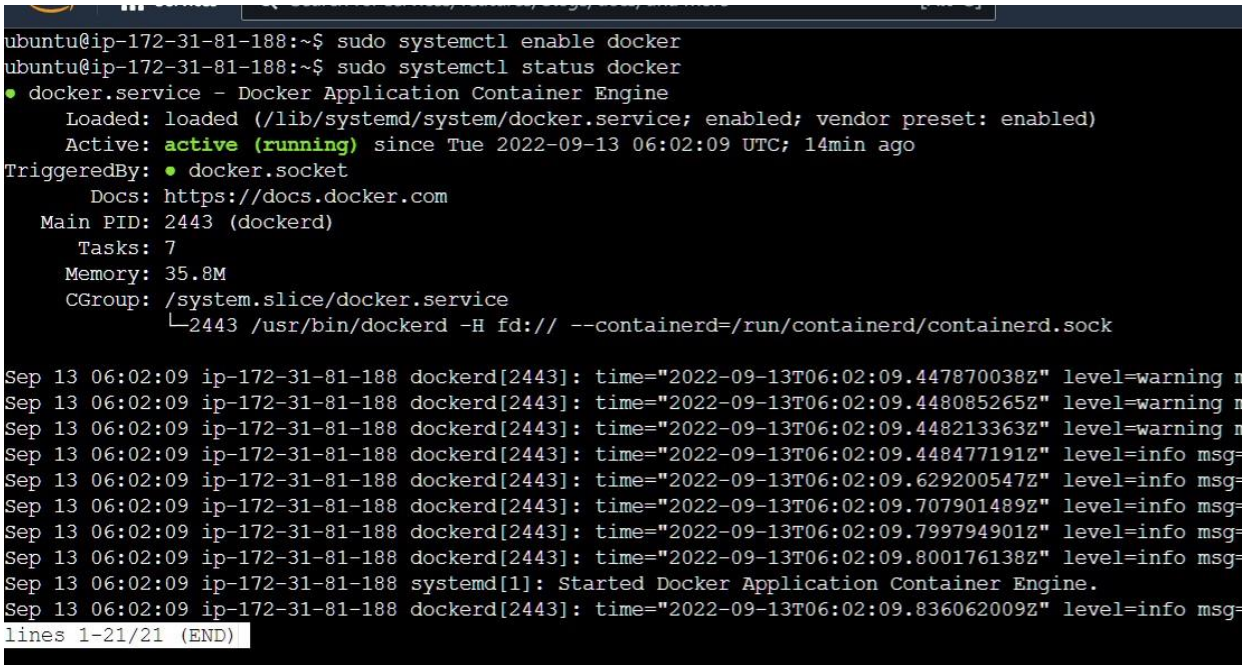
1. Set Docker to launch at boot by entering the following:

$sudo systemctl enable docker

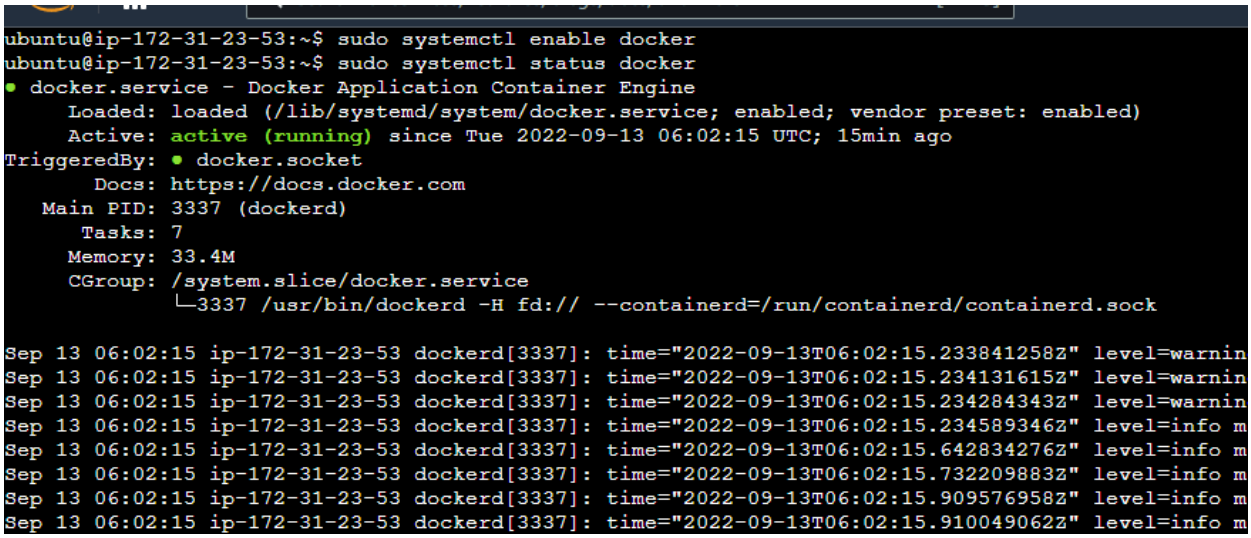
2. Verify Docker is running:

$sudo systemctl status docker

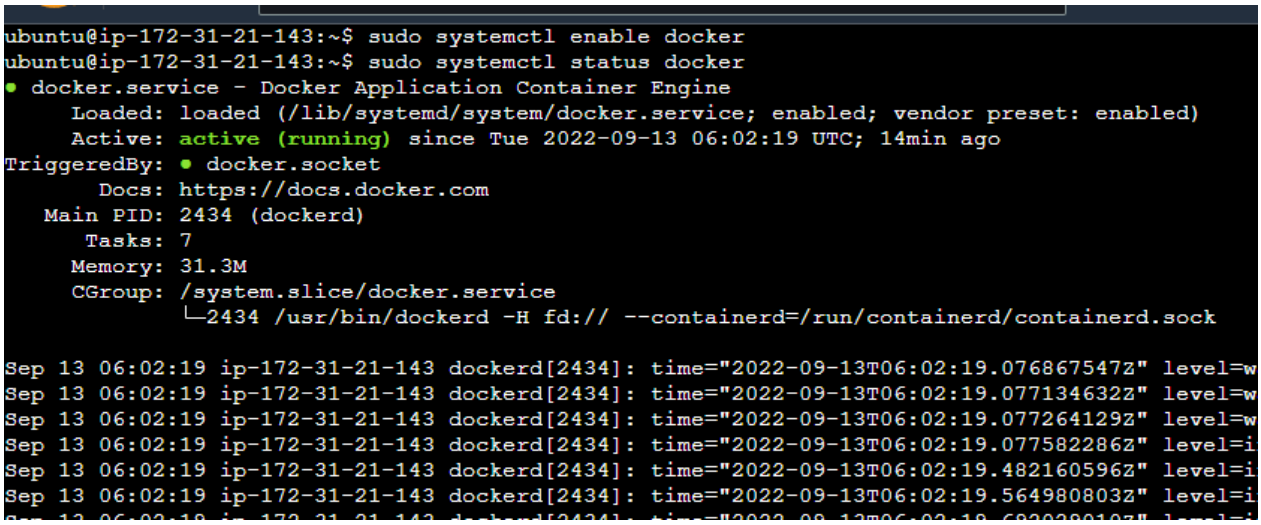
Master



Worker 1



Worker 2

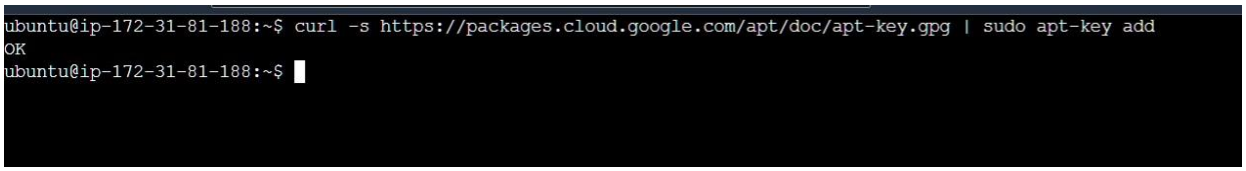


Step 3: Add Kubernetes Signing Key

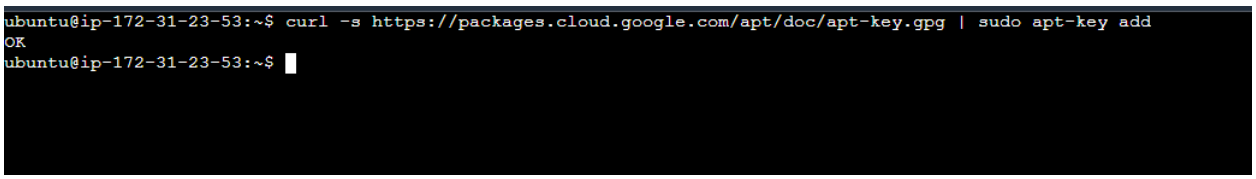
1. Enter the following to add a signing key:

$curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | sudo apt-key add

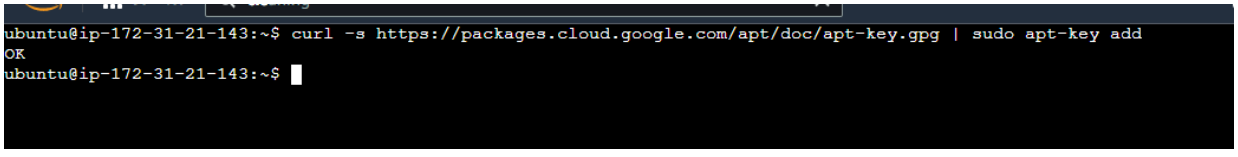
Master



Worker1



Worker2

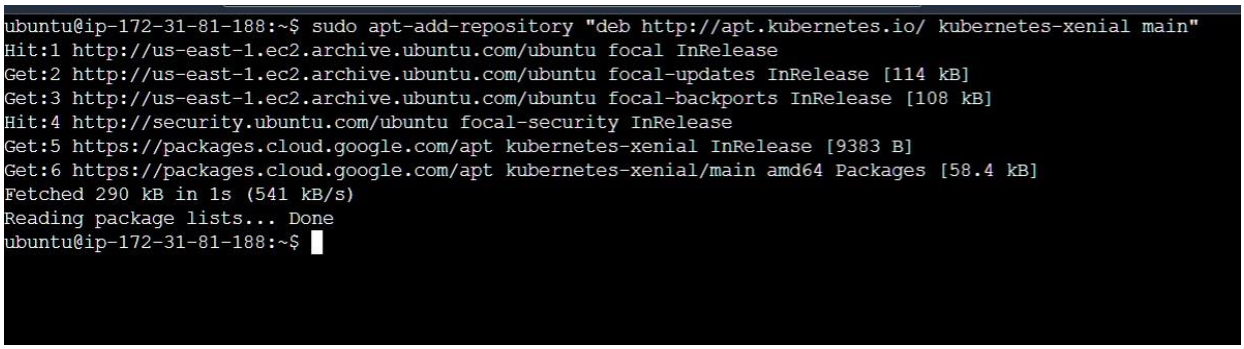


Step 4: Add Software Repositories

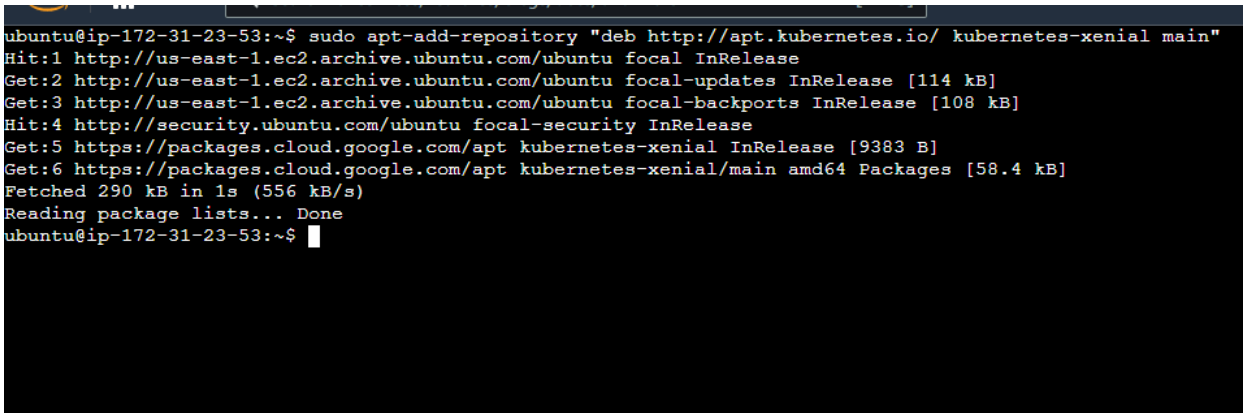
Kubernetes is not included in the default repositories. To add them, enterthe following:

$sudo apt-add-repository "deb http://apt.kubernetes.io/ kubernetes-xenial main"

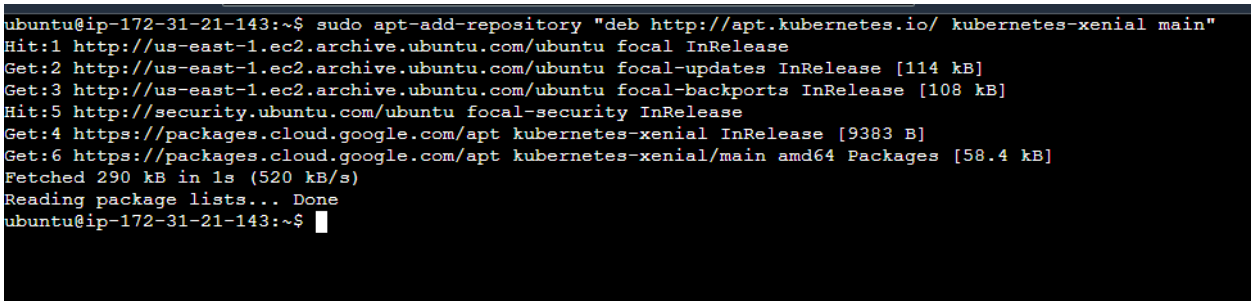
Master



Worker1



Worker2



Step 5: Kubernetes Installation Tools

1. Install Kubernetes tools with the command:

$sudo apt-get install kubeadm kubelet kubectl -y

Master

```
ubuntu@ip-172-31-81-188:~$ sudo apt-get install kubeadm kubelet kubectl -y
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  conntrack cri-tools ebtables kubernetes-cni socat
Suggested packages:
  nftables
The following NEW packages will be installed:
  conntrack cri-tools ebtables kubeadm kubectl kubelet kubernetes-cni socat
0 upgraded, 8 newly installed, 0 to remove and 62 not upgraded.
Need to get 75.9 MB of archives.
After this operation, 310 MB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal/main amd64 conntrack amd64 1:1.4.5-2
```

Worker1

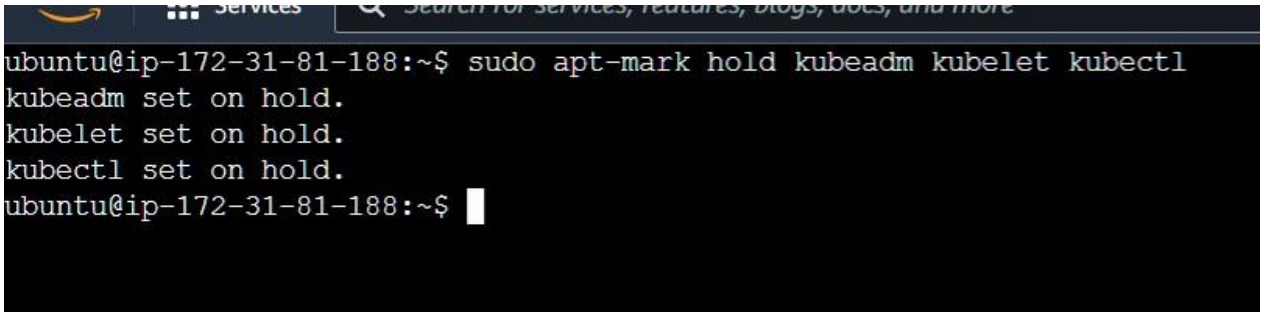
```
ubuntu@ip-172-31-23-53:~$ sudo apt-get install kubeadm kubelet kubectl -y
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  conntrack cri-tools ebtables kubernetes-cni socat
Suggested packages:
  nftables
The following NEW packages will be installed:
  conntrack cri-tools ebtables kubeadm kubectl kubelet kubernetes-cni socat
0 upgraded, 8 newly installed, 0 to remove and 62 not upgraded.
Need to get 75.9 MB of archives.
After this operation, 310 MB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal/main amd64 conntrack amd64 1:1.4.5-2 [30.3 kB
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal/main amd64 ebtables amd64 2.0.11-3build1 [80.
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal/main amd64 socat amd64 1.7.3.3-2 [323 kB]
Get:4 https://packages.cloud.google.com/apt kubernetes-xenial/main amd64 cri-tools amd64 1.24.2-00 [12.3
Get:5 https://packages.cloud.google.com/apt kubernetes-xenial/main amd64 kubernetes-cni amd64 0.8.7-00 [
Get:6 https://packages.cloud.google.com/apt kubernetes-xenial/main amd64 kubelet amd64 1.25.0-00 [19.5 M
Get:7 https://packages.cloud.google.com/apt kubernetes-xenial/main amd64 kubectl amd64 1.25.0-00 [9500 k
```

Worker2

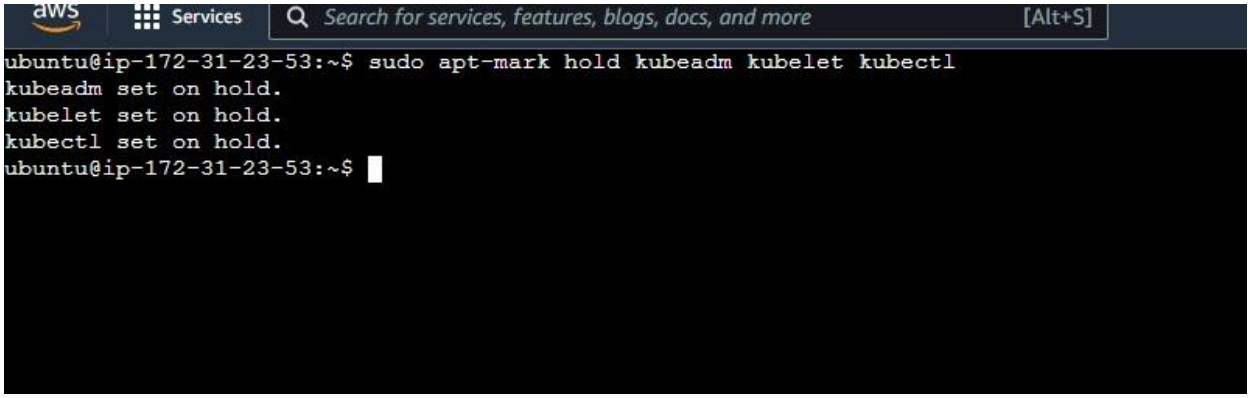
```
ubuntu@ip-172-31-21-143:~$ sudo apt-get install kubeadm kubelet kubectl -y
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  conntrack cri-tools ebtables kubernetes-cni socat
Suggested packages:
  nftables
The following NEW packages will be installed:
  conntrack cri-tools ebtables kubeadm kubectl kubelet kubernetes-cni socat
0 upgraded, 8 newly installed, 0 to remove and 62 not upgraded.
Need to get 75.9 MB of archives.
After this operation, 310 MB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal/main amd64 conntrack amd64 1:
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal/main amd64 ebtables amd64 2.0
```

$sudo apt-mark hold kubeadm kubelet kubectl

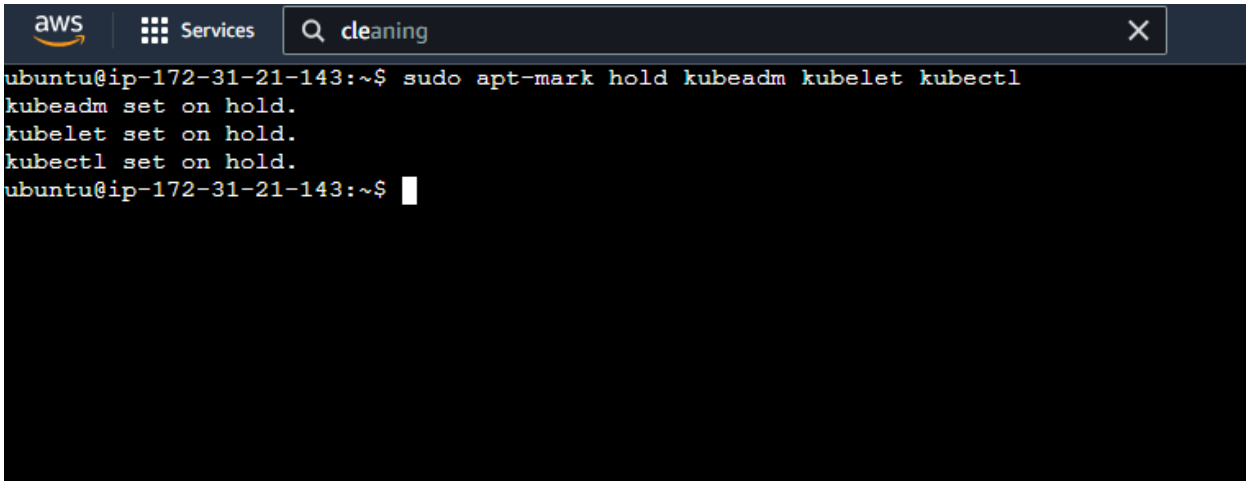
Master



Worker1



Worker2



2. Verify the installation with:

$kubeadm version

Master

Worker1

Worker2

Kubernetes Deployment

Step 6: Begin Kubernetes Deployment

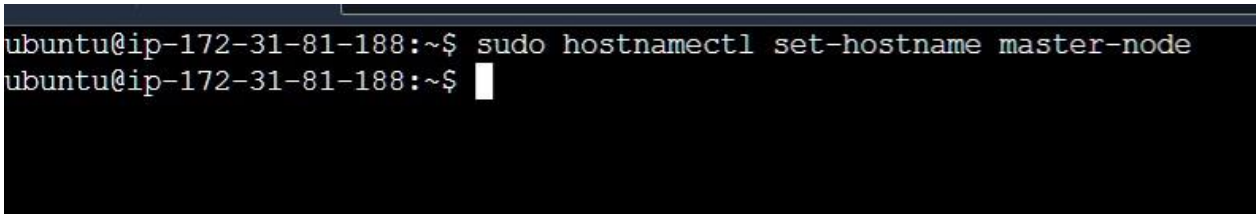
Start by disabling the swap memory on each machine:

$sudo swapoff --a

Step 7: Assign Unique Hostname for Each Server Node

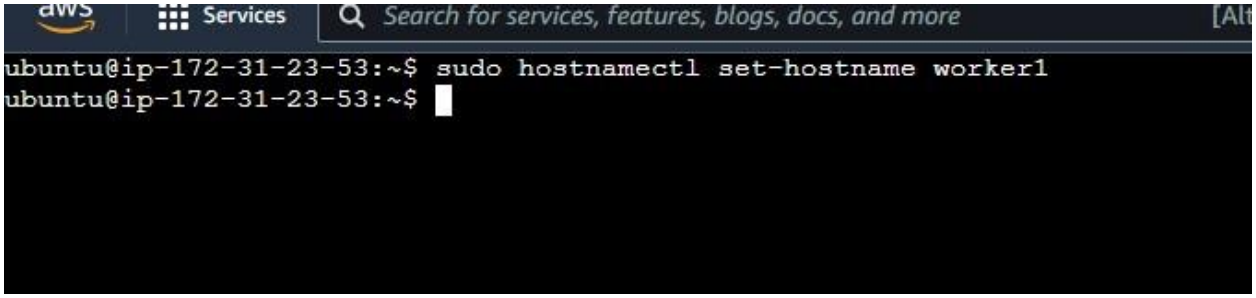
Decide which server to set as the master node. Then enter the command:

$sudo hostnamectl set-hostname master-node

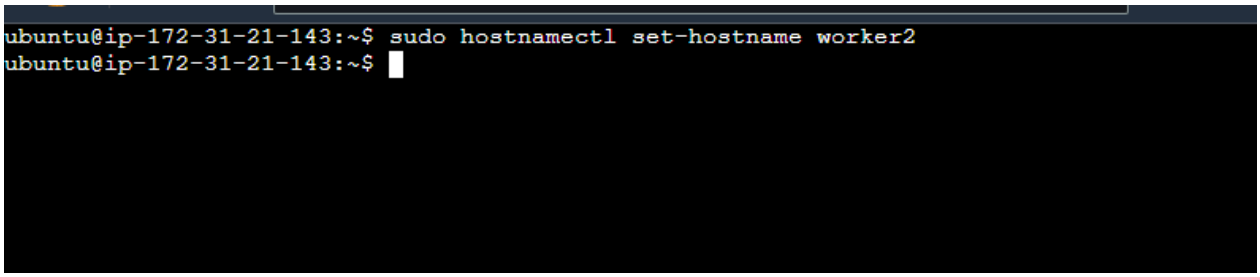


Next, set a worker node hostname by entering the following on the workerserver:

$sudo hostnamectl set-hostname worker1



$sudo hostnamectl set-hostname worker2



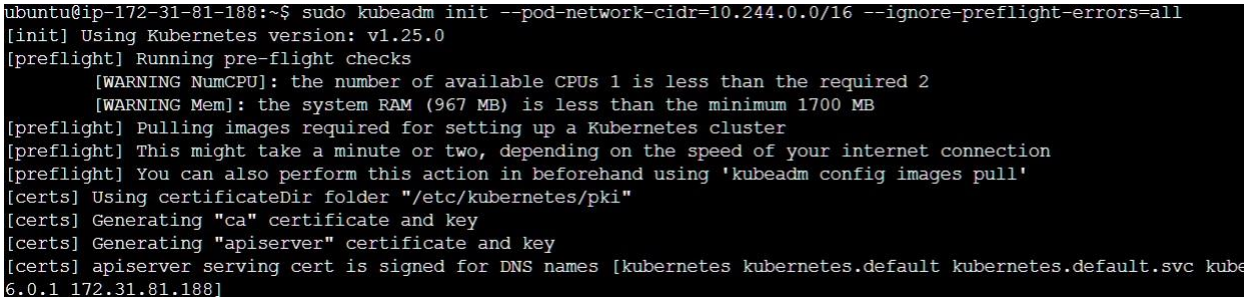
Note - Perform the next two steps i.e. Step 8 and Step 9 only on theMaster machine.

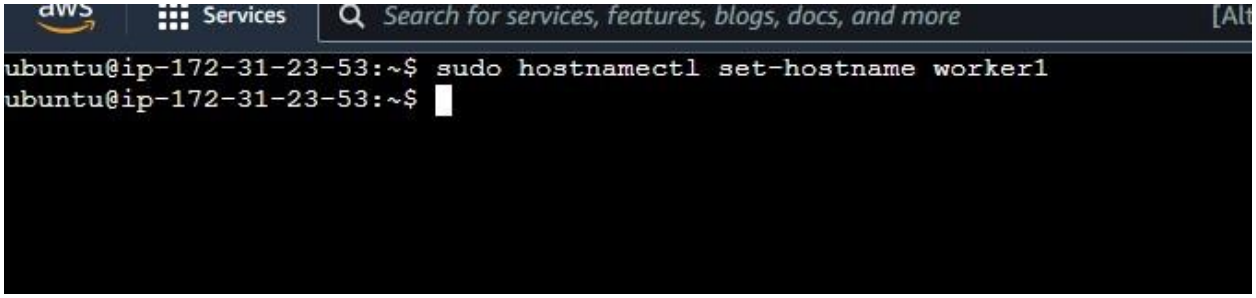
Step 8: Initialize Kubernetes only on Master Node

Switch to the master server node, and enter the following:

$sudo kubeadm init --pod-network-cidr=10.244.0.0/16 --ignore-preflight-errors=all

Once this command finishes, it will display a kubeadm join message at the end. Make a note of the whole entry. This will be used to join the worker nodes to the cluster.





Next, enter the following to create a directory for the cluster:

$ mkdir -p $HOME/.kube

$ sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config

$ sudo chown $(id -u):$(id -g) $HOME/.kube/config

Step 9: Deploy Pod Network to Cluster

A Pod Network is a way to allow communication between different nodes in the cluster. This tutorial uses the flannel virtual network.

Enter the following:

$ **kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml**

Allow the process to complete.

Verify that everything is running and communicating:

$ kubectl get pods --all-namespaces

```
ubuntu@ip-172-31-81-188:~$ kubectl get pods --all-namespaces
NAMESPACE      NAME                                    READY   STATUS    RESTARTS   AGE
kube-flannel   kube-flannel-ds-d672n                   1/1     Running   0          73s
kube-system    coredns-565d847f94-snh7r                1/1     Running   0          14m
kube-system    coredns-565d847f94-z4h4j                1/1     Running   0          14m
kube-system    etcd-master-node                        1/1     Running   0          15m
kube-system    kube-apiserver-master-node              1/1     Running   0          15m
kube-system    kube-controller-manager-master-node     1/1     Running   0          15m
kube-system    kube-proxy-zvt25                        1/1     Running   0          14m
kube-system    kube-scheduler-master-node              1/1     Running   0          15m
ubuntu@ip-172-31-81-188:~$
```

Do this step only on the worker nodes.Step 10:

Join Worker Node to Cluster

As indicated in Step 8, enter the kubeadm join command on each worker node to connect to the cluster. Switch to the **root user** of your worker system and enter the command you noted from Step 8.

$ kubeadm join 172.31.81.188:6443 --token n46tzy.ocnrf7wkiyk0t0xu --discovery-token-ca-cert-hash sha256:59c2fec9fc69aa85d306f8bfcadac2d827699b0db3d87e13192873a1044f86e2 --ignore-preflight-errors=all

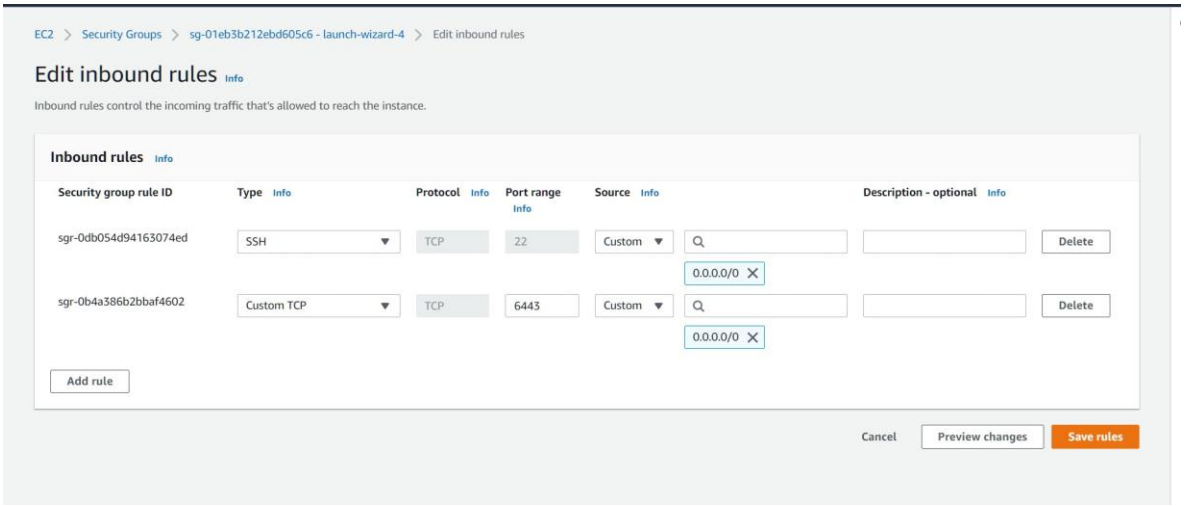
Note - Join command is different for everyone please do not use this.Worker1

```
root@worker1:~# kubeadm join 172.31.81.188:6443 --token n46tzy.ocnrf7wkiyk0t0xu        --discovery-token-ca-cert-hash sha256:59c2fec9fc69aa85d306f8bf
cadac2d827699b0db3d87e13192873a1044f86e2 --ignore-preflight-errors=all
[preflight] Running pre-flight checks
error execution phase preflight: couldn't validate the identity of the API Server: Get "https://172.31.81.188:6443/api/v1/namespaces/kube-public/confi
gmaps/cluster-info?timeout=10s": net/http: request canceled while waiting for connection (Client.Timeout exceeded while awaiting headers)
To see the stack trace of this error execute with --v=5 or higher
root@worker1:~# kubeadm join 172.31.81.188:6443 --token n46tzy.ocnrf7wkiyk0t0xu        --discovery-token-ca-cert-hash sha256:59c2fec9fc69aa85d306f8bf
cadac2d827699b0db3d87e13192873a1044f86e2 --ignore-preflight-errors=all
[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-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.
```

Worker 2

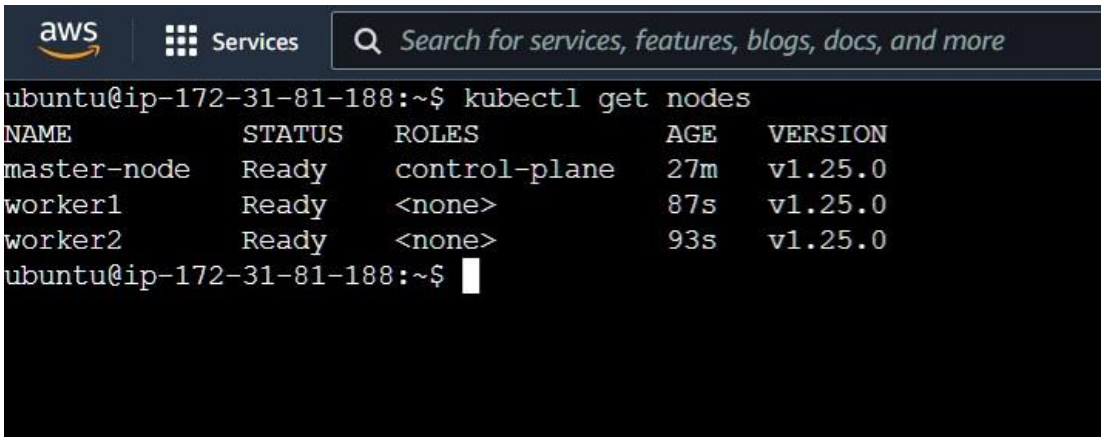
```
root@worker2:~# kubeadm join 172.31.81.188:6443 --token n46tzy.ocnrf7wkiyk0t0xu        --discovery-token-ca-cert-hash sha256:59c2fec9fc69aa85d306f8b
cadac2d827699b0db3d87e13192873a1044f86e2 --ignore-preflight-errors=all
[preflight] Running pre-flight checks
error execution phase preflight: couldn't validate the identity of the API Server: Get "https://172.31.81.188:6443/api/v1/namespaces/kube-public/conf
gmaps/cluster-info?timeout=10s": net/http: request canceled while waiting for connection (Client.Timeout exceeded while awaiting headers)
To see the stack trace of this error execute with --v=5 or higher
root@worker2:~# kubeadm join 172.31.81.188:6443 --token n46tzy.ocnrf7wkiyk0t0xu        --discovery-token-ca-cert-hash sha256:59c2fec9fc69aa85d306f8b
cadac2d827699b0db3d87e13192873a1044f86e2 --ignore-preflight-errors=all
[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-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@worker2:~#
```

If you are getting an error of port “6443” while joining to master then editthe inbound rules of your security group and add port 6443.



Switch to the master server, and enter:

$ **kubectl get nodes**



Conclusion - We have successfully created 3 instances as Master, Worker1 and Worker 2 on which we installed docker and started docker on all 3 machines. Then add kubernetes signing key, installed kubernetes tools and done the kubernetes deployment and also deployed all pod networks to cluster and displayed them using "kubectl get nodes".