Advanced DevOps Lab

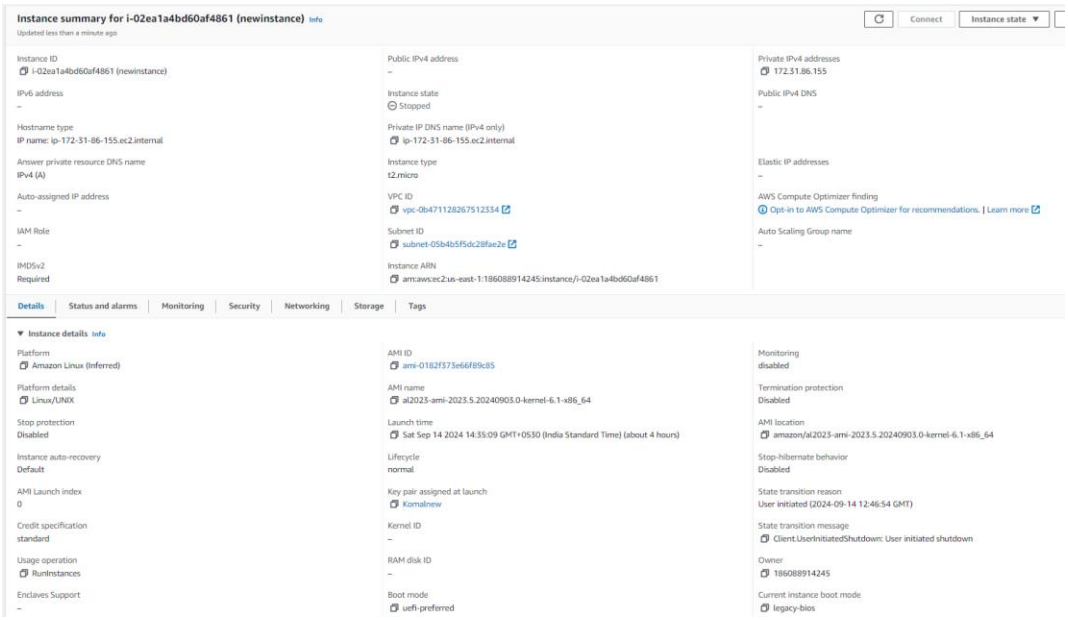
Experiment:3

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

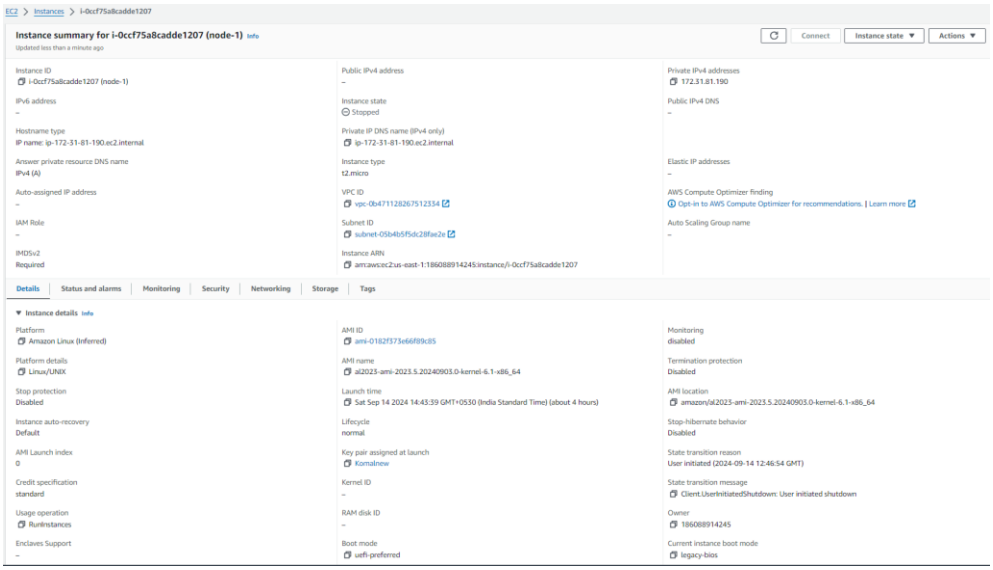
Execution:

Creating 3 instances Master and two Workers ,here ‘newinstance’ is the master whereas node-1 and node-2 are worker nodes.

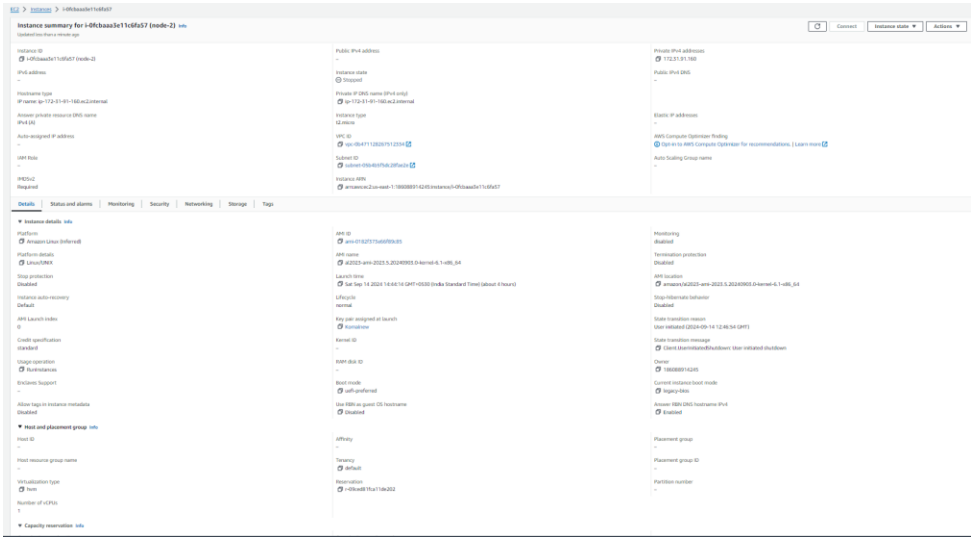
Creating newinstance:



Creating node-1:



Creating node-2:



we created and configured all three instances:

newinstance	i-02ea1a4bd60af4861	Stopped	t2.micro	-	View alarms +	us-east-1c	-	-	-	-	disabled	launch-wizard-10	Komalnew	2024/09/14 14:35 GMT+5:30	Linux/UNIX
node-2	i-0fcbaaa3e11c6fa57	Stopped	t2.micro	-	View alarms +	us-east-1c	-	-	-	-	disabled	launch-wizard-13	Komalnew	2024/09/14 14:44 GMT+5:30	Linux/UNIX
node-1	i-0ccf75a8cadde1207	Stopped	t2.micro	-	View alarms +	us-east-1c	-	-	-	-	disabled	launch-wizard-12	Komalnew	2024/09/14 14:43 GMT+5:30	Linux/UNIX

Following commands are used for making an SSH connection in all 3 machines:

```
2022k@Komal22 MINGW64 ~ (main)
$ cd Downloads

2022k@Komal22 MINGW64 ~/Downloads (main)
$ chmod 400 "Komalnew.pem"

2022k@Komal22 MINGW64 ~/Downloads (main)
$ ssh -i "Komalnew.pem" ec2-user@ec2-54-89-217-8.compute-1.amazonaws.com
The authenticity of host 'ec2-54-89-217-8.compute-1.amazonaws.com (54.89.217.8)'
 can't be established.
ED25519 key fingerprint is SHA256:+Of5/QpzofnJup3mimDOPuMOz25dueYCNnI/UOyOuJI.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-54-89-217-8.compute-1.amazonaws.com' (ED25519) t
o the list of known hosts.
Connection reset by 54.89.217.8 port 22
```

```
2022k@Komal22 MINGW64 ~/Downloads (main)
$ ssh -i "Komalnew.pem" ec2-user@ec2-54-89-217-8.compute-1.amazonaws.com
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
Last login: Sat Sep 14 09:23:20 2024 from 18.206.107.29
```

Docker Installation:

```
Last login: Sat Sep 14 10:03:54 2024 from 18.206.107.27
[ec2-user@ip-172-31-86-155 ~]$ sudo su
[root@ip-172-31-86-155 ec2-user]# sudo yum install docker -y
Last metadata expiration check: 1:30:24 ago on Sat Sep 14 09:05:49 2024.
Dependencies resolved.
================================================================================
 Package                 Architecture    Version                 Repository    Size
================================================================================
Installing:
 docker                  x86_64          25.0.6-1.amzn2023.0.2   amazonlinux   44 M
Installing dependencies:
 containerd              x86_64          1.7.20-1.amzn2023.0.1   amazonlinux   35 M
 iptables-libs           x86_64          1.8.8-3.amzn2023.0.2    amazonlinux  401 k
 iptables-nft            x86_64          1.8.8-3.amzn2023.0.2    amazonlinux  183 k
 libcgroup               x86_64          3.0-1.amzn2023.0.1      amazonlinux   75 k
 libnetfilter_conntrack  x86_64          1.0.8-2.amzn2023.0.2    amazonlinux   58 k
 libnfnetlink            x86_64          1.0.1-19.amzn2023.0.2   amazonlinux   30 k
 libnftnl                x86_64          1.2.2-2.amzn2023.0.2    amazonlinux   84 k
 pigz                    x86_64          2.5-1.amzn2023.0.3      amazonlinux   83 k
 runc                    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      6.5 MB/s | 401 kB     00:00
(2/10): iptables-nft-1.8.8-3.amzn2023.0.2.x86_64.rpm       4.1 MB/s | 183 kB     00:00
(3/10): libcgroup-3.0-1.amzn2023.0.1.x86_64.rpm            1.8 MB/s |  75 kB     00:00
(4/10): libnetfilter_conntrack-1.0.8-2.amzn2023.0.2.x86_64.rpm 2.8 MB/s |  58 kB     00:00
(5/10): libnfnetlink-1.0.1-19.amzn2023.0.2.x86_64.rpm      746 kB/s |  30 kB     00:00
(6/10): libnftnl-1.2.2-2.amzn2023.0.2.x86_64.rpm           3.0 MB/s |  84 kB     00:00
(7/10): pigz-2.5-1.amzn2023.0.3.x86_64.rpm                 1.8 MB/s |  83 kB     00:00
(8/10): runc-1.1.13-1.amzn2023.0.1.x86_64.rpm              9.4 MB/s | 3.2 MB     00:00
(9/10): containerd-1.7.20-1.amzn2023.0.1.x86_64.rpm         30 MB/s |  35 MB     00:01
(10/10): docker-25.0.6-1.amzn2023.0.2.x86_64.rpm            30 MB/s |  44 MB     00:01
--------------------------------------------------------------------------------
Total                                                       55 MB/s |  84 MB     00:01
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                                        1/1
  Installing       : runc-1.1.13-1.amzn2023.0.1.x86_64                     1/10
  Installing       : containerd-1.7.20-1.amzn2023.0.1.x86_64               2/10
  Running scriptlet: containerd-1.7.20-1.amzn2023.0.1.x86_64               2/10
  Installing       : pigz-2.5-1.amzn2023.0.3.x86_64                        3/10
  Installing       : libnftnl-1.2.2-2.amzn2023.0.2.x86_64                  4/10
  Installing       : libnfnetlink-1.0.1-19.amzn2023.0.2.x86_64             5/10
```

```
  Verifying        : runc-1.1.13-1.amzn2023.0.1.x86_64

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
  libnfnetlink-1.0.1-19.amzn2023.0.2.x86_64
  libnftnl-1.2.2-2.amzn2023.0.2.x86_64
  pigz-2.5-1.amzn2023.0.3.x86_64
  runc-1.1.13-1.amzn2023.0.1.x86_64

Complete!
```

```
For more help on how to use Docker, head to https://docs.docker.com/go/guides/
[root@ip-172-31-86-155 docker]# sudo systemctl start docker
sudo systemctl enable docker
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /usr/lib/systemd/system/docker.service.
[root@ip-172-31-86-155 docker]# sudo systemctl status docker
● docker.service - Docker Application Container Engine
     Loaded: loaded (/usr/lib/systemd/system/docker.service; enabled; preset: disabled)
     Active: active (running) since Sat 2024-09-14 10:48:29 UTC; 14s ago
TriggeredBy: ● docker.socket
       Docs: https://docs.docker.com
   Main PID: 30834 (dockerd)
      Tasks: 7
     Memory: 29.8M
        CPU: 243ms
     CGroup: /system.slice/docker.service
             └─30834 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock --default-ulimit nofile=3276>

Sep 14 10:48:29 ip-172-31-86-155.ec2.internal systemd[1]: Starting docker.service - Docker Application Container Engine.>
Sep 14 10:48:29 ip-172-31-86-155.ec2.internal dockerd[30834]: time="2024-09-14T10:48:29.147749917Z" level=info msg="Star>
Sep 14 10:48:29 ip-172-31-86-155.ec2.internal dockerd[30834]: time="2024-09-14T10:48:29.199782843Z" level=info msg="Load>
Sep 14 10:48:29 ip-172-31-86-155.ec2.internal dockerd[30834]: time="2024-09-14T10:48:29.588709885Z" level=info msg="Load>
Sep 14 10:48:29 ip-172-31-86-155.ec2.internal dockerd[30834]: time="2024-09-14T10:48:29.611632961Z" level=info msg="Dock>
Sep 14 10:48:29 ip-172-31-86-155.ec2.internal dockerd[30834]: time="2024-09-14T10:48:29.611926406Z" level=info msg="Daem>
Sep 14 10:48:29 ip-172-31-86-155.ec2.internal dockerd[30834]: time="2024-09-14T10:48:29.647372870Z" level=info msg="API >
Sep 14 10:48:29 ip-172-31-86-155.ec2.internal systemd[1]: Started docker.service - Docker Application Container Engine.
lines 1-20/20 (END)
```

Kubernetes installation:

```
[root@ip-172-31-86-155 docker]# sudo setenforce 0
[root@ip-172-31-86-155 docker]# sudo sed -i 's/^SELINUX=enforcing/SELINUX=permissive/' /etc/selinux/config
[root@ip-172-31-86-155 docker]# 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-172-31-86-155 docker]# sudo yum update
Kubernetes                                                                    66 kB/s | 9.4 kB     00:00
Dependencies resolved.
Nothing to do.
Complete!
```

```
[root@ip-172-31-86-155 docker]# sudo yum install -y kubelet kubeadm kubectl --disableexcludes=kubernetes
Last metadata expiration check: 0:02:35 ago on Sat Sep 14 10:52:57 2024.
Dependencies resolved.
================================================================================
 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_cttimeout-1.0.0-19.amzn2023.0.2.x86_64.rpm  398 kB/s |  24 kB     00:00

(2/9): libnetfilter_cthelper-1.0.0-21.amzn2023.0.2.x86_64.rpm   317 kB/s |  24 kB     00:00

(3/9): libnetfilter_queue-1.0.5-2.amzn2023.0.2.x86_64.rpm       1.4 MB/s |  30 kB     00:00

(4/9): conntrack-tools-1.4.6-2.amzn2023.0.2.x86_64.rpm          1.7 MB/s | 208 kB     00:00

(5/9): cri-tools-1.31.1-150500.1.1.x86_64.rpm                    33 MB/s | 6.9 MB     00:00

(6/9): kubectl-1.31.1-150500.1.1.x86_64.rpm                      34 MB/s |  11 MB     00:00

(7/9): kubeadm-1.31.1-150500.1.1.x86_64.rpm                      20 MB/s |  11 MB     00:00
```

```
/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-172-31-86-155 docker]# sudo swapoff -a
[root@ip-172-31-86-155 docker]# echo "net.bridge.bridge-nf-call-iptables=1" | sudo tee -a /etc/sysctl.conf
net.bridge.bridge-nf-call-iptables=1
[root@ip-172-31-86-155 docker]# sudo sysctl -p
net.bridge.bridge-nf-call-iptables = 1
net.bridge.bridge-nf-call-iptables = 1
net.bridge.bridge-nf-call-iptables = 1
```

Master machine:

```
[root@ip-172-31-86-155 docker]# sudo kubeadm init --ignore-preflight-errors=all
[init] Using Kubernetes version: v1.31.0
[preflight] Running pre-flight checks
        [WARNING NumCPU]: the number of available CPUs 1 is less than the required 2
        [WARNING Mem]: the system RAM (949 MB) is less than the minimum 1700 MB
        [WARNING FileExisting-socat]: socat not found in system path
        [WARNING FileExisting-tc]: tc not found in system path
        [WARNING Service-Kubelet]: kubelet service is not enabled, please run 'systemctl enable kubelet.service'
[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'
W0914 11:14:01.808435   32440 checks.go:846] detected that the sandbox image "registry.k8s.io/pause:3.8" of the contai
ner 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-172-31-86-155.ec2.internal kubernetes kubernetes.default ku
bernetes.default.svc kubernetes.default.svc.cluster.local] and IPs [10.96.0.1 172.31.86.155]
[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-172-31-86-155.ec2.internal localhost] and IPs [172.31.86.
155 127.0.0.1 ::1]
[certs] Generating "etcd/peer" certificate and key
[certs] etcd/peer serving cert is signed for DNS names [ip-172-31-86-155.ec2.internal localhost] and IPs [172.31.86.15
5 127.0.0.1 ::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
[kubeconfig] Writing "super-admin.conf" kubeconfig file
[kubeconfig] Writing "kubelet.conf" kubeconfig file
```

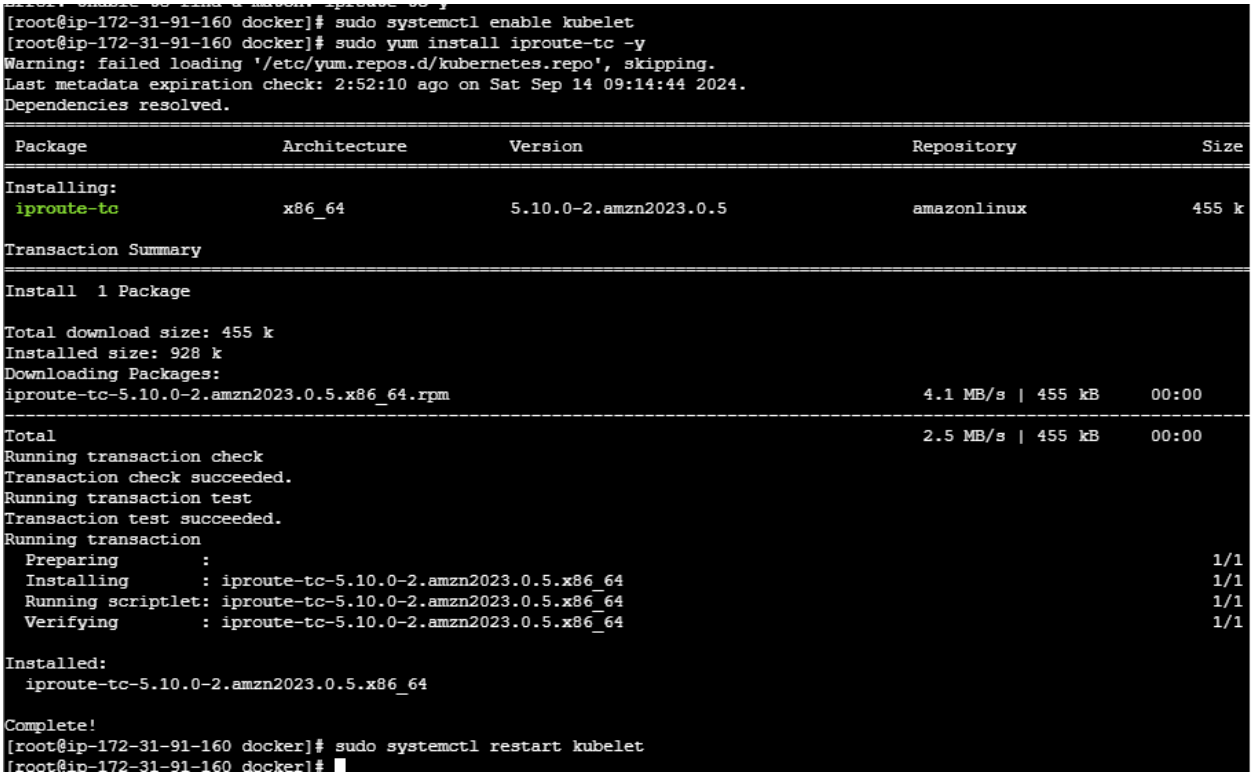
```
kubernetes.io/control-plane node.kubernetes.io/exclude-from-external-load-balancers]
[mark-control-plane] Marking the node ip-172-31-86-155.ec2.internal as control-plane by adding the taints [node-role.k
ubernetes.io/control-plane:NoSchedule]
[bootstrap-token] Using token: wx9djo.k8jr6g8juepvd40l
[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 ter
m certificate credentials
[bootstrap-token] Configured RBAC rules to allow the csrapprover controller automatically approve CSRs from a Node Boo
tstrap 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:

kubeadm join 172.31.86.155:6443 --token wx9djo.k8jr6g8juepvd40l \
        --discovery-token-ca-cert-hash sha256:e63842ad110988e7ac0d12e9e01a1f9ed865ca065db32a044fb2b597757c5c2b
```

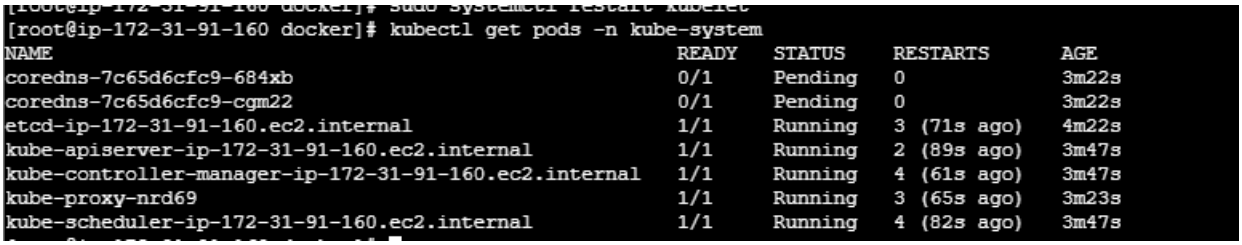
```
        --discovery-token-ca-cert-hash sha256:e63842ad110988e7ac0d12e9e01a1f9ed865ca065db32a044fb2b597757c5c2b
[root@ip-172-31-86-155 docker]# mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
```

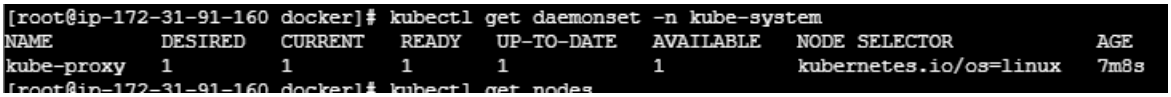
```
[root@ip-172-31-86-155 docker]# kubectl apply -f https://raw.githubusercontent.com/coreos/flannel/master/Documentat
/kube-flannel.yml
namespace/kube-flannel created
clusterrole.rbac.authorization.k8s.io/flannel created
clusterrolebinding.rbac.authorization.k8s.io/flannel created
serviceaccount/flannel created
configmap/kube-flannel-cfg created
daemonset.apps/kube-flannel-ds created
```

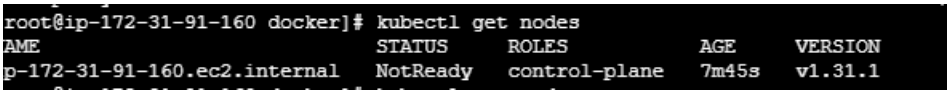
```
[root@ip-172-31-86-155 docker]# kubectl get pods
No resources found in default namespace.
[root@ip-172-31-86-155 docker]# sudo kubectl get pods -n kube-system
NAME                                                READY   STATUS             RESTARTS        AGE
coredns-7c65d6cfc9-nqdfc                            0/1     Pending            0               7m53s
coredns-7c65d6cfc9-vlstf                            0/1     Pending            0               7m53s
etcd-ip-172-31-86-155.ec2.internal                  1/1     Running            4 (96s ago)     8m2s
kube-apiserver-ip-172-31-86-155.ec2.internal        1/1     Running            3 (3m30s ago)   8m2s
kube-controller-manager-ip-172-31-86-155.ec2.internal   0/1     CrashLoopBackOff   5 (113s ago)    8m2s
kube-proxy-4vcr2                                    1/1     Running            5 (2m26s ago)   7m53s
kube-scheduler-ip-172-31-86-155.ec2.internal        1/1     Running            5 (107s ago)    8m2s
[root@ip-172-31-86-155 docker]#
```

Worker machine:









Conclusion: Here, we successfully created an EC2 instance on AWS Linux, insntalled Docker and Kubernetes on the same. Then, using a token I tried to make a connection of master nodes and worker nodes. However, despite of getting Ready status on running ‘kubectl get daemonset -n kube-system’,

The node status shows NotReady.