SHRAVANI ANIL PATIL **D15A-38**

Advanced DevOps Lab

Experiment:3

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

Theory:

To understand Kubernetes Cluster Architecture and how to install and spin up a Kubernetes cluster on Linux machines or cloud platforms, it's essential to grasp the fundamental components and design principles of Kubernetes.

Overview of Kubernetes

Kubernetes is an open-source container orchestration platform developed by Google, designed to automate the deployment, scaling, and management of containerized applications. It provides a robust infrastructure that supports microservices architecture, offering features such as self-healing, scaling, and zero-downtime deployments. Kubernetes can run on various environments, including public clouds (like AWS and Azure), private clouds, and bare metal servers.

Kubernetes Architecture

Kubernetes architecture is primarily composed of two main components: the Control Plane and the Data Plane.

Control Plane

The Control Plane manages the overall state of the Kubernetes cluster and includes several key components:

- kube-apiserver: The API server acts as the gateway for all interactions with the cluster, processing REST requests and managing the state of the cluster.
- etcd: A distributed key-value store that holds the configuration data and state of the cluster, ensuring consistency and availability.
- kube-scheduler: Responsible for assigning Pods to worker nodes based on resource availability and other constraints.
- kube-controller-manager: Manages controllers that regulate the state of the cluster, ensuring that the desired state matches the actual state.

- cloud-controller-manager (optional): Integrates with cloud provider APIs to manage resources specific to the cloud environment.

Data Plane

The Data Plane consists of the worker nodes that run the containerized applications. Each worker node includes:

- kubelet: An agent that ensures containers are running in Pods. It communicates with the Control Plane to receive instructions.
- kube-proxy: Maintains network rules and facilitates communication between Pods and services.
- Container Runtime: Software responsible for running containers, such as Docker or containerd.

Core Concepts

Key concepts in Kubernetes include:

- Pods: The smallest deployable units in Kubernetes, which can contain one or more containers.
- Services: Abstracts a set of Pods, providing a stable network endpoint for accessing them.
- Deployments: Define the desired state for Pods and manage their lifecycle, including scaling and updates.

Installing and Spinning Up a Kubernetes Cluster

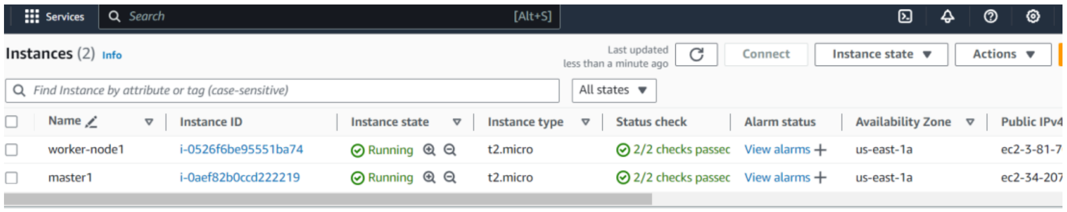
To install and set up a Kubernetes cluster, follow these general steps:

1. Choose an Environment: Decide whether to deploy on local machines or a cloud platform. For cloud platforms, services like Google Kubernetes Engine (GKE), Amazon EKS, or Azure AKS can simplify the process.
2. Install Prerequisites: Ensure that you have the necessary tools installed, such as kubectl (the command-line tool for interacting with the cluster) and a container runtime.
3. Set Up the Control Plane: This can be done using tools like kubeadm, which helps bootstrap the cluster by initializing the Control Plane components.

4. Join Worker Nodes: Once the Control Plane is set up, you can join worker nodes to the cluster using the token generated during the initialization.
5. Deploy Applications: After the cluster is up and running, you can deploy your applications using YAML configuration files that define the desired state of your Pods and Services..

Steps:

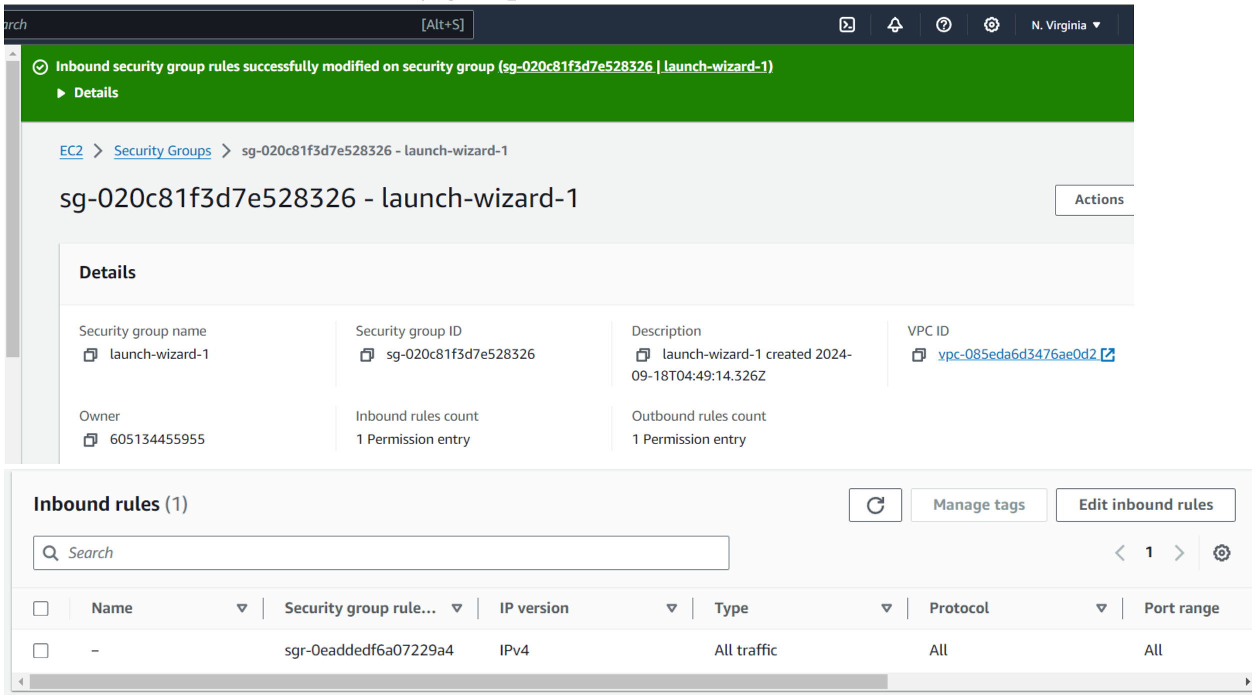
1.Creation of 2 EC2 Ubuntu Instances on AWS.


Services
Search [Alt+S]
Last updated less than a minute ago
Connect
Instance state
Actions
Find Instance by attribute or tag (case-sensitive)
All states

Instances (2) Info

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4
worker-node1	i-0526f6be95551ba74	Running	t2.micro	2/2 checks passec	View alarms	us-east-1a	ec2-3-81-7
master1	i-0aef82b0ccd222219	Running	t2.micro	2/2 checks passec	View alarms	us-east-1a	ec2-34-207

2.Edit inbound rules of security group ‘launch-wizard-1’ and set ‘All Traffic’



Inbound security group rules successfully modified on security group (sg-020c81f3d7e528326 | launch-wizard-1)

sg-020c81f3d7e528326 - launch-wizard-1

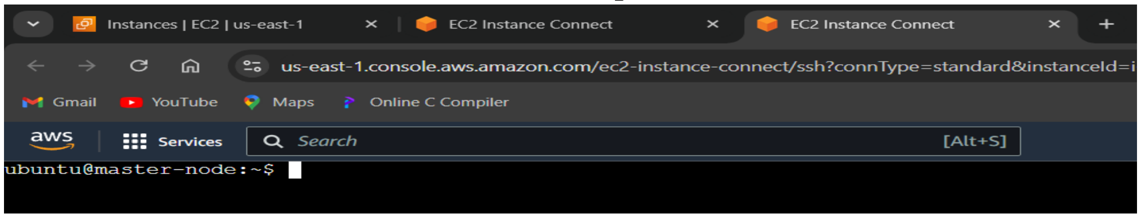
Details

Security group name	Security group ID	Description	VPC ID
launch-wizard-1	sg-020c81f3d7e528326	launch-wizard-1 created 2024-09-18T04:49:14.326Z	vpc-085eda6d3476ae0d2
Owner	Inbound rules count	Outbound rules count	
605134455955	1 Permission entry	1 Permission entry	

Inbound rules (1)

Name	Security group rule...	IP version	Type	Protocol	Port range
–	sgr-0eaddedf6a07229a4	IPv4	All traffic	All	All

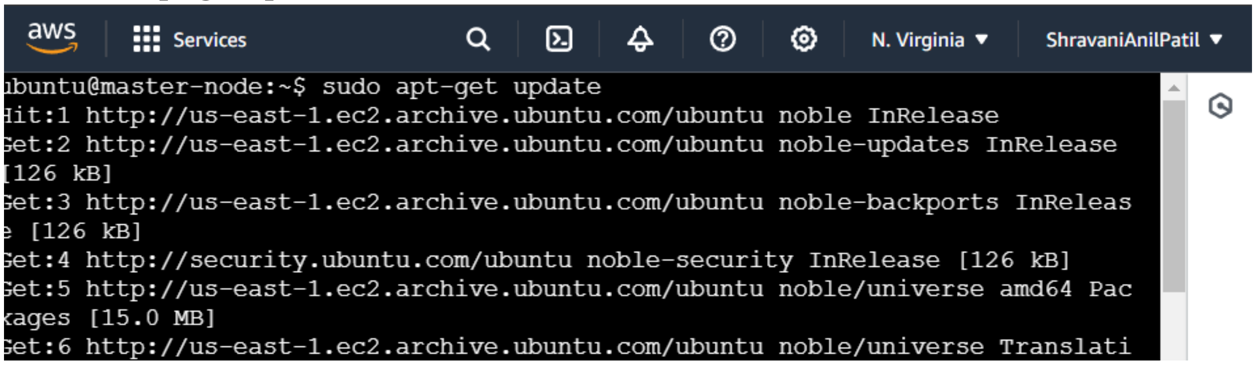
3. Set master and worker as hostname on respective servers


Instances | EC2 | us-east-1
EC2 Instance Connect
EC2 Instance Connect
us-east-1.console.aws.amazon.com/ec2-instance-connect/ssh?connType=standard&instanceId=i
Gmail
YouTube
Maps
Online C Compiler
aws
Services
Search
[Alt+S]

```
ubuntu@master-node:~$
```

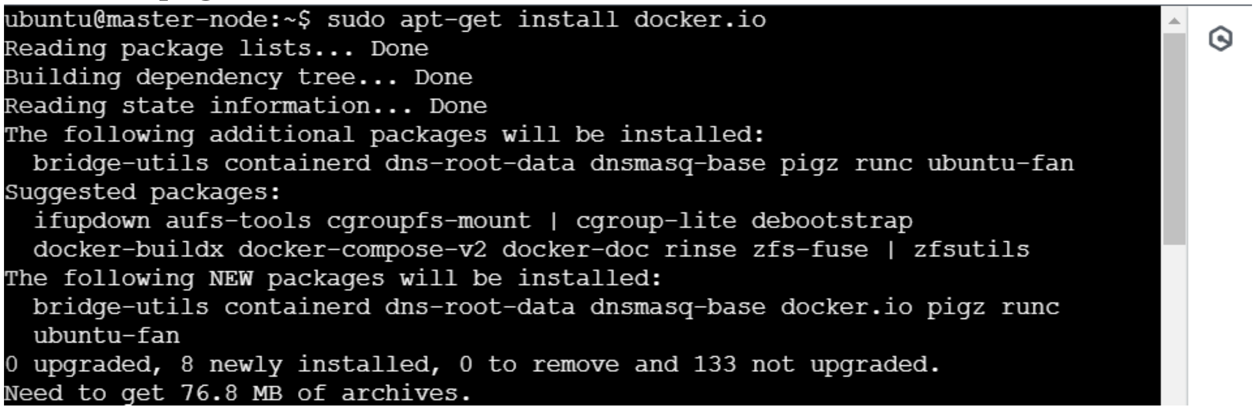
4.Installation of docker

4.1 - sudo apt-get update



```
ubuntu@master-node:~$ sudo apt-get update
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
[126 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InReleas
e [126 kB]
Get:4 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Pac
kages [15.0 MB]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe Translati
```

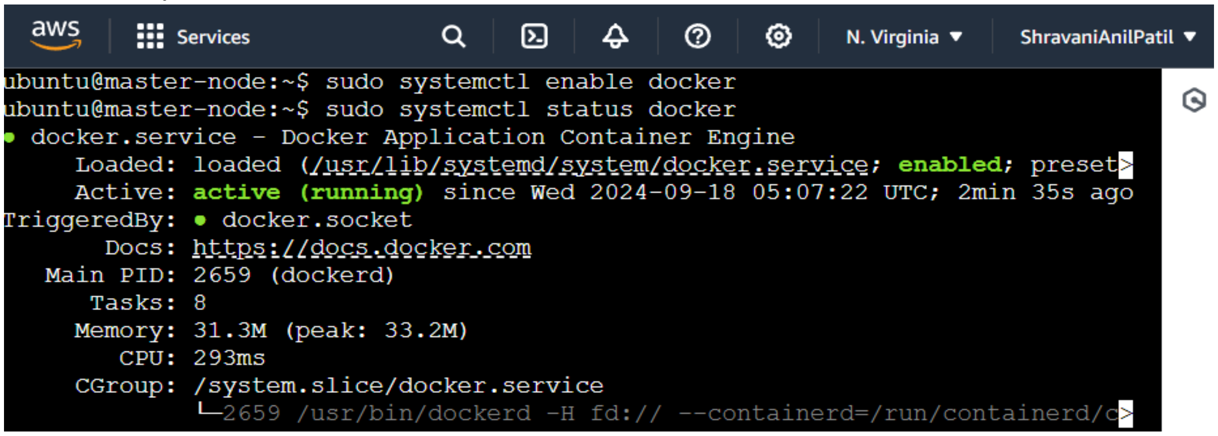
4.2 - sudo apt-get install docker.io



```
ubuntu@master-node:~$ sudo apt-get install docker.io
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  bridge-utils containerd dns-root-data dnsmasq-base pigz runc ubuntu-fan
Suggested packages:
  ifupdown aufs-tools cgroupfs-mount | cgroup-lite debootstrap
  docker-buildx docker-compose-v2 docker-doc rinse zfs-fuse | zfsutils
The following NEW packages will be installed:
  bridge-utils containerd dns-root-data dnsmasq-base docker.io pigz runc
  ubuntu-fan
0 upgraded, 8 newly installed, 0 to remove and 133 not upgraded.
Need to get 76.8 MB of archives.
```

4.3 – sudo systmectl enable docker

4.4 – sudo systemctl status docker

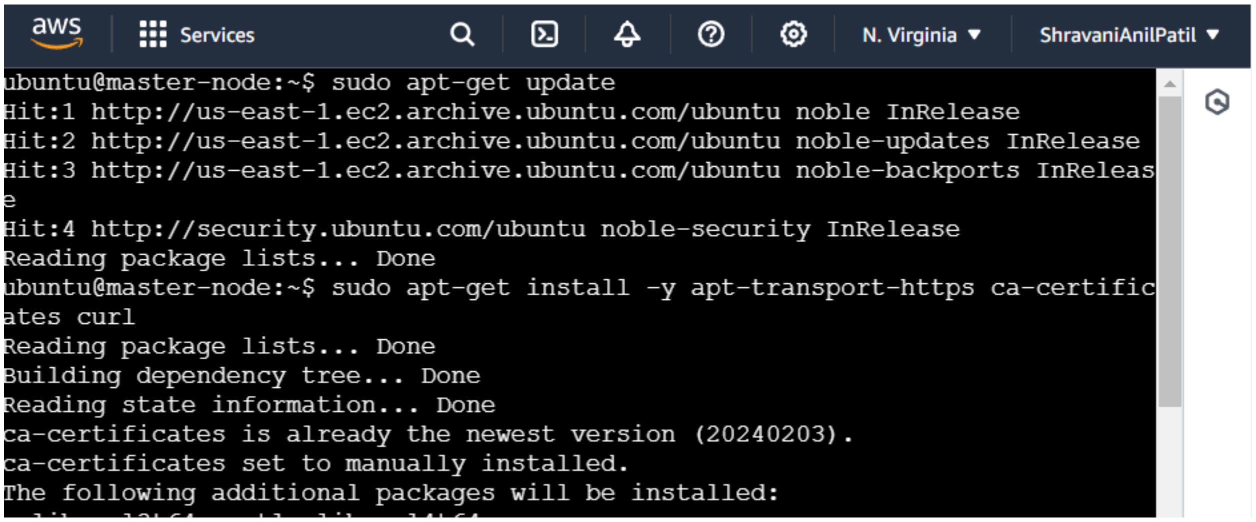


```
ubuntu@master-node:~$ sudo systemctl enable docker
ubuntu@master-node:~$ sudo systemctl status docker
● docker.service - Docker Application Container Engine
     Loaded: loaded (/usr/lib/systemd/system/docker.service; enabled; preset>
     Active: active (running) since Wed 2024-09-18 05:07:22 UTC; 2min 35s ago
TriggeredBy: ● docker.socket
       Docs: https://docs.docker.com
   Main PID: 2659 (dockerd)
      Tasks: 8
     Memory: 31.3M (peak: 33.2M)
        CPU: 293ms
     CGroup: /system.slice/docker.service
             └─2659 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/c>
```

5- Installation of Kubernetes-

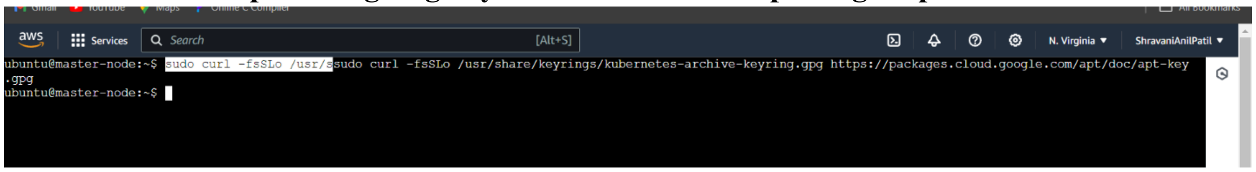
5.1 sudo apt-get update

5.2 install ca certificate



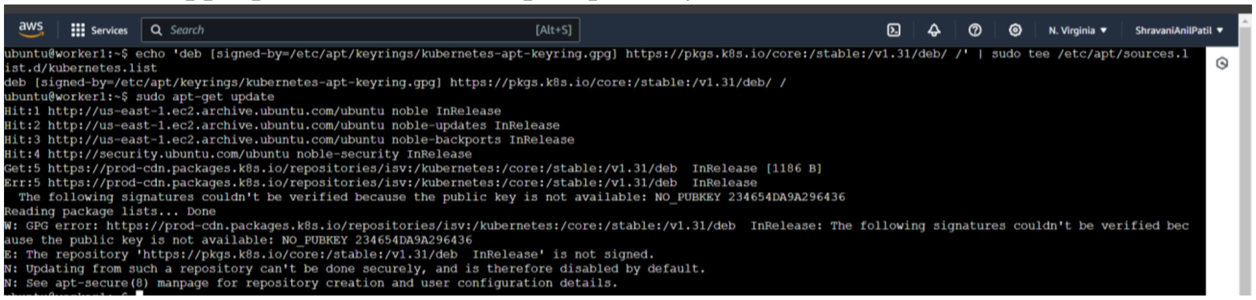
```
ubuntu@master-node:~$ sudo apt-get update
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InReleas
e
Hit:4 http://security.ubuntu.com/ubuntu noble-security InRelease
Reading package lists... Done
ubuntu@master-node:~$ sudo apt-get install -y apt-transport-https ca-certific
ates curl
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
ca-certificates is already the newest version (20240203).
ca-certificates set to manually installed.
The following additional packages will be installed:
```

5.3 Download the public signing key for the Kubernetes package repositories.



```
ubuntu@master-node:~$ sudo curl -fsSLo /usr/ssudo curl -fsSLo /usr/share/keyrings/kubernetes-archive-keyring.gpg https://packages.cloud.google.com/apt/doc/apt-key
.gpg
ubuntu@master-node:~$
```

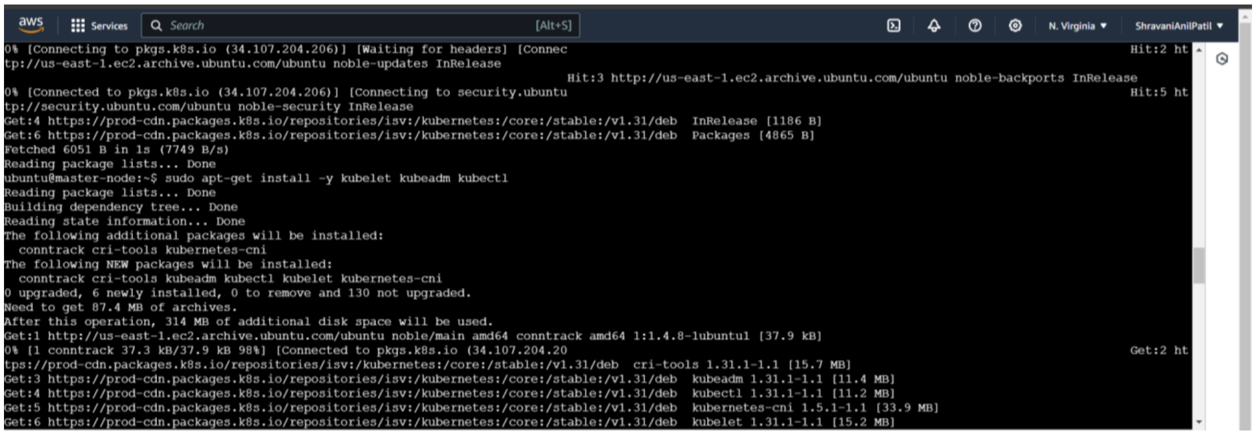
5.4 Add the appropriate Kubernetes apt repository



```
ubuntu@worker1:~$ echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.31/deb/ /' | sudo tee /etc/apt/sources.l
ist.d/kubernetes.list
deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.31/deb/ /
ubuntu@worker1:~$ sudo apt-get update
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu noble-security InRelease
Get:5 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb  InRelease [1186 B]
Err:5 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb  InRelease
  The following signatures couldn't be verified because the public key is not available: NO_PUBKEY 234654DA9A296436
Reading package lists... Done
W: GPG error: https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb  InRelease: The following signatures couldn't be verified bec
ause the public key is not available: NO_PUBKEY 234654DA9A296436
E: The repository 'https://pkgs.k8s.io/core:/stable:/v1.31/deb  InRelease' is not signed.
N: Updating from such a repository can't be done securely, and is therefore disabled by default.
N: See apt-secure(8) manpage for repository creation and user configuration details.
```

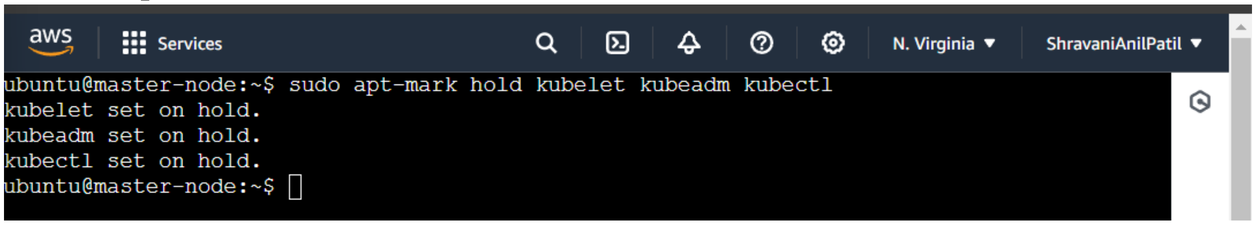
5.5 sudo apt-get update

5.6 sudo apt-get install -y kubelet kubeadm kubectl



```
0% [Connecting to pkgs.k8s.io (34.107.204.206)] [Waiting for headers] [Connec                                                     Hit:2 ht
tp://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
                                                                             Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
0% [Connected to pkgs.k8s.io (34.107.204.206)] [Connecting to security.ubuntu                                                    Hit:5 ht
tp://security.ubuntu.com/ubuntu noble-security InRelease
Get:4 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb  InRelease [1186 B]
Get:6 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb  Packages [4865 B]
Fetched 6051 B in 1s (7749 B/s)
Reading package lists... Done
ubuntu@master-node:~$ sudo apt-get install -y kubelet kubeadm kubectl
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  conntrack cri-tools kubernetes-cni
The following NEW packages will be installed:
  conntrack cri-tools kubeadm kubectl kubelet kubernetes-cni
0 upgraded, 6 newly installed, 0 to remove and 130 not upgraded.
Need to get 87.4 MB of archives.
After this operation, 314 MB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/main amd64 conntrack amd64 1:1.4.8-1ubuntu1 [37.9 kB]
0% [1 conntrack 37.3 kB/37.9 kB 98%] [Connected to pkgs.k8s.io (34.107.204.20                                                    Get:2 ht
tps://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb  cri-tools 1.31.1-1.1 [15.7 MB]
Get:3 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb  kubeadm 1.31.1-1.1 [11.4 MB]
Get:4 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb  kubectl 1.31.1-1.1 [11.2 MB]
Get:5 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb  kubernetes-cni 1.5.1-1.1 [33.9 MB]
Get:6 https://prod-cdn.packages.k8s.io/repositories/isv:/kubernetes:/core:/stable:/v1.31/deb  kubelet 1.31.1-1.1 [15.2 MB]
```

5.7 sudo apt-mark hold kubelet kubeadm kubectl



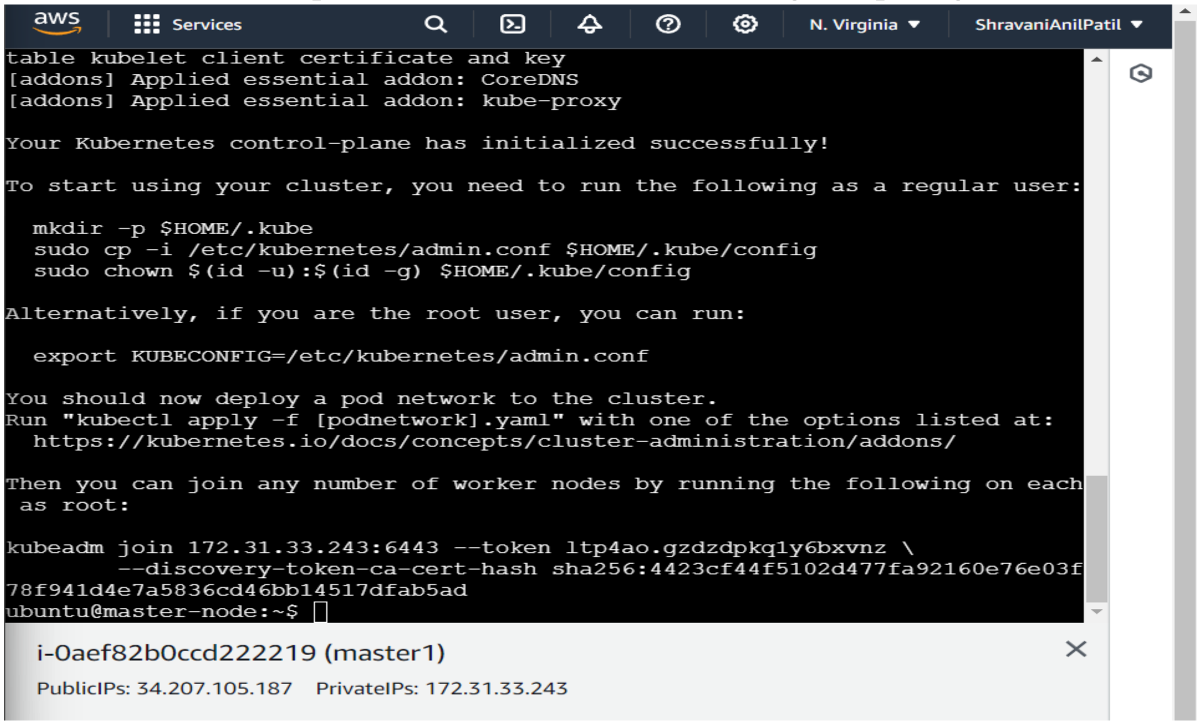
```
ubuntu@master-node:~$ sudo apt-mark hold kubelet kubeadm kubectl
kubelet set on hold.
kubeadm set on hold.
kubectl set on hold.
ubuntu@master-node:~$ 
```

6. Kubernetes Deployment

6.1 sudo swapoff –a

6.2 Initialize Kubernetes on Master Node -

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



```
table 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 172.31.33.243:6443 --token ltp4ao.gzdzdpkq1y6bxvnz \
        --discovery-token-ca-cert-hash sha256:4423cf44f5102d477fa92160e76e03f
78f941d4e7a5836cd46bb14517dfab5ad
ubuntu@master-node:~$ 
```

i-0aef82b0ccd222219 (master1)
PublicIPs: 34.207.105.187   PrivateIPs: 172.31.33.243

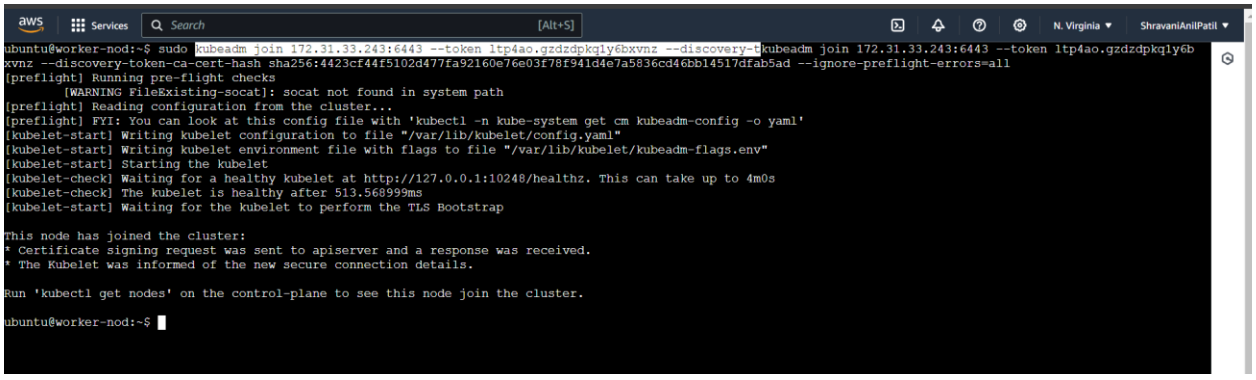
7.to create a directory for the cluster:

7.1mkdir -p $HOME/.kube

7.2sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config

7.3sudo chown HOME/.kube/config

8. Deploy Pod Network to Cluster and Join Worker Node to Cluster

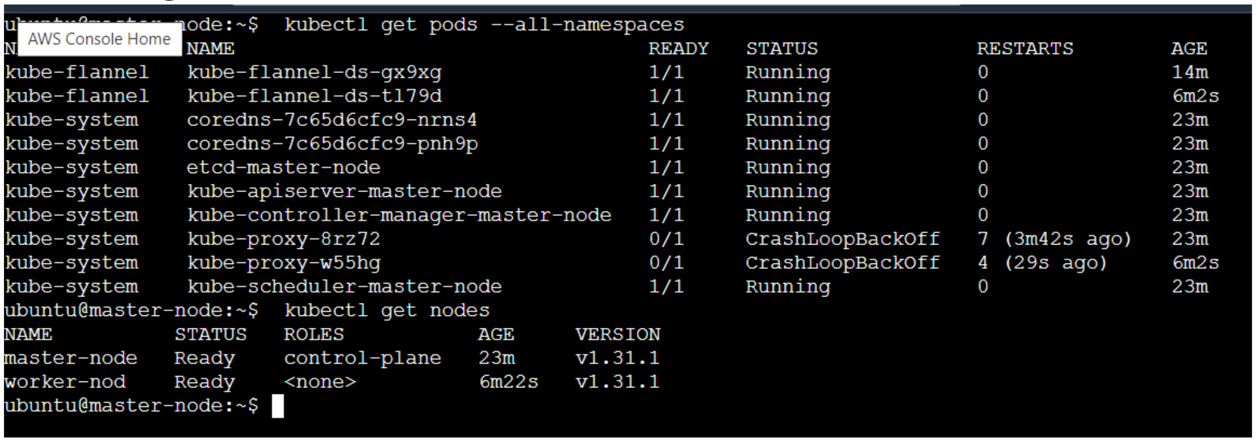


```
ubuntu@worker-nod:~$ sudo kubeadm join 172.31.33.243:6443 --token ltp4ao.gzdzdpkq1y6bxvnz --discovery-tkubeadm join 172.31.33.243:6443 --token ltp4ao.gzdzdpkq1y6b
xvnz --discovery-token-ca-cert-hash sha256:4423cf44f5102d477fa92160e76e03f78f941d4e7a5836cd46bb14517dfab5ad --ignore-preflight-errors=all
[preflight] Running pre-flight checks
        [WARNING FileExisting-socat]: socat not found in system path
[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-check] Waiting for a healthy kubelet at http://127.0.0.1:10248/healthz. This can take up to 4m0s
[kubelet-check] The kubelet is healthy after 513.568999ms
[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.

ubuntu@worker-nod:~$
```

Verify that everything is running and communicating:

8.1kubectl get pods --all-namespaces

8.2kubectl get nodes



```
ubuntu@master-node:~$  kubectl get pods --all-namespaces
N              NAME                                   READY   STATUS             RESTARTS        AGE
kube-flannel   kube-flannel-ds-gx9xg                  1/1     Running            0               14m
kube-flannel   kube-flannel-ds-tl79d                  1/1     Running            0               6m2s
kube-system    coredns-7c65d6cfc9-nrns4               1/1     Running            0               23m
kube-system    coredns-7c65d6cfc9-pnh9p               1/1     Running            0               23m
kube-system    etcd-master-node                       1/1     Running            0               23m
kube-system    kube-apiserver-master-node             1/1     Running            0               23m
kube-system    kube-controller-manager-master-node    1/1     Running            0               23m
kube-system    kube-proxy-8rz72                       0/1     CrashLoopBackOff   7 (3m42s ago)   23m
kube-system    kube-proxy-w55hg                       0/1     CrashLoopBackOff   4 (29s ago)     6m2s
kube-system    kube-scheduler-master-node             1/1     Running            0               23m
ubuntu@master-node:~$  kubectl get nodes
NAME           STATUS   ROLES           AGE     VERSION
master-node    Ready    control-plane   23m     v1.31.1
worker-nod     Ready    <none>          6m22s   v1.31.1
ubuntu@master-node:~$
```