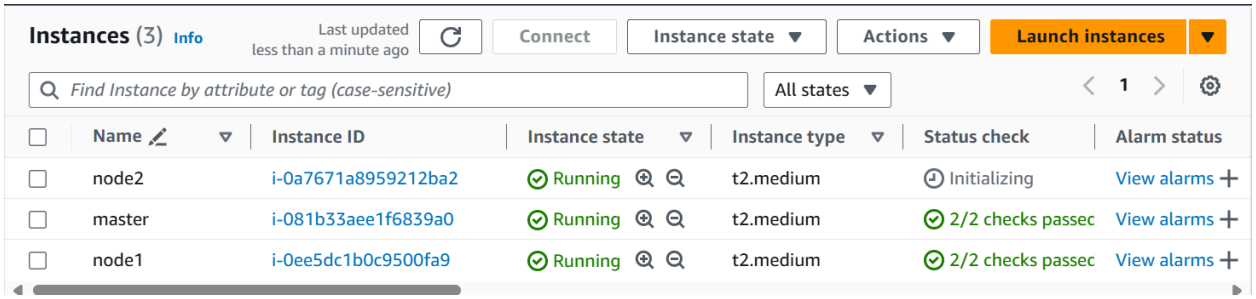
Name: Anish Kulkarni Roll No.: 29 Class: D15C AY: 2024-25

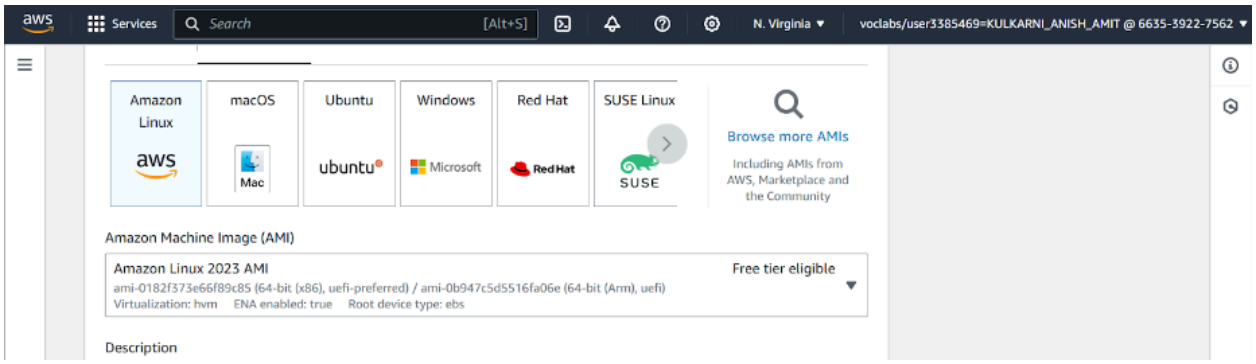
Experiment 3

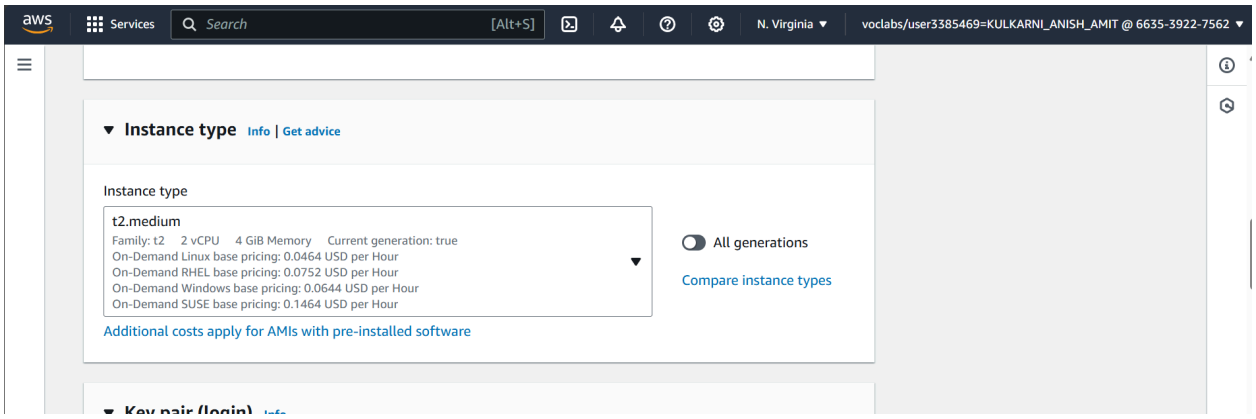
Aim: To understand the Kubernetes Cluster Architecture, install and Spin Up a Kubernetes Cluster on Linux Machines/Cloud Platforms.

Steps:

Step 1: Create 3 EC2 Amazon Linux instances on AWS (1 master and 2 other nodes). While doing so, make sure that t2.medium is selected as 'instance type' instead of the default t2.micro. This is because t2.medium provides more CPU, memory, and consistent performance, which are crucial for effectively running Kubernetes components and managing cluster workloads.







Step 2: To establish a connection with a remote server through SSH using the terminal, use the following command:-
ssh -i <keyname>.pem ubuntu@<public_ip_address>
where ‘keyname’ is the name of the key pair created by the user. “<keyname>.pem” is the name of the pem file of the key pair which is present in the ‘Downloads’ folder. The command “chmod 400 <keyname>.pem” is used to set the file permissions for the private key file (<keyname>.pem) so that only the file's owner can read it, and no one else can access or modify it.

```
anish@ANISH MINGW64 ~
$ cd Downloads

anish@ANISH MINGW64 ~/Downloads
$ chmod 400 "keypair1.pem"

anish@ANISH MINGW64 ~/Downloads
$ ssh -i "keypair1.pem" ec2-user@ec2-3-88-175-3.compute-1.amazonaws.com
The authenticity of host 'ec2-3-88-175-3.compute-1.amazonaws.com (3.88.175.3)' can't be established.
ED25519 key fingerprint is SHA256:BDZyiU2C3cXlWyKsTQUSoDxQyZm82EMdxoJTjfl53+s.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-3-88-175-3.compute-1.amazonaws.com' (ED25519) to the list of known hosts.
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
```

Step 3: Install docker on all 3 machines using the “yum install docker -y” command.

```
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
[ec2-user@ip-172-31-19-66 ~]$ sudo su
[root@ip-172-31-19-66 ec2-user]# yum install docker -y
Last metadata expiration check: 0:07:37 ago on Sat Sep 14 02:48:41 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
```

i-081b33aee1f6839a0 (master)

PublicIPs: 34.227.221.101 PrivateIPs: 172.31.19.66

```
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        : libnfnetlink-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        : pigz-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
  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!
[root@ip-172-31-19-66 ec2-user]#
```

i-081b33aee1f6839a0 (master)

PublicIPs: 34.227.221.101 PrivateIPs: 172.31.19.66

aws Services Search [Alt+S] N. Virginia voclabs/user3385469=KULKARNI_ANISH_AMIT @ 6635-3922-75

```
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
Last login: Sat Sep 14 02:53:50 2024 from 18.206.107.27
[ec2-user@ip-172-31-17-3 ~]$ sudo su
[root@ip-172-31-17-3 ec2-user]# yum install docker -y
Last metadata expiration check: 0:14:16 ago on Sat Sep 14 02:50:18 2024.
Dependencies resolved.
=================================================================================================================================
 Package                          Architecture          Version                              Repository                  Size
=================================================================================================================================
Installing:
 docker                           x86_64                25.0.6-1.amzn2023.0.2                amazonlinux                 44 M
Installing dependencies:
```

i-0a7671a8959212ba2 (node2)

PublicIPs: 52.91.112.45 PrivateIPs: 172.31.17.3

```
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        : libnfnetlink-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        : pigz-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
  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!
[root@ip-172-31-17-3 ec2-user]#
```

i-0a7671a8959212ba2 (node2)

PublicIPs: 52.91.112.45 PrivateIPs: 172.31.17.3

```
  ,     #_
  ~\_  ####_        Amazon Linux 2023
 ~~  \_#####\
 ~~     \###|
 ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
  ~~       V~' '->
   ~~~         /
     ~~._.   _/
        _/ _/
      _m/'
Last login: Sat Sep 14 02:54:13 2024 from 18.206.107.28
[ec2-user@ip-172-31-22-72 ~]$ sudo su
[root@ip-172-31-22-72 ec2-user]# yum install docker -y
Last metadata expiration check: 0:16:32 ago on Sat Sep 14 02:49:23 2024.
Dependencies resolved.
================================================================================================================================
 Package                                Architecture          Version                                Repository                Size
================================================================================================================================
Installing:
 docker                                 x86_64                25.0.6-1.amzn2023.0.2                  amazonlinux               44 M
Installing dependencies:
```

i-0ee5dc1b0c9500fa9 (node1)

PublicIPs: 54.209.154.170 PrivateIPs: 172.31.22.72

```
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        : libnfnetlink-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        : pigz-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
  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!
[root@ip-172-31-22-72 ec2-user]#
```

i-0ee5dc1b0c9500fa9 (node1)

PublicIPs: 54.209.154.170 PrivateIPs: 172.31.22.72

Step 4: Use the “systemctl start docker” command on all 3 machines to start the Docker service, allowing the user to manage and run Docker containers on the system.

```
[root@ip-172-31-19-66 ec2-user]# systemctl start docker
```

```
[root@ip-172-31-17-3 ec2-user]# systemctl start docker
```

```
[root@ip-172-31-22-72 ec2-user]# systemctl start docker
```

Step 5: Configure Docker on all 3 machines to use systemd for managing cgroups by updating its configuration file, ensuring Docker starts automatically on boot, reloading the systemd configuration, and restarting Docker to apply the changes. Use the following commands to do so:

```
cd /etc/docker
cat <<EOF | sudo tee /etc/docker/daemon.json
{
"exec-opts": ["native.cgroupdriver=systemd"]
}
EOF
sudo systemctl enable docker
sudo systemctl daemon-reload
sudo systemctl restart docker
```

Step 6: Install kubernetes on all 3 machines using the following commands:

```
sudo tee /etc/yum.repos.d/kubernetes.repo <<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
EOF

sudo setenforce 0
sudo sed -i 's/^SELINUX=enforcing$/SELINUX=permissive/' /etc/selinux/config

sudo yum clean all
sudo yum install -y kubelet kubeadm kubectl --disableexcludes=Kubernetes

sudo systemctl enable --now kubelet
```

```
[ec2-user@ip-172-31-19-66 ~]$ # Update the Kubernetes repo file and install the required packages
sudo tee /etc/yum.repos.d/kubernetes.repo <<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
EOF

# Set SELinux to permissive
sudo setenforce 0
sudo sed -i 's/^SELINUX=enforcing$/SELINUX=permissive/' /etc/selinux/config

# Clean yum cache and install kubelet, kubeadm, and kubectl
sudo yum clean all
sudo yum install -y kubelet kubeadm kubectl --disableexcludes=Kubernetes

# Enable and start kubelet
sudo systemctl enable --now kubelet
```

```
  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!
Created symlink /etc/systemd/system/multi-user.target.wants/kubelet.service → /usr/lib/systemd/system/kubelet.service.
[ec2-user@ip-172-31-19-66 ~]$ yum repolist
```

i-081b33aee1f6839a0 (master)

PublicIPs: 34.227.221.101 PrivateIPs: 172.31.19.66

```
  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!
Created symlink /etc/systemd/system/multi-user.target.wants/kubelet.service → /usr/lib/systemd/system/kubelet.service.
[root@ip-172-31-17-3 ec2-user]#
```

i-0a7671a8959212ba2 (node2)

PublicIPs: 52.91.112.45 PrivateIPs: 172.31.17.3

```
  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!
Created symlink /etc/systemd/system/multi-user.target.wants/kubelet.service → /usr/lib/systemd/system/kubelet.service.
[root@ip-172-31-22-72 ec2-user]#
```

i-0ee5dc1b0c9500fa9 (node1)

PublicIPs: 54.209.154.170 PrivateIPs: 172.31.22.72

Step 7: Initialise the kubernetes cluster by using the “sudo kubeadm init” command **only on the master machine.**

```
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 172.31.19.66:6443 --token vgt0c0.9wv1jt67cb7z9bm1 \
        --discovery-token-ca-cert-hash sha256:b6ddb13c29b230866e008c40098f09b6a9eadde1e81ea1e98b9125401af8c317
[ec2-user@ip-172-31-19-66 ~]$
```

i-081b33aee1f6839a0 (master)

PublicIPs: 34.227.221.101 PrivateIPs: 172.31.19.66

Step 8: Copy the mkdir and chown commands from the top and execute them:
mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config
export KUBECONFIG=/etc/kubernetes/admin.conf

Step 9: Execute the “sudo kubectl get nodes –kubeconfig=/etc/kubernetes/admin.conf” command to get a list of the nodes present in the kubernetes cluster. It is observed that for the time being, only the master machine is present in the kubernetes cluster.

```
[ec2-user@ip-172-31-19-66 ~]$ sudo kubectl get nodes --kubeconfig=/etc/kubernetes/admin.conf
NAME                                STATUS     ROLES           AGE     VERSION
ip-172-31-19-66.ec2.internal        NotReady   control-plane   7m36s   v1.31.1
```

Step 10: Copy the following part from the output of the “sudo kubeadm init” command:
sudo kubeadm join <ip> --token <token> \
--discovery-token-ca-cert-hash <hash>
Execute the above commands only on the worker machines. It adds the worker nodes to an existing Kubernetes cluster by connecting it to the master node specified by <ip>, using the provided authentication token and certificate hash for secure communication.

```
[ec2-user@ip-172-31-17-3 ~]$ sudo kubeadm join 172.31.19.66:6443 --token vgt0c0.9wv1jt67cb7z9bm1 \
        --discovery-token-ca-cert-hash sha256:b6ddb13c29b230866e008c40098f09b6a9eadde1e81ea1e98b9125401af8c317
[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://172.31.19.66:6443/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
[ec2-user@ip-172-31-17-3 ~]$
```

i-0a7671a8959212ba2 (node2)

PublicIPs: 52.91.112.45 PrivateIPs: 172.31.17.3

```
[root@ip-172-31-22-72 ec2-user]# sudo kubeadm join 172.31.19.66:6443 --token vgt0c0.9wv1jt67cb7z9bm1 \
        --discovery-token-ca-cert-hash sha256:b6ddb13c29b230866e008c40098f09b6a9eadde1e81ea1e98b9125401af8c317
[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://172.31.19.66:6443/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-172-31-22-72 ec2-user]#
```

i-0ee5dc1b0c9500fa9 (node1)

PublicIPs: 54.209.154.170 PrivateIPs: 172.31.22.72

On executing the previously mentioned commands, it is observed that the above error occurs on both the worker machines. This is due to the inability of the worker machines to get added to the kubernetes cluster within a deadline (a certain fixed amount of time).

Conclusion:

1. In the above experiment, we learned how to install and spin up a Kubernetes Cluster on Linux Machines/Cloud Platforms.
2. First, we created 3 EC2 Amazon Linux instances on AWS (1 master and 2 worker nodes) and established their connections with a remote server through SSH.
3. Next, we installed, configured and started docker on all 3 machines.
4. Then, we installed kubernetes on all 3 machines.
5. Next, we initialised kubernetes on only the master machine and added the master machine to the kubernetes cluster.
6. Next, we tried to add the worker machines to the kubernetes cluster by executing the “join” command. Here, we encountered an error due to the failure of the worker machines to be added to the kubernetes cluster.