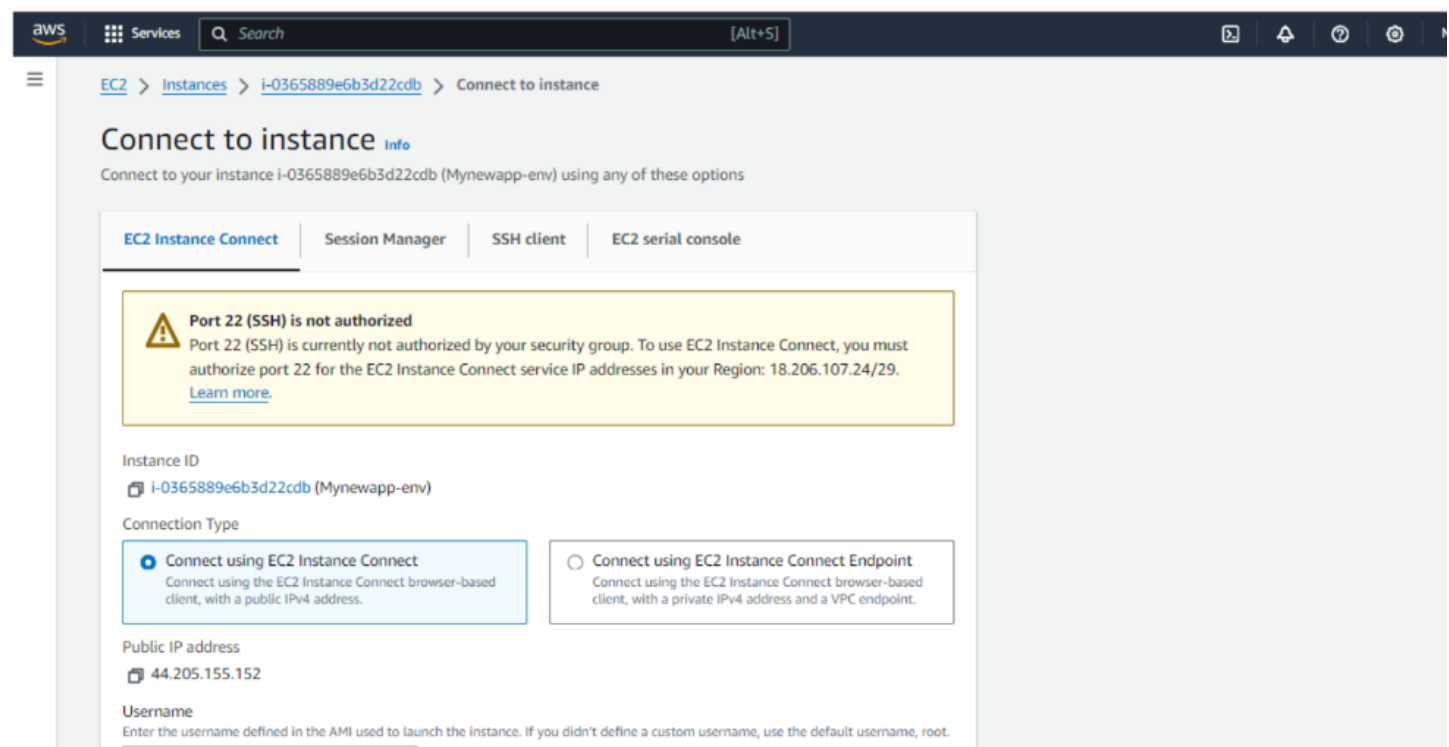
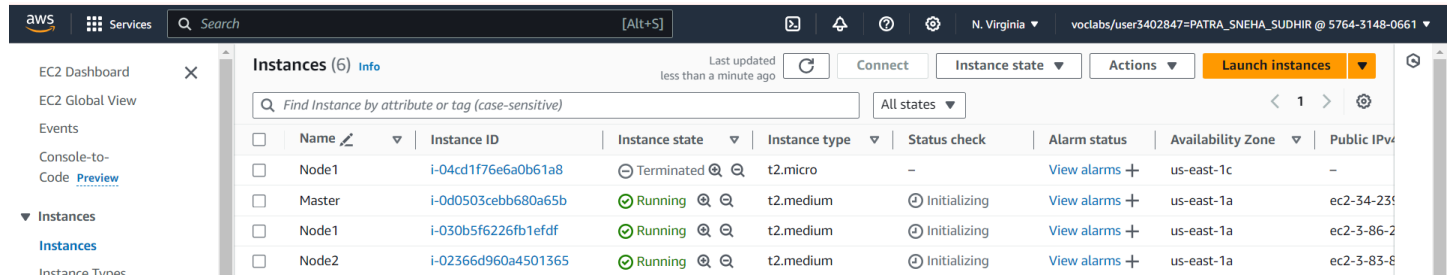


### Advanced DevOps Experiment 3

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

Step 1: Go to AWS Academy in services select EC2 and create 3 instance with and name them as master, node1, node2 and remember to select instance type as t2.medium.



Step 2: Then, Select and connect each instance and run the following commands inside the console of each instance.

- `sudo su`
- `yum install docker -y`
- `systemctl start docker`
- `docker --version`
- `yum repolist`

Services

Search

[Alt+S]

N. Virginia

voclabs/user3402847-PATRA\_SNEHA\_SUDHIR @ 5764-3148-0661

```

[ec2-user@ip-172-31-44-214 ~]$ sudo su
[root@ip-172-31-44-214 ec2-user]# yum install docker -y
Last metadata expiration check: 0:14:58 ago on Wed Sep 18 13:11:57 2024.
Dependencies resolved.

```

Package	Architecture	Version	Repository	Size
Installing:				
<b>docker</b>	x86_64	25.0.6-1.amzn2023.0.2	amazonlinux	44 M
Installing dependencies:				
<b>containerd</b>	x86_64	1.7.20-1.amzn2023.0.1	amazonlinux	35 M
<b>iptables-libs</b>	x86_64	1.8.8-3.amzn2023.0.2	amazonlinux	401 k
<b>iptables-nft</b>	x86_64	1.8.8-3.amzn2023.0.2	amazonlinux	183 k
<b>libcgroup</b>	x86_64	3.0-1.amzn2023.0.1	amazonlinux	75 k
<b>libnetfilter_conntrack</b>	x86_64	1.0.8-2.amzn2023.0.2	amazonlinux	58 k
<b>libnftnl</b>	x86_64	1.0.1-19.amzn2023.0.2	amazonlinux	30 k
<b>libnftnl</b>	x86_64	1.2.2-2.amzn2023.0.2	amazonlinux	84 k
<b>pkgz</b>	x86_64	2.5-1.amzn2023.0.3	amazonlinux	83 k
<b>runc</b>	x86_64	1.1.13-1.amzn2023.0.1	amazonlinux	3.2 M

```

Transaction Summary
-----
Install 10 Packages

Total download size: 84 M
Installed size: 317 M
Downloading Packages:
(1/10): iptables-libs-1.8.8-3.amzn2023.0.2.x86_64.rpm 3.0 MB/s | 401 kB 00:00

```

i-030d25988270b63ec (Master)

PublicIPs: 54.152.128.140 PrivateIPs: 172.31.44.214

Services

Search

[Alt+S]

N. Virginia

voclabs/user3402847-PATRA\_SNEHA\_SUDHIR @ 5764-3148-0661

```

Installing      : libcgroup-3.0-1.amzn2023.0.1.x86_64 9/10
Running scriptlet: docker-25.0.6-1.amzn2023.0.2.x86_64 10/10
Installing      : docker-25.0.6-1.amzn2023.0.2.x86_64 10/10
Running scriptlet: docker-25.0.6-1.amzn2023.0.2.x86_64 10/10
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket → /usr/lib/systemd/system/docker.socket.

Verifying      : containerd-1.7.20-1.amzn2023.0.1.x86_64 1/10
Verifying      : docker-25.0.6-1.amzn2023.0.2.x86_64 2/10
Verifying      : iptables-libs-1.8.8-3.amzn2023.0.2.x86_64 3/10
Verifying      : iptables-nft-1.8.8-3.amzn2023.0.2.x86_64 4/10
Verifying      : libcgroup-3.0-1.amzn2023.0.1.x86_64 5/10
Verifying      : libnetfilter_conntrack-1.0.8-2.amzn2023.0.2.x86_64 6/10
Verifying      : libnftnl-1.0.1-19.amzn2023.0.2.x86_64 7/10
Verifying      : libnftnl-1.2.2-2.amzn2023.0.2.x86_64 8/10
Verifying      : pkgz-2.5-1.amzn2023.0.3.x86_64 9/10
Verifying      : runc-1.1.13-1.amzn2023.0.1.x86_64 10/10

Installed:
containerd-1.7.20-1.amzn2023.0.1.x86_64 docker-25.0.6-1.amzn2023.0.2.x86_64 iptables-libs-1.8.8-3.amzn2023.0.2.x86_64
iptables-nft-1.8.8-3.amzn2023.0.2.x86_64 libcgroup-3.0-1.amzn2023.0.1.x86_64 libnetfilter_conntrack-1.0.8-2.amzn2023.0.2.x86_64
libnftnl-1.0.1-19.amzn2023.0.2.x86_64 libnftnl-1.2.2-2.amzn2023.0.2.x86_64 pkgz-2.5-1.amzn2023.0.3.x86_64
runc-1.1.13-1.amzn2023.0.1.x86_64

Complete!
[root@ip-172-31-44-214 ec2-user]# systemctl start docker
[root@ip-172-31-44-214 ec2-user]# docker --version
Docker version 25.0.5, build 5dc9bcc
[root@ip-172-31-44-214 ec2-user]#

```

i-030d25988270b63ec (Master)

PublicIPs: 54.152.128.140 PrivateIPs: 172.31.44.214

Step 3: Now, visit the following link <https://kubernetes.io/docs/setup/production-environment/tools/kubeadm/install-kubeadm/> and scroll down till you find Red-Hat and then select Red-Hat based distributions tab copy all the commands one by one in each console of instance.

- Documentation
- Getting started
  - Learning environment
  - Production environment
    - Container Runtimes
    - Installing Kubernetes with deployment tools
      - Bootstrapping clusters with kubeadm
        - Installing kubeadm
        - Troubleshooting kubeadm
        - Creating a cluster with kubeadm
        - Customizing components with the kubeadm API
        - Options for Highly Available Topology

[Debian-based distributions](#)
[Red Hat-based distributions](#)
[Without a package manager](#)

1. Set SELinux to `permissive` mode:

These instructions are for Kubernetes 1.31.

```
# Set SELinux in permissive mode (effectively disabling it)
sudo setenforce 0
sudo sed -i 's/^SELINUX=enforcing$/SELINUX=permissive/' /etc/selinux/config
```

#### Caution:

- Setting SELinux in permissive mode by running `setenforce 0` and `sed ...` effectively disables it. This is required to allow containers to access the host filesystem; for example, some cluster network plugins require that. You have to do this until SELinux support is improved in the kubelet.
- You can leave SELinux enabled if you know how to configure it but it may require settings that are not supported by kubeadm.

2. Add the Kubernetes `yum` repository. The `exclude` parameter in the repository definition ensures that the packages related to Kubernetes are not upgraded upon running `yum update` as there's a special procedure that must be followed for upgrading Kubernetes. Please note that this repository have packages only for Kubernetes 1.31; for other Kubernetes minor versions, you need to change the Kubernetes minor version in the URL to match your desired minor version (you should also check that you are reading the documentation for the version of Kubernetes that you plan to install).

[Edit this page](#)
[Create child page](#)
[Create documentation issue](#)
[Print entire section](#)

Before you begin

Verify the MAC address and product\_uuid are

unique for every node

Check network adapters

Check required ports

Swap configuration

Installing a container runtime

Installing kubeadm, kubelet and kubectl

Configuring a cgroup driver

Troubleshooting

What's next

[Alt+S]

N. Virginia
voclabs/user3402847=PATRA\_SNEHA\_SUDHIR @ 5764-3148-0661

```
Complete!
[root@ip-172-31-44-214 ec2-user]# systemctl start docker
[root@ip-172-31-44-214 ec2-user]# docker --version
Docker version 25.0.5, build 5dc9bcc
[root@ip-172-31-44-214 ec2-user]# yum repolist
repo id                                repo name
amazonlinux                            Amazon Linux 2023 repository
kernel-livepatch                       Amazon Linux 2023 Kernel Livepatch repository
[root@ip-172-31-44-214 ec2-user]# sudo setenforce 0
sudo sed -i 's/^SELINUX=enforcing$/SELINUX=permissive/' /etc/selinux/config
[root@ip-172-31-44-214 ec2-user]# cat <<EOF | sudo tee /etc/yum.repos.d/kubernetes.repo
[kubernetes]
name=Kubernetes
baseurl=https://pkgs.k8s.io/core:/stable:/v1.31/rpm/
enabled=1
gpgcheck=1
gpgkey=https://pkgs.k8s.io/core:/stable:/v1.31/rpm/repodata/repomd.xml.key
exclude=kubelet kubeadm kubectl cri-tools kubernetes-cni
EOF
[kubernetes]
name=Kubernetes
baseurl=https://pkgs.k8s.io/core:/stable:/v1.31/rpm/
enabled=1
gpgcheck=1
gpgkey=https://pkgs.k8s.io/core:/stable:/v1.31/rpm/repodata/repomd.xml.key
exclude=kubelet kubeadm kubectl cri-tools kubernetes-cni
[root@ip-172-31-44-214 ec2-user]# sudo yum install -y kubelet kubeadm kubectl --disableexcludes=kubernetes
```

i-030d25988270b63ec (Master)
PublicIPs: 54.152.128.140 PrivateIPs: 172.31.44.214

```
aws Services Search [Alt+S] N. Virginia voclabs/user3402847=PATRA_SNEHA_SUDHIR @ 5764-3148-0661
[root@ip-172-31-44-214 ec2-user]# sudo yum install -y kubelet kubeadm kubectl --disableexcludes=kubernetes
Kubernetes
Dependencies resolved.
43 kB/s | 9.4 kB 00:00
=====
Package Architecture Version Repository Size
-----
Installing:
kubeadm x86_64 1.31.1-150500.1.1 kubernetes 11 M
kubectl x86_64 1.31.1-150500.1.1 kubernetes 11 M
kubelet x86_64 1.31.1-150500.1.1 kubernetes 15 M
Installing dependencies:
conntrack-tools x86_64 1.4.6-2.amzn2023.0.2 amazonlinux 208 k
cri-tools x86_64 1.31.1-150500.1.1 kubernetes 6.9 M
kubernetes-cni x86_64 1.5.1-150500.1.1 kubernetes 7.1 M
libnetfilter_cthelper x86_64 1.0.0-21.amzn2023.0.2 amazonlinux 24 k
libnetfilter_cttimeout x86_64 1.0.0-19.amzn2023.0.2 amazonlinux 24 k
libnetfilter_queue x86_64 1.0.5-2.amzn2023.0.2 amazonlinux 30 k
Transaction Summary
-----
Install 9 Packages
Total download size: 51 M
Installed size: 269 M
Downloading Packages:
(1/9): libnetfilter_cthelper-1.0.0-21.amzn2023.0.2.x86_64.rpm 378 kB/s | 24 kB 00:00
(2/9): libnetfilter_cttimeout-1.0.0-19.amzn2023.0.2.x86_64.rpm 360 kB/s | 24 kB 00:00
(3/9): conntrack-tools-1.4.6-2.amzn2023.0.2.x86_64.rpm 2.7 MB/s | 208 kB 00:00
-----

i-030d25988270b63ec (Master)
PublicIPs: 54.152.128.140 PrivateIPs: 172.31.44.214

aws Services Search [Alt+S] N. Virginia voclabs/user3402847=PATRA_SNEHA_SUDHIR @ 5764-3148-0661
Installing : libnetfilter_cttimeout-1.0.0-19.amzn2023.0.2.x86_64 4/9
Installing : libnetfilter_cthelper-1.0.0-21.amzn2023.0.2.x86_64 5/9
Installing : conntrack-tools-1.4.6-2.amzn2023.0.2.x86_64 6/9
Running scriptlet: conntrack-tools-1.4.6-2.amzn2023.0.2.x86_64 6/9
Installing : kubelet-1.31.1-150500.1.1.x86_64 7/9
Running scriptlet: kubelet-1.31.1-150500.1.1.x86_64 7/9
Installing : kubeadm-1.31.1-150500.1.1.x86_64 8/9
Installing : kubectl-1.31.1-150500.1.1.x86_64 9/9
Running scriptlet: kubectl-1.31.1-150500.1.1.x86_64 9/9
Verifying : conntrack-tools-1.4.6-2.amzn2023.0.2.x86_64 1/9
Verifying : libnetfilter_cthelper-1.0.0-21.amzn2023.0.2.x86_64 2/9
Verifying : libnetfilter_cttimeout-1.0.0-19.amzn2023.0.2.x86_64 3/9
Verifying : libnetfilter_queue-1.0.5-2.amzn2023.0.2.x86_64 4/9
Verifying : cri-tools-1.31.1-150500.1.1.x86_64 5/9
Verifying : kubeadm-1.31.1-150500.1.1.x86_64 6/9
Verifying : kubectl-1.31.1-150500.1.1.x86_64 7/9
Verifying : kubelet-1.31.1-150500.1.1.x86_64 8/9
Verifying : kubernetes-cni-1.5.1-150500.1.1.x86_64 9/9
Installed:
conntrack-tools-1.4.6-2.amzn2023.0.2.x86_64 cri-tools-1.31.1-150500.1.1.x86_64 kubeadm-1.31.1-150500.1.1.x86_64
kubectl-1.31.1-150500.1.1.x86_64 kubelet-1.31.1-150500.1.1.x86_64 kubernetes-cni-1.5.1-150500.1.1.x86_64
libnetfilter_cthelper-1.0.0-21.amzn2023.0.2.x86_64 libnetfilter_cttimeout-1.0.0-19.amzn2023.0.2.x86_64 libnetfilter_queue-1.0.5-2.amzn2023.0.2.x86_64
Complete!
[root@ip-172-31-44-214 ec2-user]# sudo systemctl enable --now kubelet
Created symlink /etc/systemd/system/multi-user.target.wants/kubelet.service -> /usr/lib/systemd/system/kubelet.service.
[root@ip-172-31-44-214 ec2-user]#

i-030d25988270b63ec (Master)
PublicIPs: 54.152.128.140 PrivateIPs: 172.31.44.214
```

Step 4: Now, run the following command in the mater instance -  
kubeadm init

```
aws Services Search [Alt+S] N. Virginia voclabs/user3402847=PATRA_SNEHA_SUDHIR @ 5764-3148-0661
[root@ip-172-31-81-4 ec2-user]# kubeadm init
[init] Using Kubernetes version: v1.31.0
[preflight] Running pre-flight checks
[WARNING FileExisting-socat]: socat not found in system path
[WARNING FileExisting-tc]: tc not found in system path
[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 beforehand using 'kubeadm config images pull'
W0918 14:26:24.654814 28225 checks.go:846] detected that the sandbox image "registry.k8s.io/pause:3.8" of the container runtime is inconsistent with that used by kubeadm. It is recommended to use "registry.k8s.io/pause:3.10" as the CRI sandbox image.
[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 [ip-172-31-81-4.ec2.internal kubernetes.default kubernetes.default.svc kubernetes.default.svc.cluster.local] and IPs [10.96.0.1 172.31.81.4]
[certs] Generating "apiserver-kubelet-client" certificate and key
[certs] Generating "front-proxy-ca" certificate and key
[certs] Generating "front-proxy-client" certificate and key
[certs] Generating "etcd/ca" certificate and key
[certs] Generating "etcd/server" certificate and key
[certs] etcd/server serving cert is signed for DNS names [ip-172-31-81-4.ec2.internal localhost] and IPs [172.31.81.4 127.0.0.1 ::1]
[certs] Generating "etcd/peer" certificate and key
[certs] etcd/peer serving cert is signed for DNS names [ip-172-31-81-4.ec2.internal localhost] and IPs [172.31.81.4 127.0.0.1 ::1]
[certs] Generating "etcd/healthcheck-client" certificate and key
[certs] Generating "apiserver-etcd-client" certificate and key
[certs] Generating "sa" key and public key
[kubeconfig] Using kubeconfig folder "/etc/kubernetes"
[kubeconfig] Writing "admin.conf" kubeconfig file

i-0d0503cebb680a65b (Master)
PublicIPs: 34.239.132.160 PrivateIPs: 172.31.81.4
```

Step 5: Now, run the following commands in master instance's console –

- a. `mkdir -p $HOME/.kube`  
`sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config`  
`sudo chown $(id -u):$(id -g) $HOME/.kube/config`
- b. `export KUBECONFIG=/etc/kubernetes/admin.conf`
- c. `kubeadm join 172.31.93.102:6443 --token 6ccgvw.o10vq5f2n5d9fa42 \`  
`--discovery-token-ca-cert-hash`  
`sha256:1bbcc9939e895e8de0e0ddd7ec72d881a9ef3b8f51a42f3145857e54b13c3818`

```
aws [Alt+S] N. Virginia voclabs/user3402847=PATRA_SNEHA_SUDHIR @ 5764-3148-0661
[bootstrap-token] Configuring bootstrap tokens, cluster-info ConfigMap, RBAC Roles
[bootstrap-token] Configured RBAC rules to allow Node Bootstrap tokens to get nodes
[bootstrap-token] Configured RBAC rules to allow Node Bootstrap tokens to post CSRs in order for nodes to get long term certificate credentials
[bootstrap-token] Configured RBAC rules to allow the csrapprover controller automatically approve CSRs from a Node Bootstrap Token
[bootstrap-token] Configured RBAC rules to allow certificate rotation for all node client certificates in the cluster
[bootstrap-token] Creating the "cluster-info" ConfigMap in the "kube-public" namespace
[kubelet-finalize] Updating "/etc/kubernetes/kubelet.conf" to point to a rotatable kubelet client certificate and key
[addons] Applied essential addon: CoreDNS
[addons] Applied essential addon: kube-proxy

Your Kubernetes control-plane has initialized successfully!

To start using your cluster, you need to run the following as a regular user:

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

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:
```

Step 6: Run this command in node1 and node2 -

`kubeadm join 172.31.93.102:6443 --token 6ccgvw.o10vq5f2n5d9fa42 \`  
`--discovery-token-ca-cert-hash sha256:1bbcc9939e895e8de0e0ddd7ec72d881a9ef3b8f51a42f3145857e54b13c3818`

```
Installing : conntrack-tools-1.4.6-2.amzn2023.0.2.x86_64 6/9
Running scriptlet: conntrack-tools-1.4.6-2.amzn2023.0.2.x86_64 6/9
Installing : kubelet-1.31.1-150500.1.1.x86_64 7/9
Running scriptlet: kubelet-1.31.1-150500.1.1.x86_64 7/9
Installing : kubeadm-1.31.1-150500.1.1.x86_64 8/9
Installing : kubectcl-1.31.1-150500.1.1.x86_64 9/9
Running scriptlet: kubectcl-1.31.1-150500.1.1.x86_64 9/9
Verifying : conntrack-tools-1.4.6-2.amzn2023.0.2.x86_64 1/9
Verifying : libnetfilter_cthelper-1.0.0-21.amzn2023.0.2.x86_64 2/9
Verifying : libnetfilter_cttimeout-1.0.0-19.amzn2023.0.2.x86_64 3/9
Verifying : libnetfilter_queue-1.0.5-2.amzn2023.0.2.x86_64 4/9
Verifying : cri-tools-1.31.1-150500.1.1.x86_64 5/9
Verifying : kubeadm-1.31.1-150500.1.1.x86_64 6/9
Verifying : kubectcl-1.31.1-150500.1.1.x86_64 7/9
Verifying : kubelet-1.31.1-150500.1.1.x86_64 8/9
Verifying : kubernetes-cni-1.5.1-150500.1.1.x86_64 9/9

Installed:
  conntrack-tools-1.4.6-2.amzn2023.0.2.x86_64      cri-tools-1.31.1-150500.1.1.x86_64      kubeadm-1.31.1-150500.1.1.x86_64
  kubectcl-1.31.1-150500.1.1.x86_64              kubelet-1.31.1-150500.1.1.x86_64      kubernetes-cni-1.5.1-150500.1.1.x86_64
  libnetfilter_cthelper-1.0.0-21.amzn2023.0.2.x86_64  libnetfilter_cttimeout-1.0.0-19.amzn2023.0.2.x86_64  libnetfilter_queue-1.0.5-2.amzn2023.0.2.x86_64

Complete!
[root@ip-172-31-95-221 ec2-user]# sudo systemctl enable --now kubelet
Created symlink /etc/systemd/system/multi-user.target.wants/kubelet.service -> /usr/lib/systemd/system/kubelet.service.
[root@ip-172-31-95-221 ec2-user]# kubeadm join 172.31.93.102:6443 --token 6ccgvw.o10vq5f2n5d9fa42 \
--discovery-token-ca-cert-hash sha256:1bbcc9939e895e8de0e0ddd7ec72d881a9ef3b8f51a42f3145857e54b13c3818
[preflight] Running pre-flight checks
[WARNING FileExisting-socat]: socat not found in system path
[WARNING FileExisting-tc]: tc not found in system path
error execution phase preflight: couldn't validate the identity of the API Server: failed to request the cluster-info ConfigMap: Get "https://172.31.93.102:6443/api/v1/namespaces/kube-public/configmaps/cluster-info?timeout=10s": context deadline exceeded
To see the stack trace of this error execute with --v=5 or higher
[root@ip-172-31-95-221 ec2-user]#
```

```
Installing      : conntrack-tools-1.4.6-2.amzn2023.0.2.x86_64 6/9
Running scriptlet: conntrack-tools-1.4.6-2.amzn2023.0.2.x86_64 6/9
Installing      : kubelet-1.31.1-150500.1.1.x86_64 7/9
Running scriptlet: kubelet-1.31.1-150500.1.1.x86_64 7/9
Installing      : kubeadm-1.31.1-150500.1.1.x86_64 8/9
Installing      : kubect1-1.31.1-150500.1.1.x86_64 9/9
Running scriptlet: kubect1-1.31.1-150500.1.1.x86_64 9/9
Verifying       : conntrack-tools-1.4.6-2.amzn2023.0.2.x86_64 1/9
Verifying       : libnetfilter_cthelper-1.0.0-21.amzn2023.0.2.x86_64 2/9
Verifying       : libnetfilter_cttimeout-1.0.0-19.amzn2023.0.2.x86_64 3/9
Verifying       : libnetfilter_queue-1.0.5-2.amzn2023.0.2.x86_64 4/9
Verifying       : cri-tools-1.31.1-150500.1.1.x86_64 5/9
Verifying       : kubeadm-1.31.1-150500.1.1.x86_64 6/9
Verifying       : kubect1-1.31.1-150500.1.1.x86_64 7/9
Verifying       : kubelet-1.31.1-150500.1.1.x86_64 8/9
Verifying       : kubernetes-cni-1.5.1-150500.1.1.x86_64 9/9

Installed:
  conntrack-tools-1.4.6-2.amzn2023.0.2.x86_64      cri-tools-1.31.1-150500.1.1.x86_64      kubeadm-1.31.1-150500.1.1.x86_64
  kubect1-1.31.1-150500.1.1.x86_64                kubelet-1.31.1-150500.1.1.x86_64      kubernetes-cni-1.5.1-150500.1.1.x86_64
  libnetfilter_cthelper-1.0.0-21.amzn2023.0.2.x86_64  libnetfilter_cttimeout-1.0.0-19.amzn2023.0.2.x86_64  libnetfilter_queue-1.0.5-2.amzn2023.0.2.x86_64

Complete!
[root@ip-172-31-94-95 ec2-user]# sudo systemctl enable --now kubelet
Created symlink /etc/systemd/system/multi-user.target.wants/kubelet.service → /usr/lib/systemd/system/kubelet.service.
[root@ip-172-31-94-95 ec2-user]# kubeadm join 172.31.93.102:6443 --token 6ccqvw.o10vq5f2n5d9fa42 \
--discovery-token-ca-cert-hash sha256:1bbcc9939e095e8de0e0dd7ec72d881a9ef3b8f51a42f3145857e54b13c3818
preflight] Running pre-flight checks
[WARNING FileExisting-socat]: socat not found in system path
[WARNING FileExisting-tc]: tc not found in system path
error execution phase preflight: couldn't validate the identity of the API Server: failed to request the cluster-info ConfigMap: Get "https://172.31.93.102:6443/api/v1/namespaces/kube-public/configmaps/cluster-info?timeout=10s": context deadline exceeded
to see the stack trace of this error execute with --v=5 or higher
[root@ip-172-31-94-95 ec2-user]#
```

Step 7: Run the following command in master instance console -  
kubect1 get nodes

```
aws Services Search [Alt+S] N. Virginia voclabs/user3402647=PATRA_SNEHA_SUDHIR @ 5764-3148-0661

[root@ip-172-31-01-4 ec2-user]# kubect1 get nodes
NAME                                STATUS    ROLES    AGE   VERSION
ip-172-31-01-4.ec2.internal        NotReady control-plane 26m   v1.31.1
[root@ip-172-31-01-4 ec2-user]# kubect1 get nodes
NAME                                STATUS    ROLES    AGE   VERSION
ip-172-31-01-4.ec2.internal        NotReady control-plane 26m   v1.31.1
ip-172-31-94-95.ec2.internal        NotReady <none>      17s   v1.31.1
ip-172-31-95-221.ec2.internal        NotReady <none>      13s   v1.31.1
[root@ip-172-31-01-4 ec2-user]#
```