Name: Manav Jawrani

Roll No.: 19

Subject: Advanced DevOps

Experiment No.: 3

Experiment 3

Aim : To understand the Kubernetes cluster Architecture, install and spin up a Kubernetes cluster on Linux machines / Cloud platforms.

Theory :

- What is Kubernetes ?

  Kubernetes is open source container platform that automates many of the manual processes involved in deploying, managing and scaling containerized applications.

- what is kubernetes cluster ?

  A Kubernetes cluster is a set of nodes that run containerized applications. Containerizing applications packages an app with its dependences and some necessary services. They are more lightweight and flexible than virtual machines. In this way, Kubernetes clusters allow for applications to be more easily developed, moved and managed.

- Features of Kubernetes :

1. continuous development, integration, and deployment.
2. Containerized infrastructure.
3. Self monitoring
4. Run everywhere

- Kubernetes components:

A Kubernetes cluster consists of set of worker machines, called nodes, that run containerized. applications. Every cluster has at least one worker node.

The worker nodes host the Pods that are the components of application workload. The control plane manages the worker nodes and Pods in cluster

- Control Plane components:

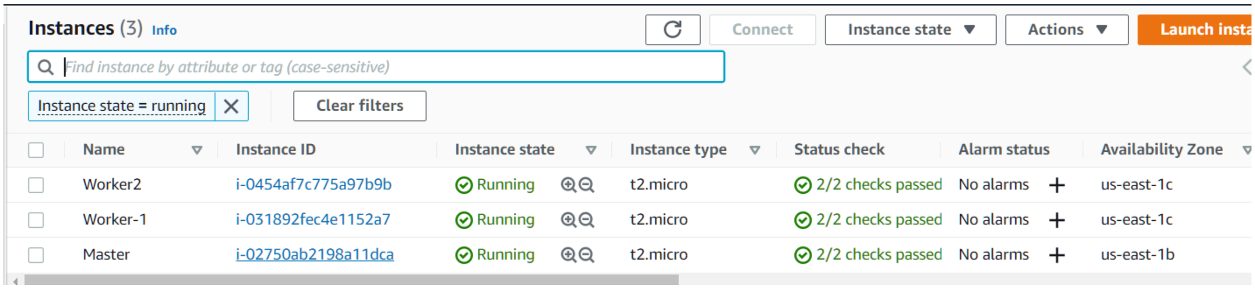
1. Kube-apiserver
2. Kube - controller-manager

- Node components:

1. Kubelet
2. Kube -proxy

Implementation:

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



Instance state = running

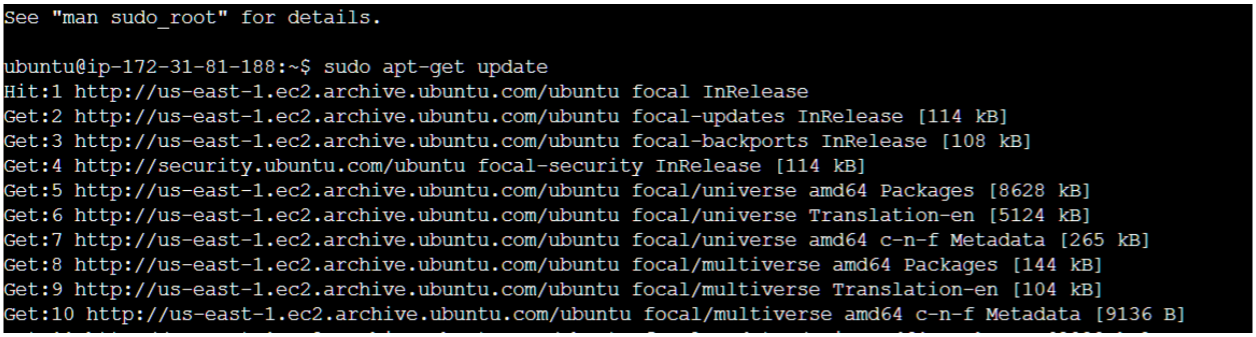
□	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
□	Worker2	i-0454af7c775a97b9b	✓ Running	t2.micro	✓ 2/2 checks passed	No alarms +	us-east-1c
□	Worker-1	i-031892fec4e1152a7	✓ Running	t2.micro	✓ 2/2 checks passed	No alarms +	us-east-1c
□	Master	i-02750ab2198a11dca	✓ Running	t2.micro	✓ 2/2 checks passed	No alarms +	us-east-1b

Steps to Install Kubernetes on Ubuntu

Step 1: Install Docker

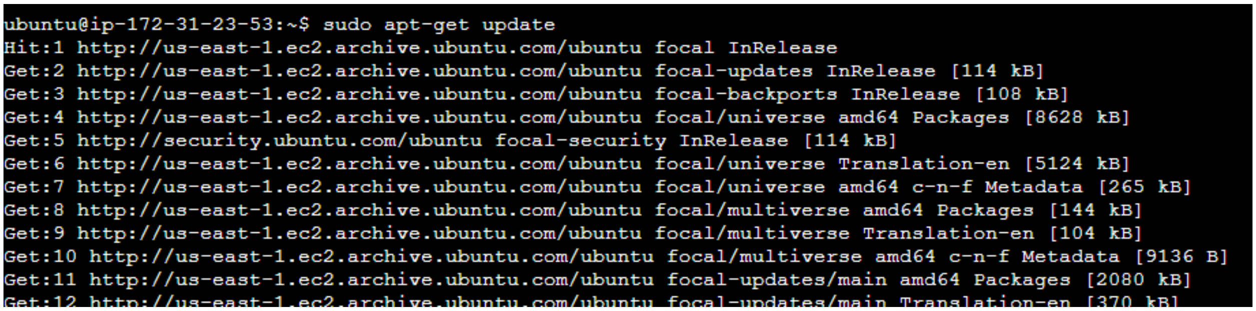
1. Update the package list with the command:
$sudo apt-get update

Master



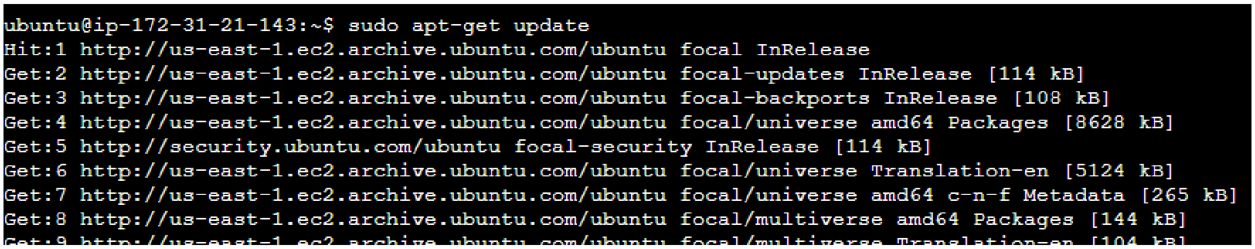
```
See "man sudo_root" for details.

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]
```

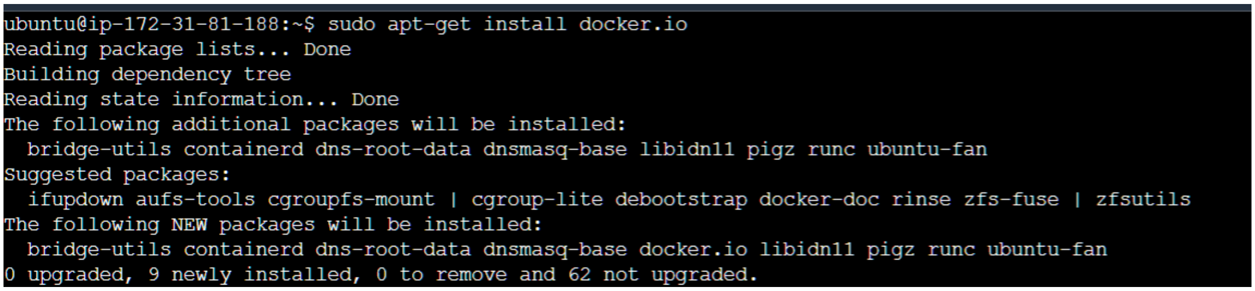
Worker-2



```
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]
Get:9 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal/multiverse Translation-en [104 kB]
```

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

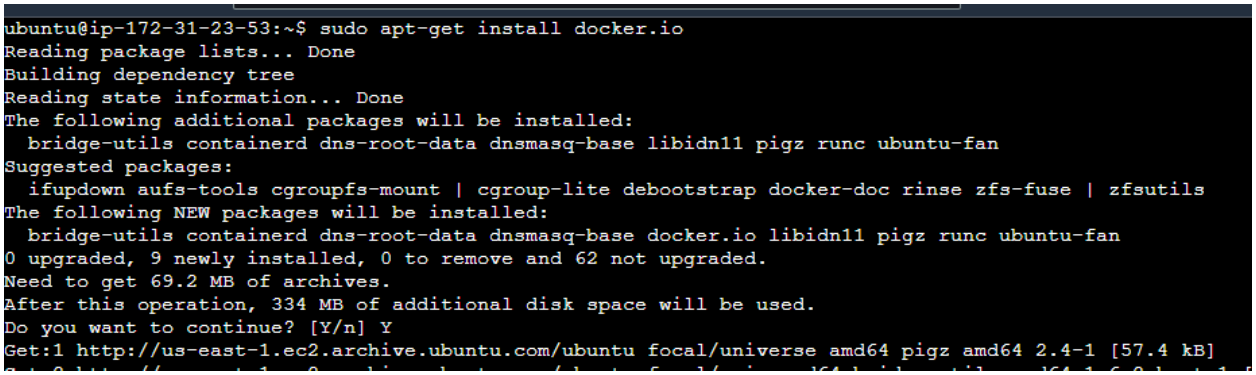
Master



```
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.
```

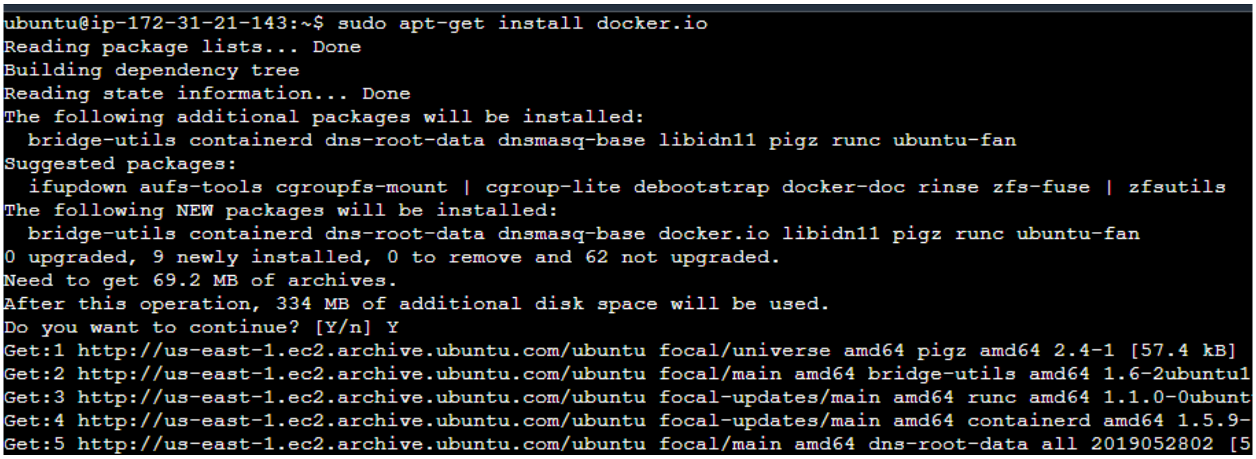
3. Repeat the process on each server that will act as a node.

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]
```

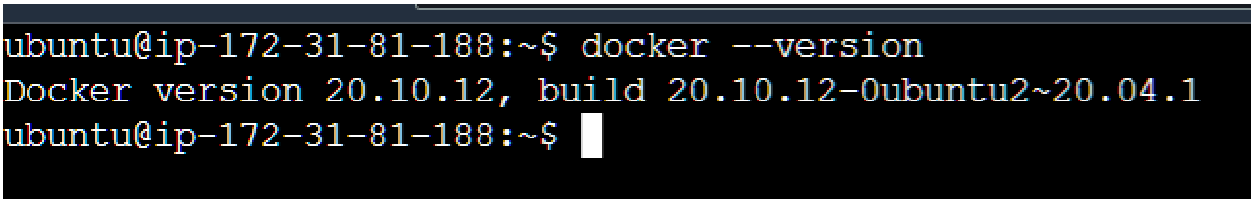
Worker 2



```
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
```

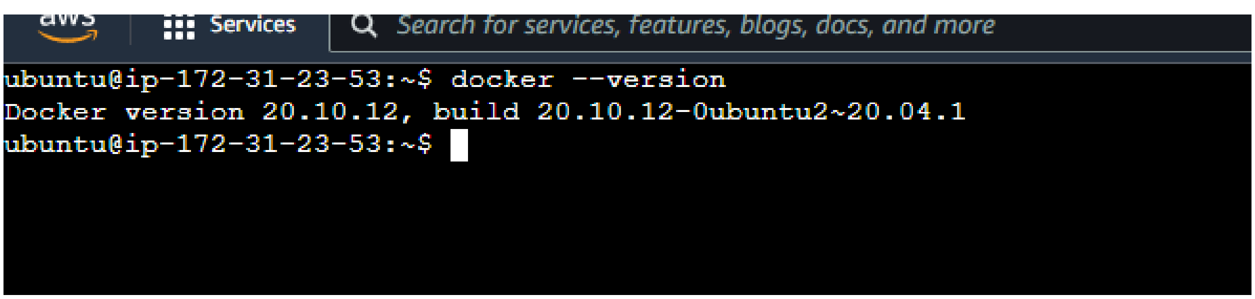
4. Check the installation (and version) by entering the following:
$docker --version

Master



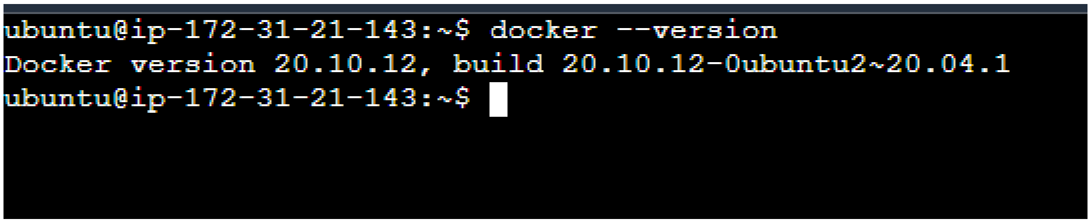
```
ubuntu@ip-172-31-81-188:~$ docker --version
Docker version 20.10.12, build 20.10.12-0ubuntu2~20.04.1
ubuntu@ip-172-31-81-188:~$
```

Worker1



```
ubuntu@ip-172-31-23-53:~$ docker --version
Docker version 20.10.12, build 20.10.12-0ubuntu2~20.04.1
ubuntu@ip-172-31-23-53:~$
```

Worker2



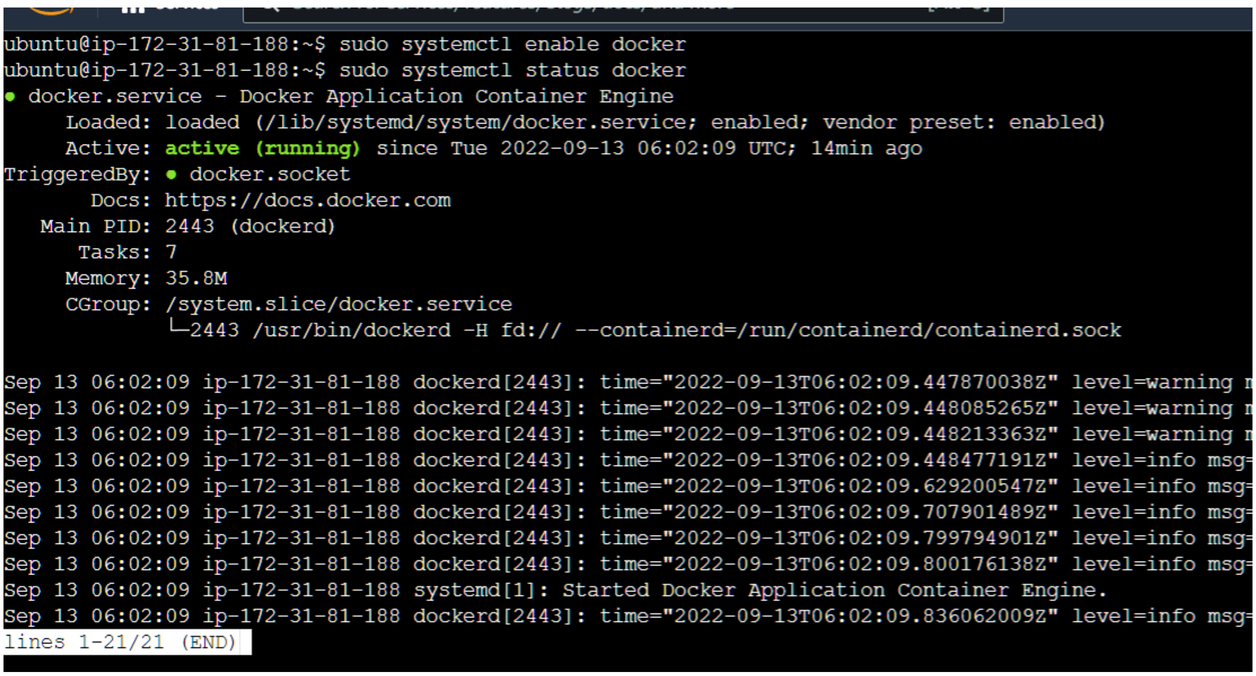
```
ubuntu@ip-172-31-21-143:~$ docker --version
Docker version 20.10.12, build 20.10.12-0ubuntu2~20.04.1
ubuntu@ip-172-31-21-143:~$
```

Step 2: Start and Enable Docker

1. Set Docker to launch at boot by entering the following:
$sudo systemctl enable docker

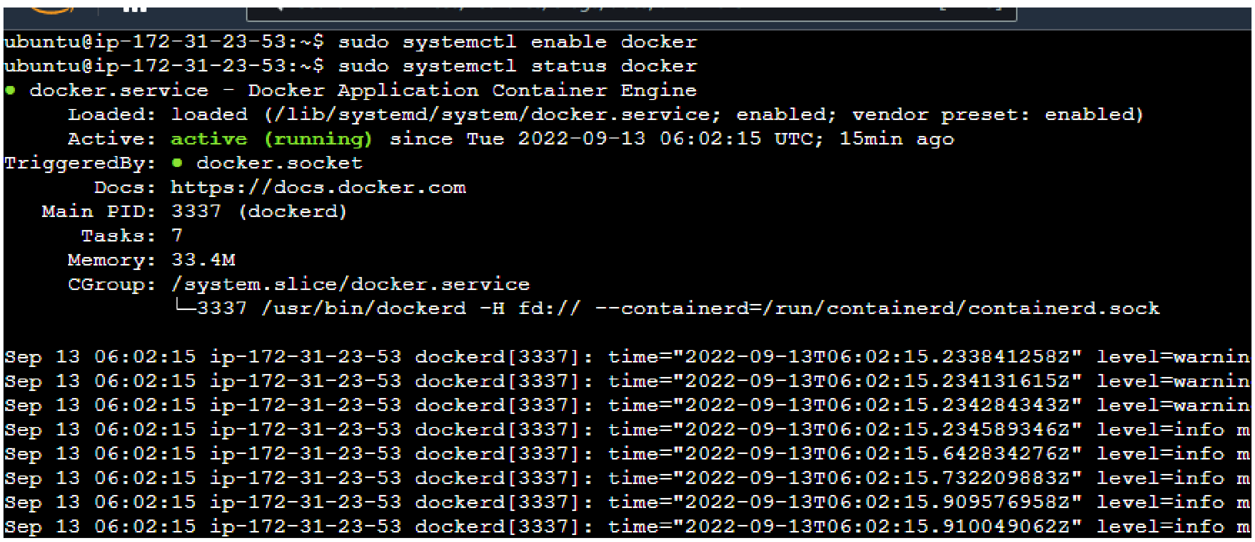
2. Verify Docker is running:
$sudo systemctl status docker

Master



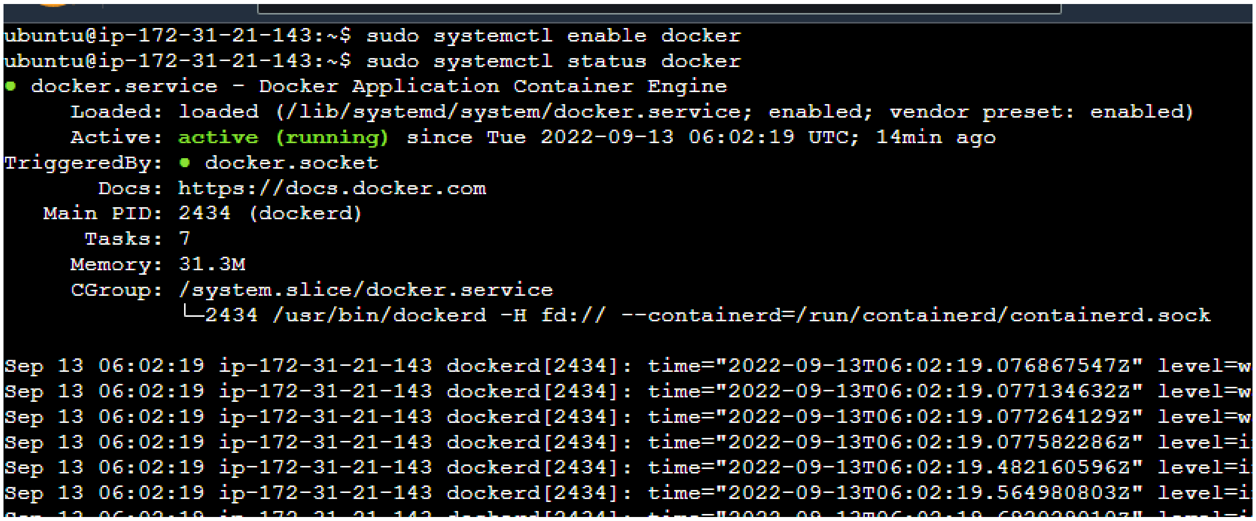
```
ubuntu@ip-172-31-81-188:~$ sudo systemctl enable docker
ubuntu@ip-172-31-81-188:~$ sudo systemctl status docker
● docker.service - Docker Application Container Engine
     Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: enabled)
     Active: active (running) since Tue 2022-09-13 06:02:09 UTC; 14min ago
TriggeredBy: ● docker.socket
       Docs: https://docs.docker.com
   Main PID: 2443 (dockerd)
      Tasks: 7
     Memory: 35.8M
     CGroup: /system.slice/docker.service
             └─2443 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock

Sep 13 06:02:09 ip-172-31-81-188 dockerd[2443]: time="2022-09-13T06:02:09.447870038Z" level=warning m
Sep 13 06:02:09 ip-172-31-81-188 dockerd[2443]: time="2022-09-13T06:02:09.448085265Z" level=warning m
Sep 13 06:02:09 ip-172-31-81-188 dockerd[2443]: time="2022-09-13T06:02:09.448213363Z" level=warning m
Sep 13 06:02:09 ip-172-31-81-188 dockerd[2443]: time="2022-09-13T06:02:09.448477191Z" level=info msg=
Sep 13 06:02:09 ip-172-31-81-188 dockerd[2443]: time="2022-09-13T06:02:09.629200547Z" level=info msg=
Sep 13 06:02:09 ip-172-31-81-188 dockerd[2443]: time="2022-09-13T06:02:09.707901489Z" level=info msg=
Sep 13 06:02:09 ip-172-31-81-188 dockerd[2443]: time="2022-09-13T06:02:09.799794901Z" level=info msg=
Sep 13 06:02:09 ip-172-31-81-188 dockerd[2443]: time="2022-09-13T06:02:09.800176138Z" level=info msg=
Sep 13 06:02:09 ip-172-31-81-188 systemd[1]: Started Docker Application Container Engine.
Sep 13 06:02:09 ip-172-31-81-188 dockerd[2443]: time="2022-09-13T06:02:09.836062009Z" level=info msg=
lines 1-21/21 (END)
```

Worker 1



```
ubuntu@ip-172-31-23-53:~$ sudo systemctl enable docker
ubuntu@ip-172-31-23-53:~$ sudo systemctl status docker
● docker.service - Docker Application Container Engine
     Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: enabled)
     Active: active (running) since Tue 2022-09-13 06:02:15 UTC; 15min ago
TriggeredBy: ● docker.socket
       Docs: https://docs.docker.com
   Main PID: 3337 (dockerd)
      Tasks: 7
     Memory: 33.4M
     CGroup: /system.slice/docker.service
             └─3337 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock

Sep 13 06:02:15 ip-172-31-23-53 dockerd[3337]: time="2022-09-13T06:02:15.233841258Z" level=warnin
Sep 13 06:02:15 ip-172-31-23-53 dockerd[3337]: time="2022-09-13T06:02:15.234131615Z" level=warnin
Sep 13 06:02:15 ip-172-31-23-53 dockerd[3337]: time="2022-09-13T06:02:15.234284343Z" level=warnin
Sep 13 06:02:15 ip-172-31-23-53 dockerd[3337]: time="2022-09-13T06:02:15.234589346Z" level=info m
Sep 13 06:02:15 ip-172-31-23-53 dockerd[3337]: time="2022-09-13T06:02:15.642834276Z" level=info m
Sep 13 06:02:15 ip-172-31-23-53 dockerd[3337]: time="2022-09-13T06:02:15.732209883Z" level=info m
Sep 13 06:02:15 ip-172-31-23-53 dockerd[3337]: time="2022-09-13T06:02:15.909576958Z" level=info m
Sep 13 06:02:15 ip-172-31-23-53 dockerd[3337]: time="2022-09-13T06:02:15.910049062Z" level=info m
```

Worker 2



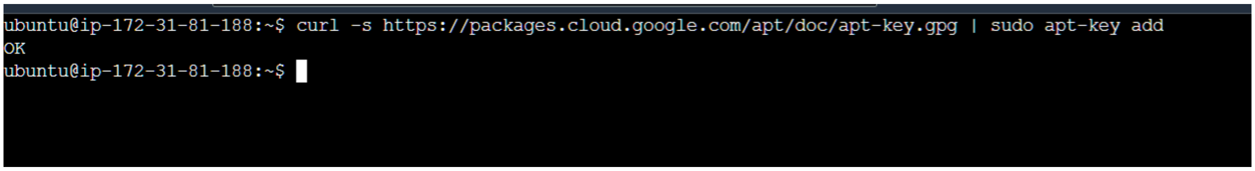
```
ubuntu@ip-172-31-21-143:~$ sudo systemctl enable docker
ubuntu@ip-172-31-21-143:~$ sudo systemctl status docker
● docker.service - Docker Application Container Engine
     Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: enabled)
     Active: active (running) since Tue 2022-09-13 06:02:19 UTC; 14min ago
TriggeredBy: ● docker.socket
       Docs: https://docs.docker.com
   Main PID: 2434 (dockerd)
      Tasks: 7
     Memory: 31.3M
     CGroup: /system.slice/docker.service
             └─2434 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock

Sep 13 06:02:19 ip-172-31-21-143 dockerd[2434]: time="2022-09-13T06:02:19.076867547Z" level=w
Sep 13 06:02:19 ip-172-31-21-143 dockerd[2434]: time="2022-09-13T06:02:19.077134632Z" level=w
Sep 13 06:02:19 ip-172-31-21-143 dockerd[2434]: time="2022-09-13T06:02:19.077264129Z" level=w
Sep 13 06:02:19 ip-172-31-21-143 dockerd[2434]: time="2022-09-13T06:02:19.077582286Z" level=i
Sep 13 06:02:19 ip-172-31-21-143 dockerd[2434]: time="2022-09-13T06:02:19.482160596Z" level=i
Sep 13 06:02:19 ip-172-31-21-143 dockerd[2434]: time="2022-09-13T06:02:19.564980803Z" level=i
```

Step 3: Add Kubernetes Signing Key

1. Enter the following to add a signing key:

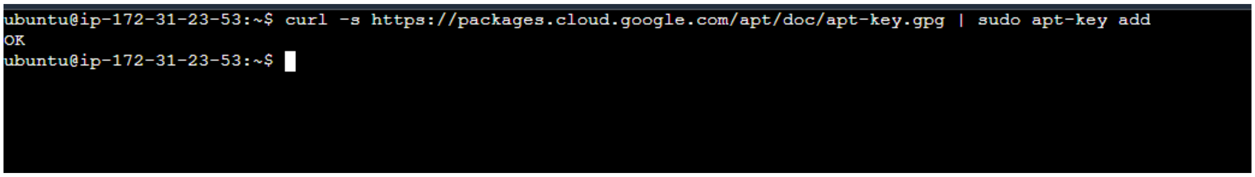
$curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | sudo apt-key add

Master



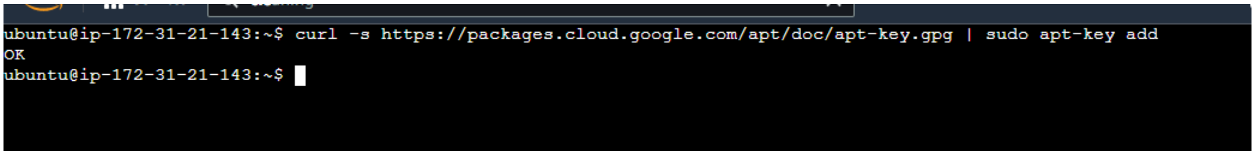
```
ubuntu@ip-172-31-81-188:~$ curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | sudo apt-key add
OK
ubuntu@ip-172-31-81-188:~$ 
```

Worker1



```
ubuntu@ip-172-31-23-53:~$ curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | sudo apt-key add
OK
ubuntu@ip-172-31-23-53:~$ 
```

Worker2



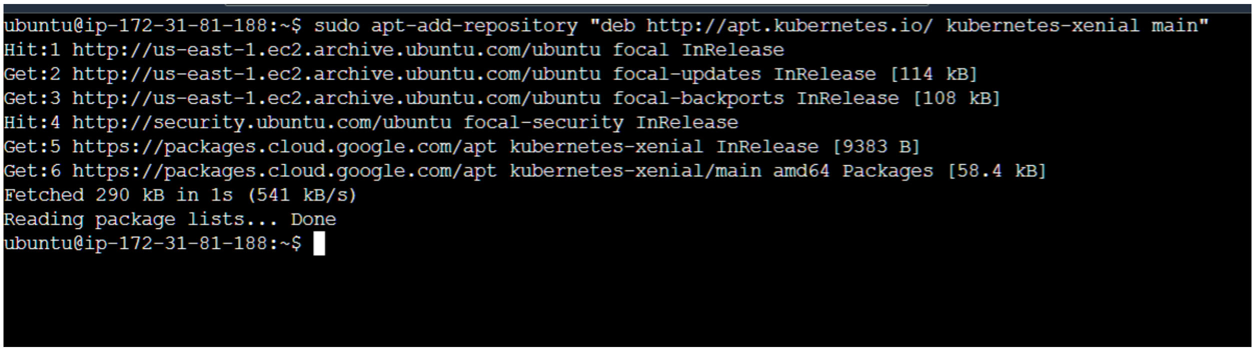
```
ubuntu@ip-172-31-21-143:~$ curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | sudo apt-key add
OK
ubuntu@ip-172-31-21-143:~$ 
```

Step 4: Add Software Repositories

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

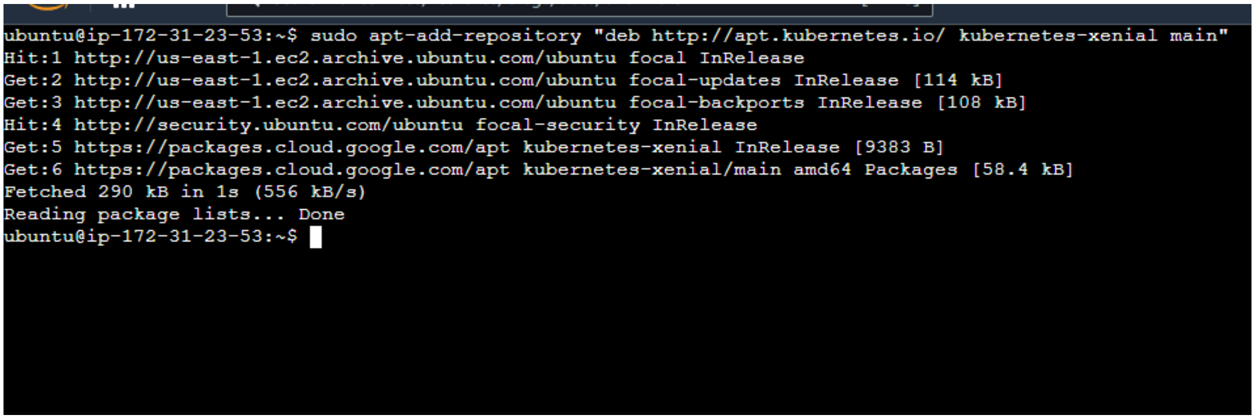
$sudo apt-add-repository "deb http://apt.kubernetes.io/ kubernetes-xenial main"

Master



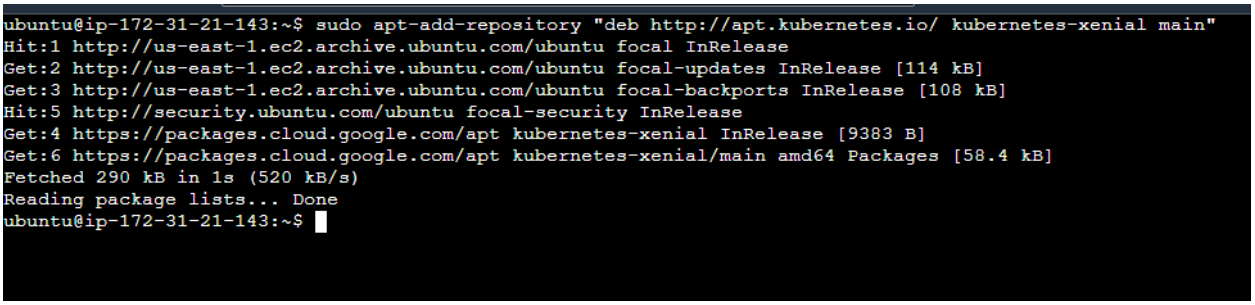
```
ubuntu@ip-172-31-81-188:~$ sudo apt-add-repository "deb http://apt.kubernetes.io/ kubernetes-xenial main"
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal-updates InRelease [114 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal-backports InRelease [108 kB]
Hit:4 http://security.ubuntu.com/ubuntu focal-security InRelease
Get:5 https://packages.cloud.google.com/apt kubernetes-xenial InRelease [9383 B]
Get:6 https://packages.cloud.google.com/apt kubernetes-xenial/main amd64 Packages [58.4 kB]
Fetched 290 kB in 1s (541 kB/s)
Reading package lists... Done
ubuntu@ip-172-31-81-188:~$
```

Worker1



```
ubuntu@ip-172-31-23-53:~$ sudo apt-add-repository "deb http://apt.kubernetes.io/ kubernetes-xenial main"
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]
Hit:4 http://security.ubuntu.com/ubuntu focal-security InRelease
Get:5 https://packages.cloud.google.com/apt kubernetes-xenial InRelease [9383 B]
Get:6 https://packages.cloud.google.com/apt kubernetes-xenial/main amd64 Packages [58.4 kB]
Fetched 290 kB in 1s (556 kB/s)
Reading package lists... Done
ubuntu@ip-172-31-23-53:~$
```

Worker2



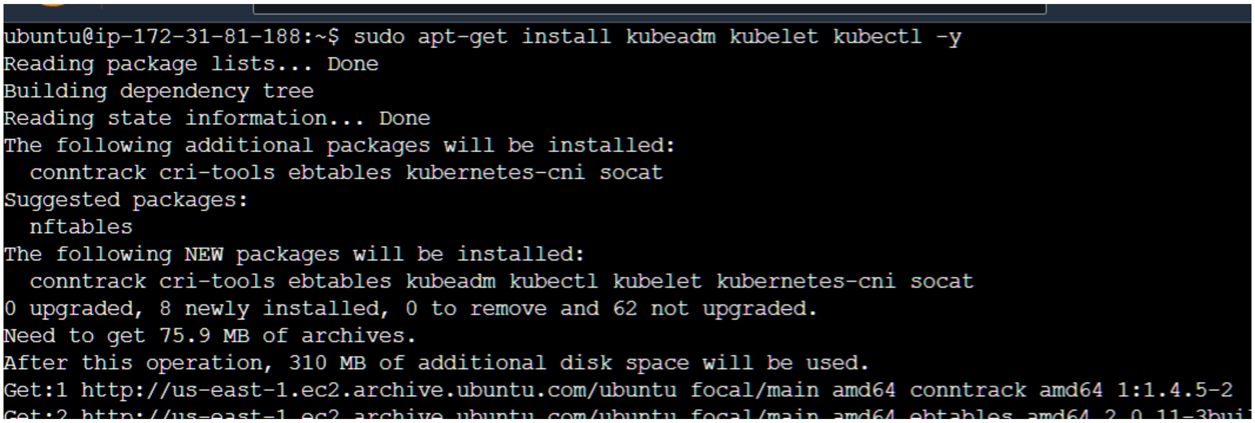
```
ubuntu@ip-172-31-21-143:~$ sudo apt-add-repository "deb http://apt.kubernetes.io/ kubernetes-xenial main"
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal-updates InRelease [114 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal-backports InRelease [108 kB]
Hit:5 http://security.ubuntu.com/ubuntu focal-security InRelease
Get:4 https://packages.cloud.google.com/apt kubernetes-xenial InRelease [9383 B]
Get:6 https://packages.cloud.google.com/apt kubernetes-xenial/main amd64 Packages [58.4 kB]
Fetched 290 kB in 1s (520 kB/s)
Reading package lists... Done
ubuntu@ip-172-31-21-143:~$
```

Step 5: Kubernetes Installation Tools

1. Install Kubernetes tools with the command:

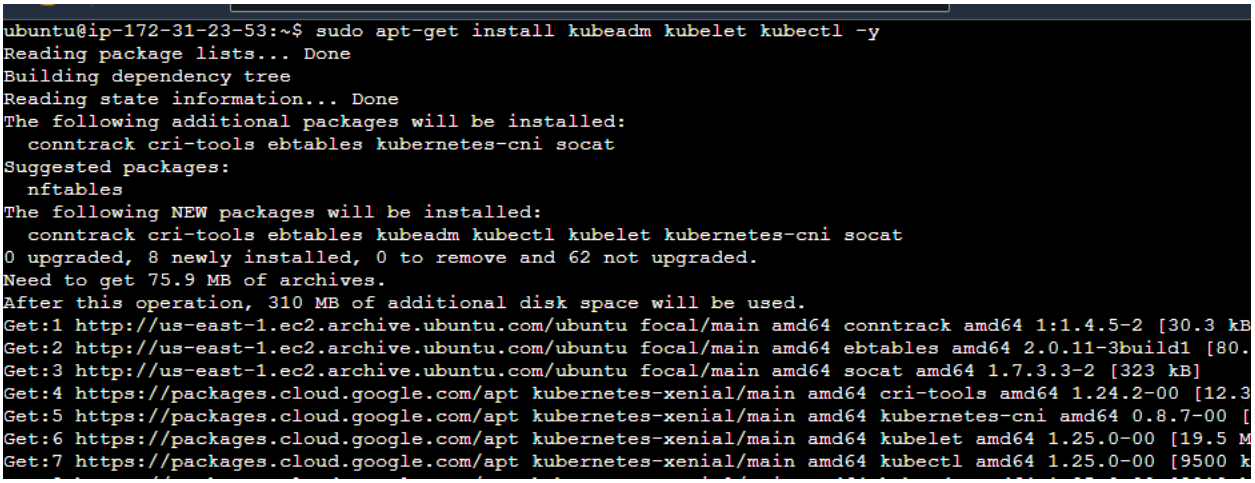
$sudo apt-get install kubeadm kubelet kubectl -y

Master



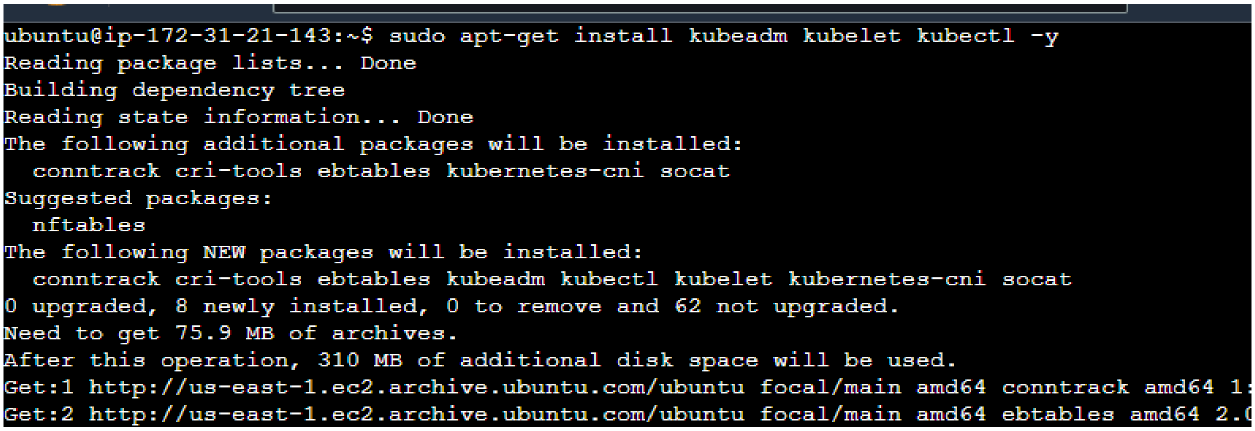
```
ubuntu@ip-172-31-81-188:~$ sudo apt-get install kubeadm kubelet kubectl -y
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  conntrack cri-tools ebtables kubernetes-cni socat
Suggested packages:
  nftables
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
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal/main amd64 ebtables amd64 2.0.11-3bui
```

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
```

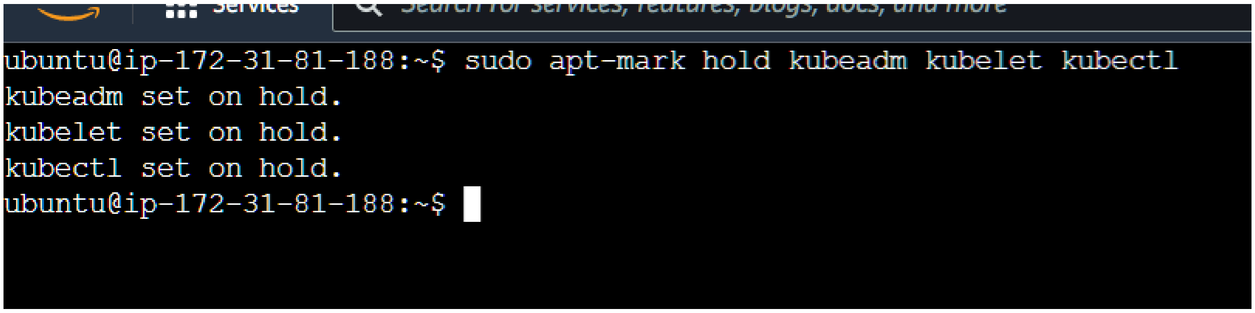
Worker2



```
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
```

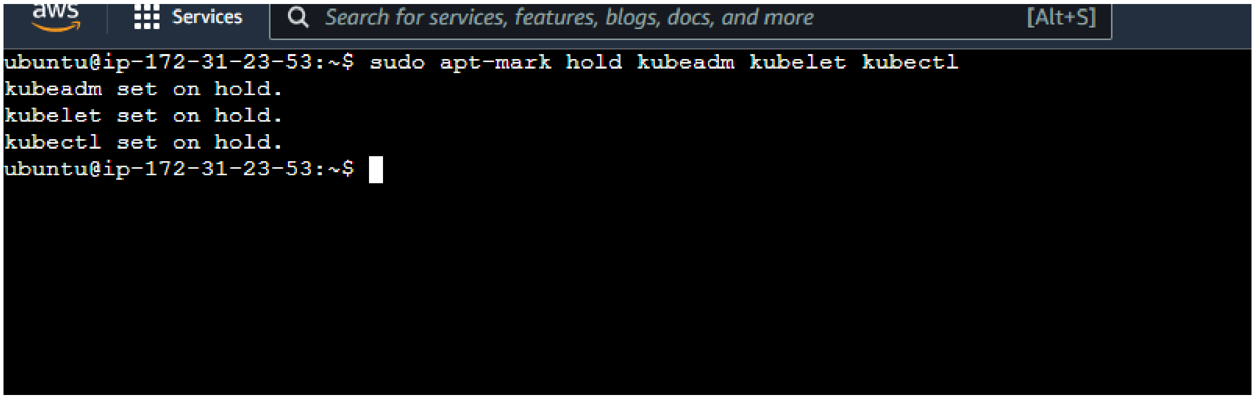
$sudo apt-mark hold kubeadm kubelet kubectl

Master



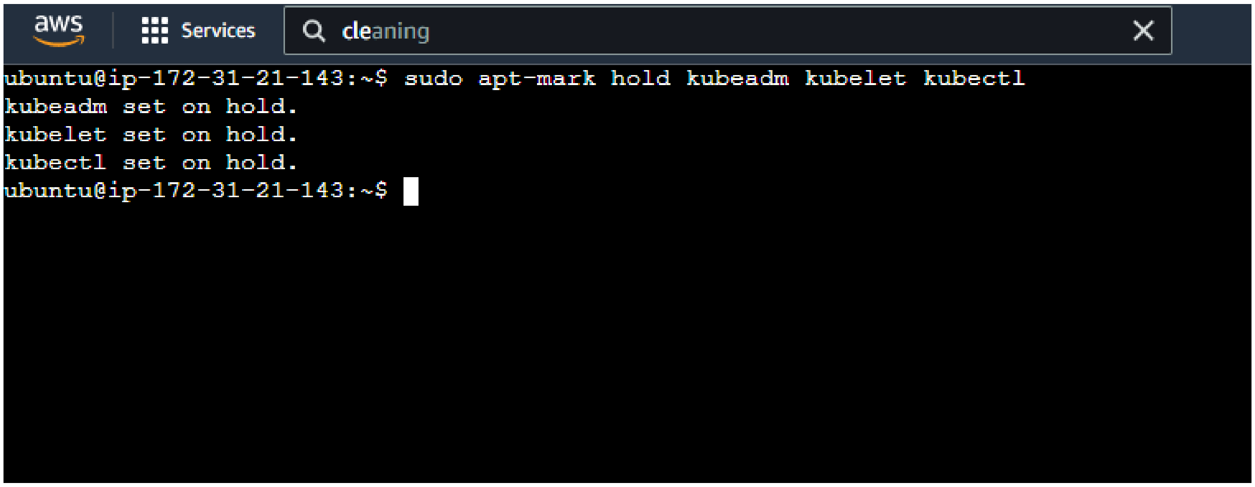
```
ubuntu@ip-172-31-81-188:~$ sudo apt-mark hold kubeadm kubelet kubectl
kubeadm set on hold.
kubelet set on hold.
kubectl set on hold.
ubuntu@ip-172-31-81-188:~$
```

Worker1



```
ubuntu@ip-172-31-23-53:~$ sudo apt-mark hold kubeadm kubelet kubectl
kubeadm set on hold.
kubelet set on hold.
kubectl set on hold.
ubuntu@ip-172-31-23-53:~$
```

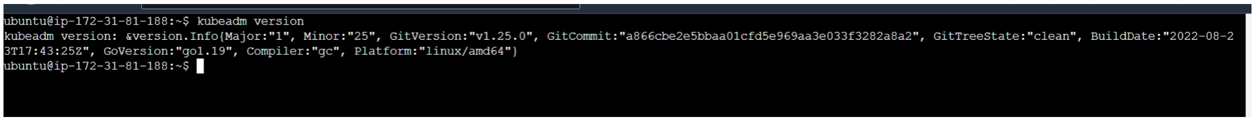
Worker2



```
ubuntu@ip-172-31-21-143:~$ sudo apt-mark hold kubeadm kubelet kubectl
kubeadm set on hold.
kubelet set on hold.
kubectl set on hold.
ubuntu@ip-172-31-21-143:~$
```

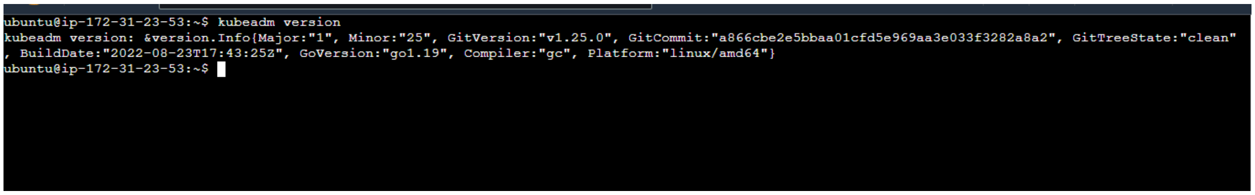
2. Verify the installation with:
$kubeadm version

Master



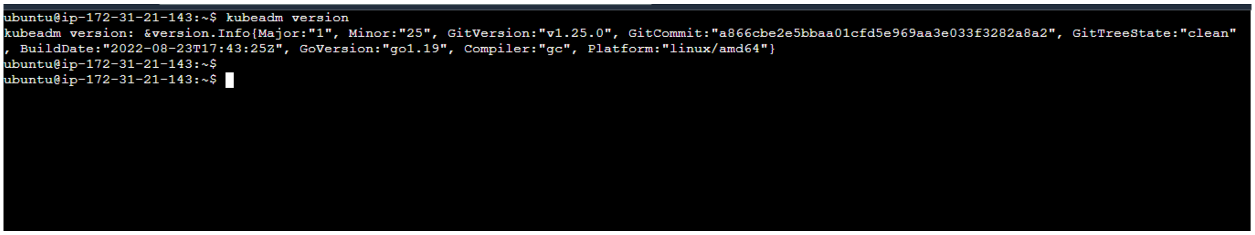
```
ubuntu@ip-172-31-81-188:~$ kubeadm version
kubeadm version: &version.Info{Major:"1", Minor:"25", GitVersion:"v1.25.0", GitCommit:"a866cbe2e5bbaa01cfd5e969aa3e033f3282a8a2", GitTreeState:"clean", BuildDate:"2022-08-2
3T17:43:25Z", GoVersion:"go1.19", Compiler:"gc", Platform:"linux/amd64"}
ubuntu@ip-172-31-81-188:~$
```

Worker1



```
ubuntu@ip-172-31-23-53:~$ kubeadm version
kubeadm version: &version.Info{Major:"1", Minor:"25", GitVersion:"v1.25.0", GitCommit:"a866cbe2e5bbaa01cfd5e969aa3e033f3282a8a2", GitTreeState:"clean"
, BuildDate:"2022-08-23T17:43:25Z", GoVersion:"go1.19", Compiler:"gc", Platform:"linux/amd64"}
ubuntu@ip-172-31-23-53:~$
```

Worker2



```
ubuntu@ip-172-31-21-143:~$ kubeadm version
kubeadm version: &version.Info{Major:"1", Minor:"25", GitVersion:"v1.25.0", GitCommit:"a866cbe2e5bbaa01cfd5e969aa3e033f3282a8a2", GitTreeState:"clean"
, BuildDate:"2022-08-23T17:43:25Z", GoVersion:"go1.19", Compiler:"gc", Platform:"linux/amd64"}
ubuntu@ip-172-31-21-143:~$
ubuntu@ip-172-31-21-143:~$
```

Kubernetes Deployment

Step 6: Begin Kubernetes Deployment

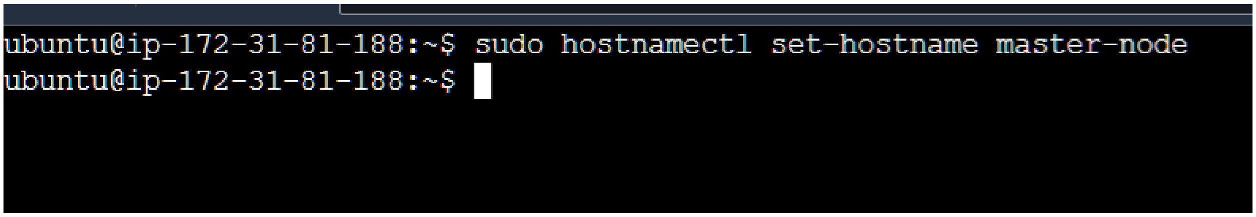
Start by disabling the swap memory on each machine:

$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:

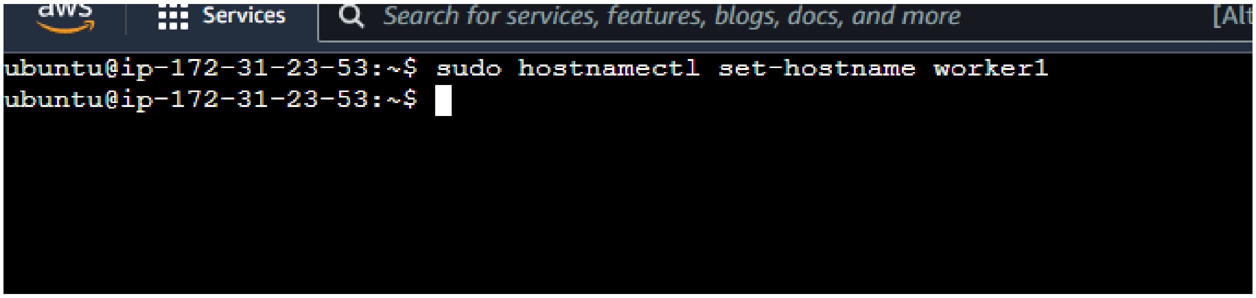
$sudo hostnamectl set-hostname master-node



```
ubuntu@ip-172-31-81-188:~$ sudo hostnamectl set-hostname master-node
ubuntu@ip-172-31-81-188:~$
```

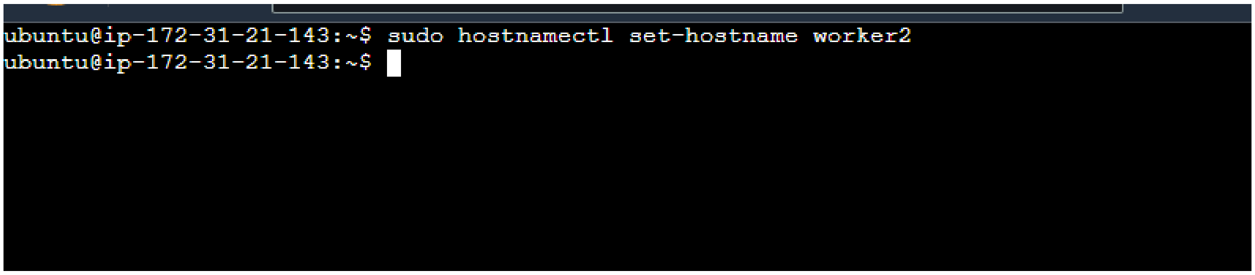
Next, set a worker node hostname by entering the following on the worker server:

$sudo hostnamectl set-hostname worker1



```
ubuntu@ip-172-31-23-53:~$ sudo hostnamectl set-hostname worker1
ubuntu@ip-172-31-23-53:~$ 
```

$sudo hostnamectl set-hostname worker2



```
ubuntu@ip-172-31-21-143:~$ sudo hostnamectl set-hostname worker2
ubuntu@ip-172-31-21-143:~$ 
```

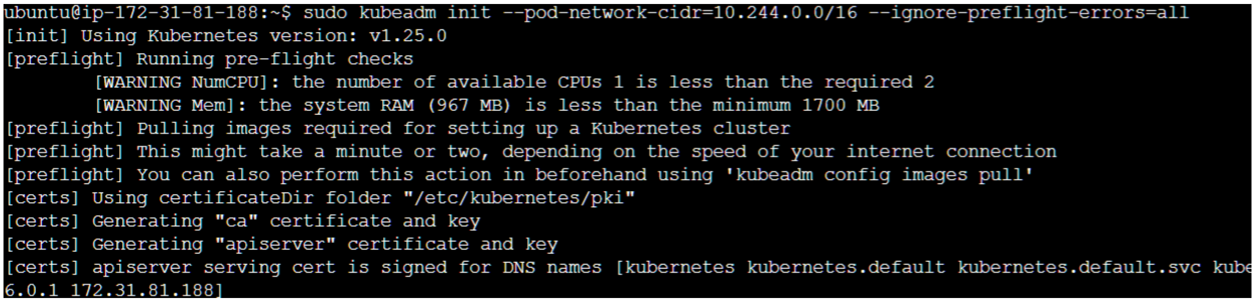
Note - Perform the next two steps i.e Step 8 and Step 9 only on the Master machine.

Step 8: Initialize Kubernetes only on Master Node

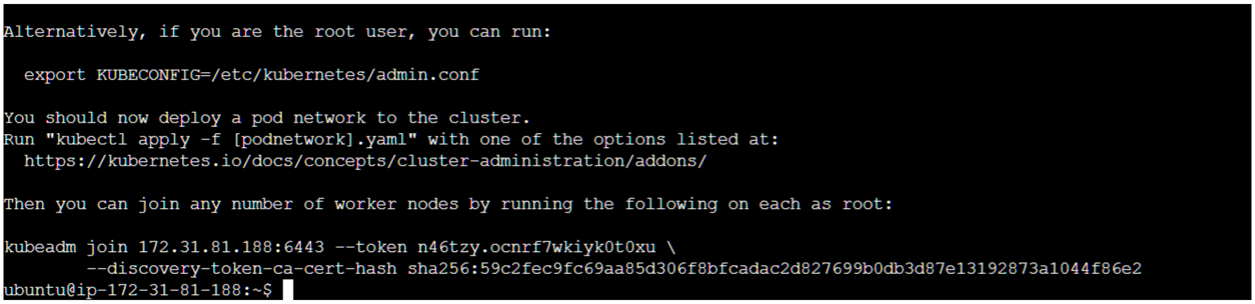
Switch to the master server node, and enter the following:

$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.



```
ubuntu@ip-172-31-81-188:~$ sudo kubeadm init --pod-network-cidr=10.244.0.0/16 --ignore-preflight-errors=all
[init] Using Kubernetes version: v1.25.0
[preflight] Running pre-flight checks
        [WARNING NumCPU]: the number of available CPUs 1 is less than the required 2
        [WARNING Mem]: the system RAM (967 MB) is less than the minimum 1700 MB
[preflight] Pulling images required for setting up a Kubernetes cluster
[preflight] This might take a minute or two, depending on the speed of your internet connection
[preflight] You can also perform this action in beforehand using 'kubeadm config images pull'
[certs] Using certificateDir folder "/etc/kubernetes/pki"
[certs] Generating "ca" certificate and key
[certs] Generating "apiserver" certificate and key
[certs] apiserver serving cert is signed for DNS names [kubernetes kubernetes.default kubernetes.default.svc kube
6.0.1 172.31.81.188]
```



```
Alternatively, if you are the root user, you can run:

  export KUBECONFIG=/etc/kubernetes/admin.conf

You should now deploy a pod network to the cluster.
Run "kubectl apply -f [podnetwork].yaml" with one of the options listed at:
  https://kubernetes.io/docs/concepts/cluster-administration/addons/

Then you can join any number of worker nodes by running the following on each as root:

kubeadm join 172.31.81.188:6443 --token n46tzy.ocnrf7wkiyk0t0xu \
        --discovery-token-ca-cert-hash sha256:59c2fec9fc69aa85d306f8bfcadac2d827699b0db3d87e13192873a1044f86e2
ubuntu@ip-172-31-81-188:~$
```

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

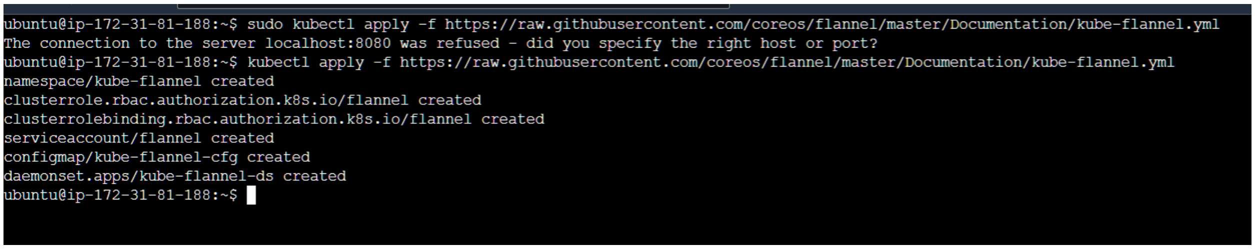
```
$ mkdir -p $HOME/.kube
$ sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
$ sudo chown $(id -u):$(id -g) $HOME/.kube/config
```

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
```

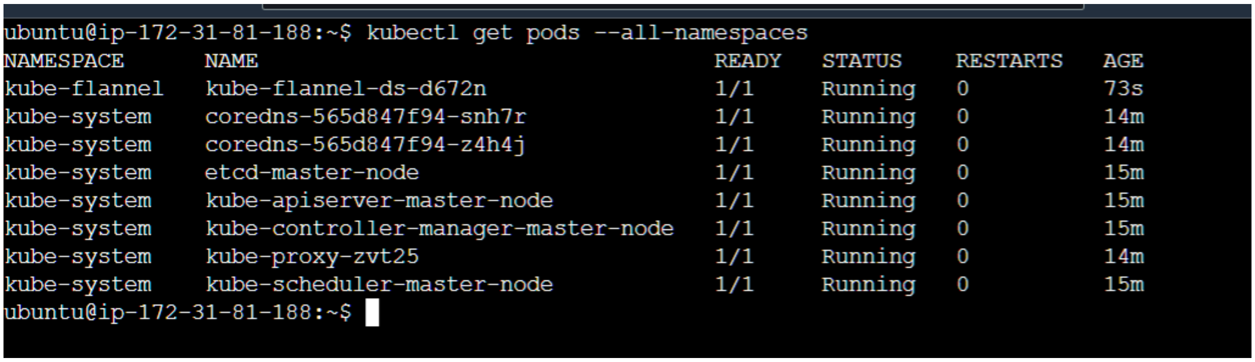


```
ubuntu@ip-172-31-81-188:~$ sudo kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml
The connection to the server localhost:8080 was refused - did you specify the right host or port?
ubuntu@ip-172-31-81-188:~$ kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml
namespace/kube-flannel created
clusterrole.rbac.authorization.k8s.io/flannel created
clusterrolebinding.rbac.authorization.k8s.io/flannel created
serviceaccount/flannel created
configmap/kube-flannel-cfg created
daemonset.apps/kube-flannel-ds created
ubuntu@ip-172-31-81-188:~$
```

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.

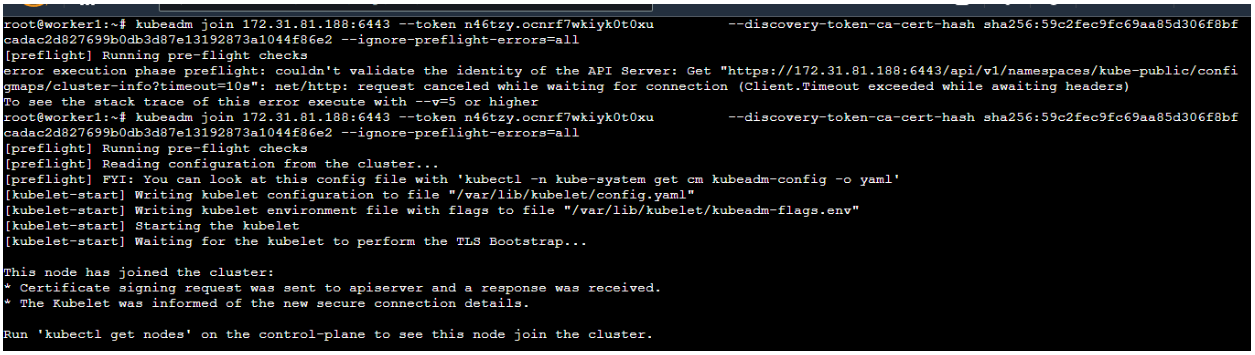
Step 10: Join Worker Node to Cluster

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

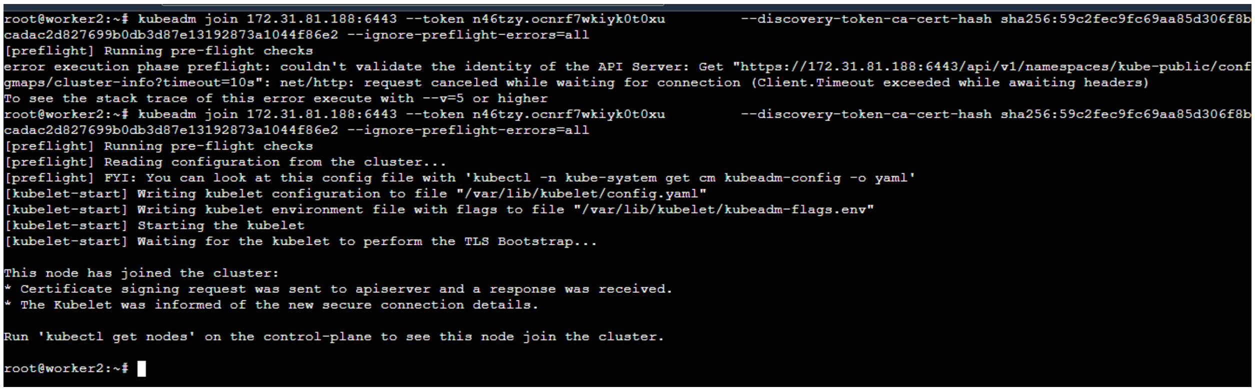
Note - Join command is different for everyone please do not use this.

Worker1



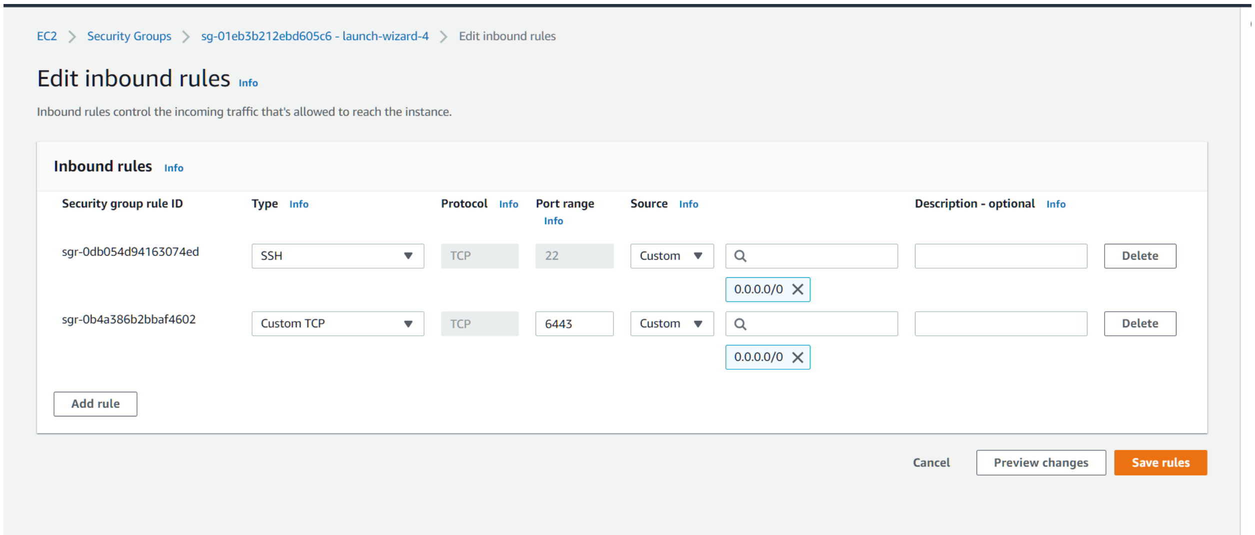
```
root@worker1:~# kubeadm join 172.31.81.188:6443 --token n46tzy.ocnrf7wkiyk0t0xu        --discovery-token-ca-cert-hash sha256:59c2fec9fc69aa85d306f8bf
cadac2d827699b0db3d87e13192873a1044f86e2 --ignore-preflight-errors=all
[preflight] Running pre-flight checks
error execution phase preflight: couldn't 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.
```

Worker2



```
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 edit the inbound rules of your security group and add port 6443.



EC2 > Security Groups > sg-01eb3b212ebd605c6 - launch-wizard-4 > Edit inbound rules

Edit inbound rules

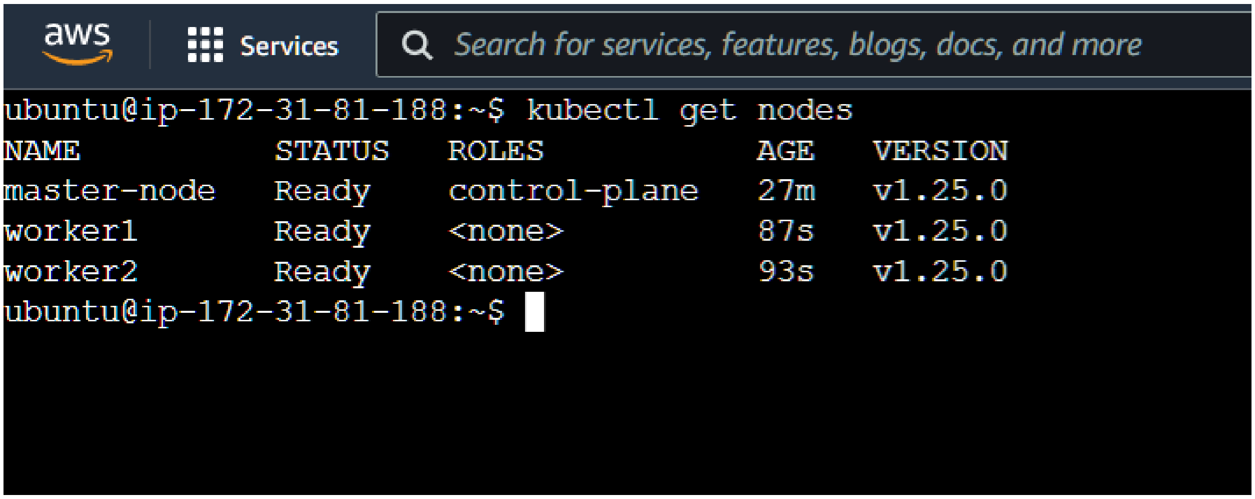
Inbound rules control the incoming traffic that's allowed to reach the instance.

Inbound rules

Security group rule ID	Type	Protocol	Port range	Source		Description - optional
sgr-0db054d94163074ed	SSH	TCP	22	Custom	0.0.0.0/0	
sgr-0b4a386b2bbaf4602	Custom TCP	TCP	6443	Custom	0.0.0.0/0	

Delete
Delete
Add rule
Cancel
Preview changes
Save rules

Switch to the master server, and enter:
$ kubectl get nodes



```
ubuntu@ip-172-31-81-188:~$ kubectl get nodes
NAME          STATUS   ROLES           AGE   VERSION
master-node   Ready    control-plane   27m   v1.25.0
worker1       Ready    <none>          87s   v1.25.0
worker2       Ready    <none>          93s   v1.25.0
ubuntu@ip-172-31-81-188:~$
```

Conclusion :

In this experiment, firstly we explored the Kubernetes tool. Also, learnt how to ~~connect~~ create a Kubernetes cluster on the AWS EC2 instance. But we have faced some issues while joining the worker nodes to the cluster but we resolved and successfully created a cluster with one master and two worker nodes.