ADITYA AHUA D15C 02

AdvDevops Experiment 3

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

STEPS :

1.We have created 3 instances

☐	Master	i-0767a02f53056b254	Running	t3.small	Initializing	View alarms +	us-east-1c	ec2-44-202-26-210.co.
☐	worker-1	i-0a98404682c2bf690	Running	t3.small	Initializing	View alarms +	us-east-1c	ec2-18-206-158-113.cc
☐	worker-2	i-0a9ea3e263873d151	Running	t3.small	Initializing	View alarms +	us-east-1c	ec2-3-89-36-106.comp

2. Now connect the three instances
And write the command on the three of the linux command promp.

sudo su
And yum install docker -y (To download docker in All three machines)

Master Worker1 Worker2

```
[ec2-user@ip-172-31-93-226 ~]$ sudo su
[root@ip-172-31-93-226 ec2-user]# yum install docker -y
Last metadata expiration check: 0:07:12 ago on Fri Sep 13 11:58:42 2024.
Dependencies resolved.
==========================================================================================================================================
======================================
 Package                                          Architecture                    Version                                          Repo
sitory                              Size
==========================================================================================================================================
======================================
Installing:
 docker                                           x86_64                          25.0.6-1.amzn2023.0.2                            amaz
onlinux                             44 M
Installing dependencies:
 containerd                                       x86_64                          1.7.20-1.amzn2023.0.1                            amaz
onlinux                             35 M
 iptables-libs                                    x86_64                          1.8.8-3.amzn2023.0.2                             amaz
onlinux                            401 k
 iptables-nft                                     x86_64                          1.8.8-3.amzn2023.0.2                             amaz
onlinux                            183 k
```

3. Now to start docker write command systemctl start docker in each instance i.e Master Worker1 Worker2

```
[root@ip-172-31-93-226 ec2-user]# systemctl start docker
[root@ip-172-31-93-226 ec2-user]# █
```

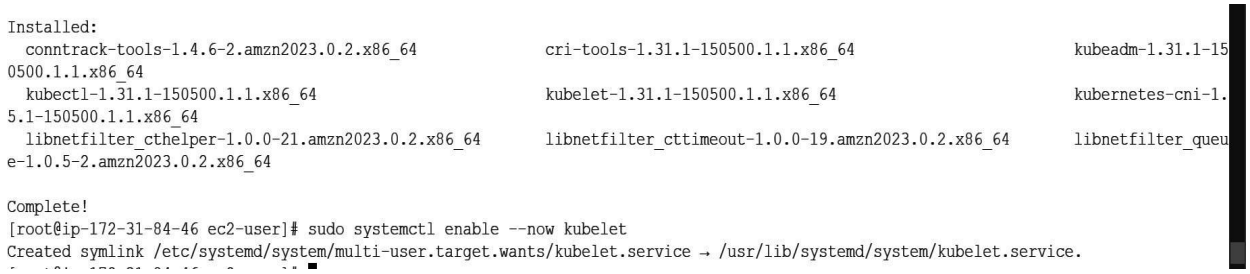
4. Now to install

kubeadm sudo

setenforce 0

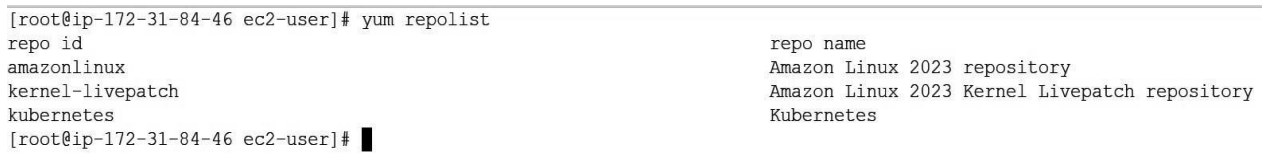
```
sudo sed -i 's/^SELINUX=enforcing$/SELINUX=permissive/' /etc/selinux/config
cat <<EOF | sudo tee /etc/yum.repos.d/kubernetes.repo
[kubernetes]
name=Kubernetes
baseurl=https://pkgs.k8s.io/core:/stable:/v1.31/rpm/
enabled=1
gpgcheck=1
gpgkey=https://pkgs.k8s.io/core:/stable:/v1.31/rpm/repodata/repomd.xml.key
exclude=kubelet kubeadm kubectl cri-tools kubernetes-cni
EOF

sudo yum install -y kubelet kubeadm kubectl --disableexcludes=kubernetes
sudo systemctl enable --now kubelet
```

Master Worker1 Worker2



5. Now to confirm that we have got repository for kubernets we will write the command yum repo list

Master Worker1 Worker2



6.NOW IN THE MASTER NODE WE NEED TO INITIALIZE KUBEADM

Only in Master

```
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 172.31.84.46:6443 --token jl06ac.t7cdzxf0x5eddmsl \
        --discovery-token-ca-cert-hash sha256:4a152b913ee4b60dc2126d55f631b86d0dafb7d58132416c4f32f0668ac553be
[root@ip-172-31-84-46 ec2-user]#
```

In the screenshot you can see the commands written in the 3rd 4th and 5th line
Copy that command , this command is used to add right permission to the user
Also copy the 7th line , here it is the credential for the user .
Also copy the last 2 lines it is a link used to join the nodes

NOW IN THE MASTER WRITE THE COMMAND WHICH YOU COPIED IN FIRST AND SECOND TIME.

This is the join link which you need to enter in Worker1 and Worker2
kubeadm join 172.31.84.46:6443 --token jl06ac.t7cdzxf0x5eddmsl \
--discovery-token-ca-cert-hash
sha256:4a152b913ee4b60dc2126d55f631b86d0dafb7d58132416c4f32f0668ac553be

Now in the master we will write the command
kubectl get node

This was the output in master node

```
[root@ip-172-31-84-46 ec2-user]# kubectl get node
NAME                           STATUS     ROLES           AGE   VERSION
ip-172-31-84-46.ec2.internal   NotReady   control-plane   56s   v1.31.1
[root@ip-172-31-84-46 ec2-user]# kubectl get node
The connection to the server 172.31.84.46:6443 was refused - did you specify the right host or port?
[root@ip-172-31-84-46 ec2-user]# kubectl get node
NAME                           STATUS     ROLES           AGE    VERSION
ip-172-31-84-46.ec2.internal   NotReady   control-plane   5m1s   v1.31.1
[root@ip-172-31-84-46 ec2-user]#
```

And this was the output when i tried connecting it with worker 1 and worker 2

```
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
Last login: Fri Sep 13 12:04:33 2024 from 18.206.107.28
[ec2-user@ip-172-31-87-149 ~]$ sudo su
[root@ip-172-31-87-149 ec2-user]# kubeadm join 172.31.84.46:6443 --token qq448c.8c0zquwywz3eqt0b         --discovery-token-ca-cert-has
h sha256:49cc770e646aab704883a3d9fb30e6146fdd83e782ebd5ec3194dacee59f2257
[preflight] Running pre-flight checks
        [WARNING FileExisting-socat]: socat not found in system path
        [WARNING FileExisting-tc]: tc not found in system path
```

There was an issue in connecting worker 1 and worker 2 to the master node, despite providing the correct joining link. The Cloud Shell did not progress beyond that point.

CONCLUSION:

During the setup of a Kubernetes cluster, several issues were encountered:

1. **Docker Installation**: Although Docker was installed on all instances, there were intermittent problems with Docker services either failing to start or being improperly installed.
2. **Network Configuration**: Connectivity issues between the master and worker nodes likely resulted from misconfigured network settings or firewall restrictions, which blocked the necessary communication ports for Kubernetes to function properly.
3. **Connection Failures**: The worker nodes encountered "connection refused" errors when attempting to connect to the Kubernetes API server on the master node. This could indicate problems with Kubernetes components or security groups.
4. **CrashLoopBackOff**: Several Kubernetes containers experienced continuous restarts, possibly due to improper configuration or resource limitations, causing them to fail during initialization.