

Lab Assignment 10

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

LO2. To deploy single and multiple container applications and manage application deployments with rollouts in Kubernetes.

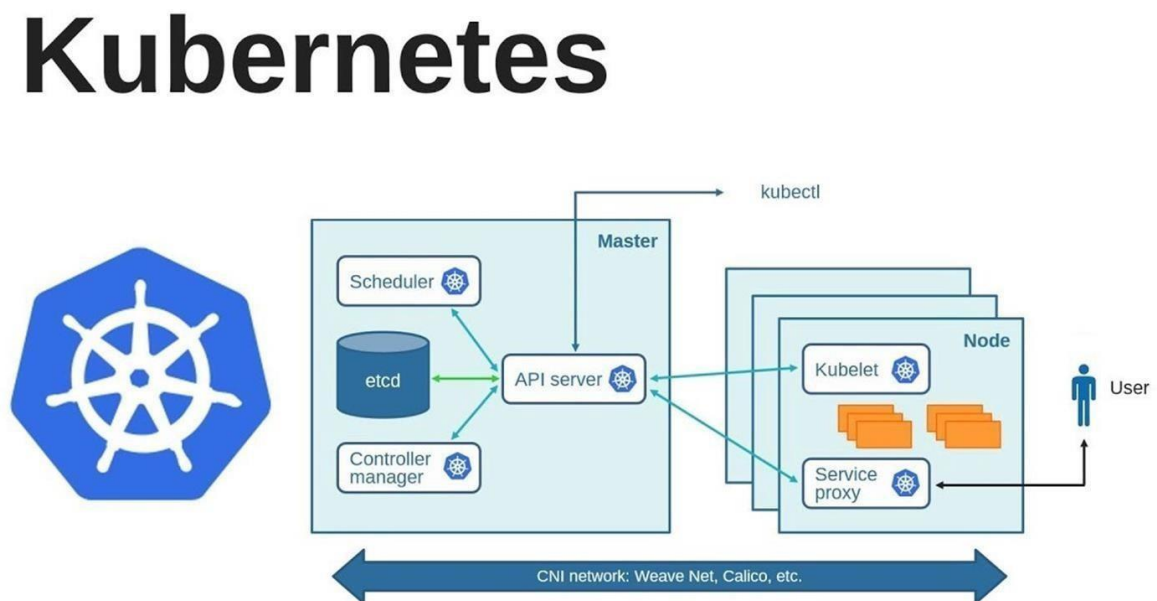
THEORY:

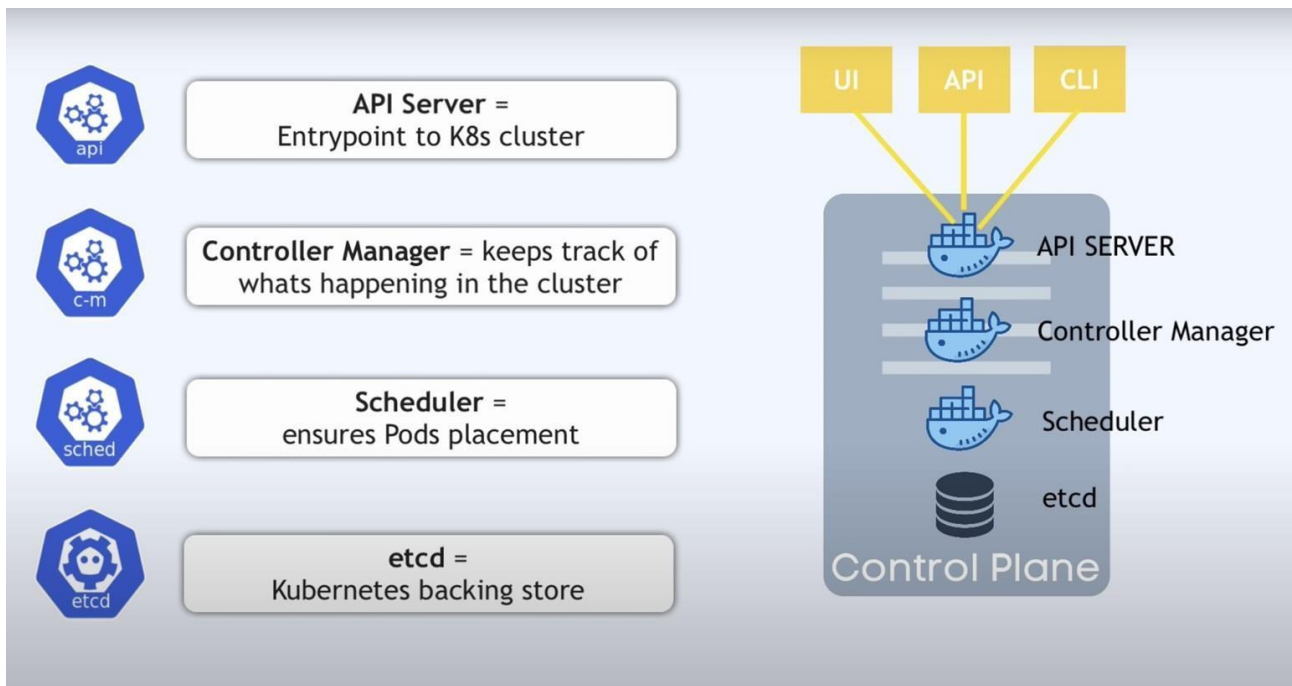
Kubernetes is an open-source container management tool that automates container deployment, scaling & load balancing.

It schedules, runs, and manages isolated containers that are running on virtual/physical/cloud machines.

All top cloud providers support Kubernetes. One popular name for Kubernetes is K8s.

ARCHITECTURE





Working with Kubernetes

- We create a Manifest (.yaml) file
- Apply those to cluster (to master) to bring it into the desired state.
- POD runs on a node, which is controlled by the master.

● Role of Master Node

- Kubernetes cluster contains containers running or Bare Metal / VM instances/cloud instances/ all mix.
- Kubernetes designates one or more of these as masters and all others as workers.
- The master is now going to run a set of K8s processes. These processes will ensure the smooth functioning of the cluster. These processes are called the 'Control Plane'.
- Can be Multi-Master for high availability.
- Master runs control plane to run cluster smoothly.

● Components of Control Plane

■ Kube-api-server → (For all communications)

- This api-server interacts directly with the user (i.e we apply .yaml or .json manifest to kubeapi-server)
- This kube-api-server is meant to scale automatically as per load.
- Kube-api-server is the front end of the control plane.

■ **etcd** • Stores metadata and status of the

cluster.

- etcd is a consistent and high-available store (key-value-store)
- Source of truth for cluster state (info about the state of the cluster)

→ **etcd has the following features**

1. Fully Replicated → The entire state is available on every node in the cluster.
2. Secure → Implements automatic TLS with optional client-certificate authentication.
3. Fast → Benchmarked at 10,000 writes per second.

■ **Kube-scheduler (action)**

- When users request the creation & management of Pods, Kube-scheduler is going to take action on these requests. • Handles POD creation and Management.
- Kube-scheduler match/assign any node to create and run pods.
- A scheduler watches for newly created pods that have no node assigned. For every pod that the scheduler discovers, the scheduler becomes responsible for finding the best node for that pod to run.
- The scheduler gets the information for hardware configuration from configuration files and schedules the Pods on nodes accordingly.

■ **Controller-Manager**

- Make sure the actual state of the cluster matches the desired state.

→ Two possible choices for controller manager— 1. If K8s is on the cloud, then it will be a cloud controller manager.

2. If K8s is on non-cloud, then it will be kube-controller-manager.

Components on the master that runs the controller

Node Controller → For checking the cloud provider to determine if a node has been detected in the cloud after it stops responding.

Route-Controller → Responsible for setting up a network, and routes on your cloud.

Service-Controller → Responsible for load Balancers on your cloud against services of type Load Balancer.

Volume-Controller → For creating, attaching, and mounting volumes and interacting with the cloud provider to orchestrate volume.

■ Nodes (Kubelet and Container Engine)

- Node is going to run 3 important pieces of software/process.

Kubelet

- The agent running on the node.
- Listens to Kubernetes master (eg- Pod creation request).
- Use port 10255.
- Send success/Fail reports to master. **Container Engine**
- Works with kubelet
- Pulling images
- Start/Stop Containers
- Exposing containers on ports specified in the manifest.

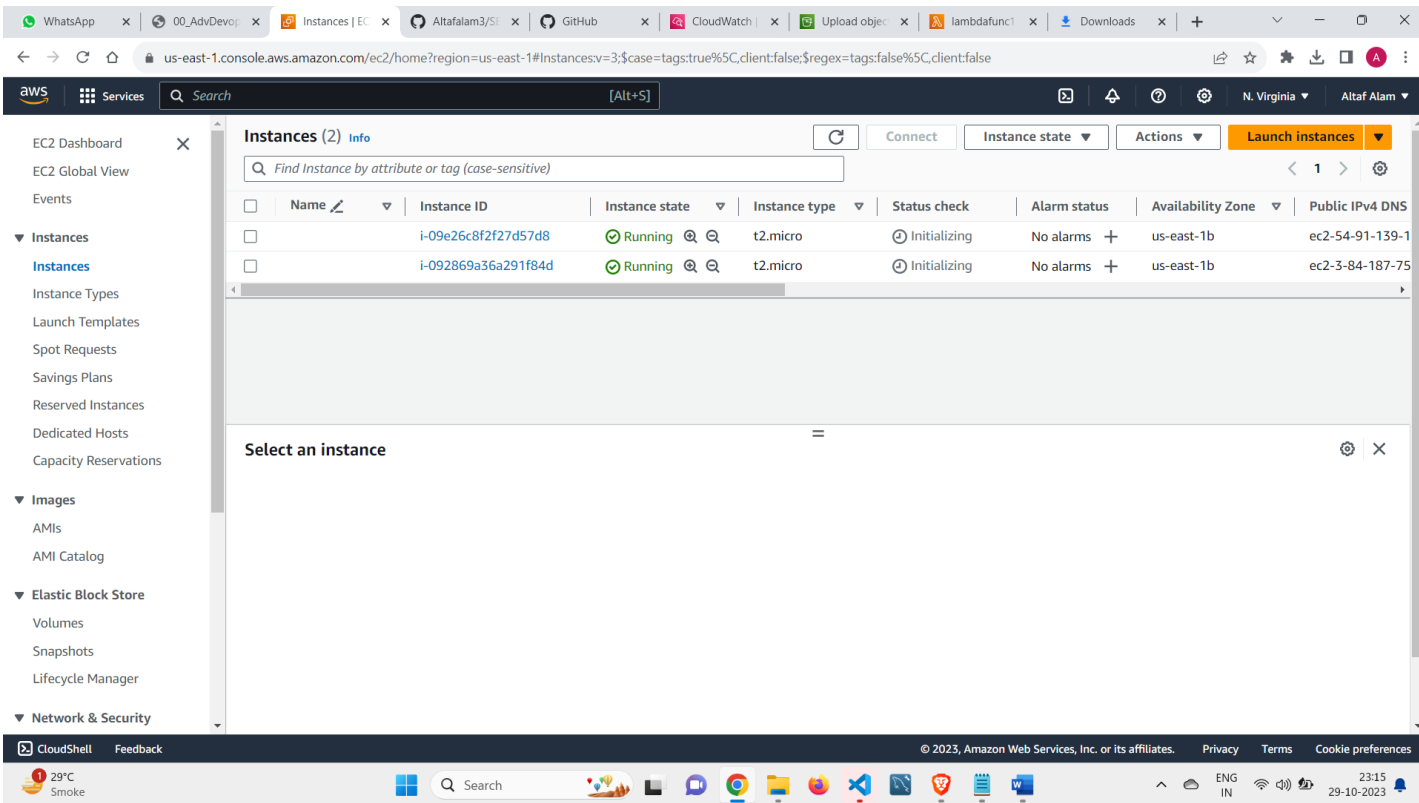
Kube-Proxy

- Assign IP to each pod.
- It is required to assign IP addresses to Pods (dynamic)
- Kube-proxy runs on each node & this makes sure that each pod will get its unique IP Address.
- These 3 components collectively consist of 'node'.

INSTALLATION:

T11 ALTAF ALAM 02

Launch 2 EC2 instance with all access



The screenshot displays the AWS Management Console for the 'us-east-1' region. The left sidebar shows the navigation menu with categories like EC2 Dashboard, Instances, Images, Elastic Block Store, and Network & Security. The main content area is titled 'Instances (2)' and shows a table of two running EC2 instances. Below the table, there is a 'Select an instance' section. The bottom of the screen shows the Windows taskbar with various application icons and the system clock indicating 23:15 on 29-10-2023.

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
	i-09e26c8f2f27d57d8	Running	t2.micro	Initializing	No alarms	us-east-1b	ec2-54-91-139-1
	i-092869a36a291f84d	Running	t2.micro	Initializing	No alarms	us-east-1b	ec2-3-84-187-75

See the security grp launch wizard-5 of both instance then

Scroll down security grp and see that security grp id

Edit inbound rules delete all and add “ALL TRAFFIC” , source “anywhere v4”

T11 ALTAF ALAM 02

Edit inbound rules [Info](#)

Inbound rules control the incoming traffic that's allowed to reach the instance.

Security group rule ID	Type Info	Protocol Info	Port range Info	Source Info	Description - optional Info
sgr-0af1d356a442e245d	All traffic	All	All	Custom	

[Add rule](#)

Rules with source of 0.0.0.0/0 or ::/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.

[Cancel](#) [Preview changes](#) [Save rules](#)

Name change

Instances (4) [Info](#)

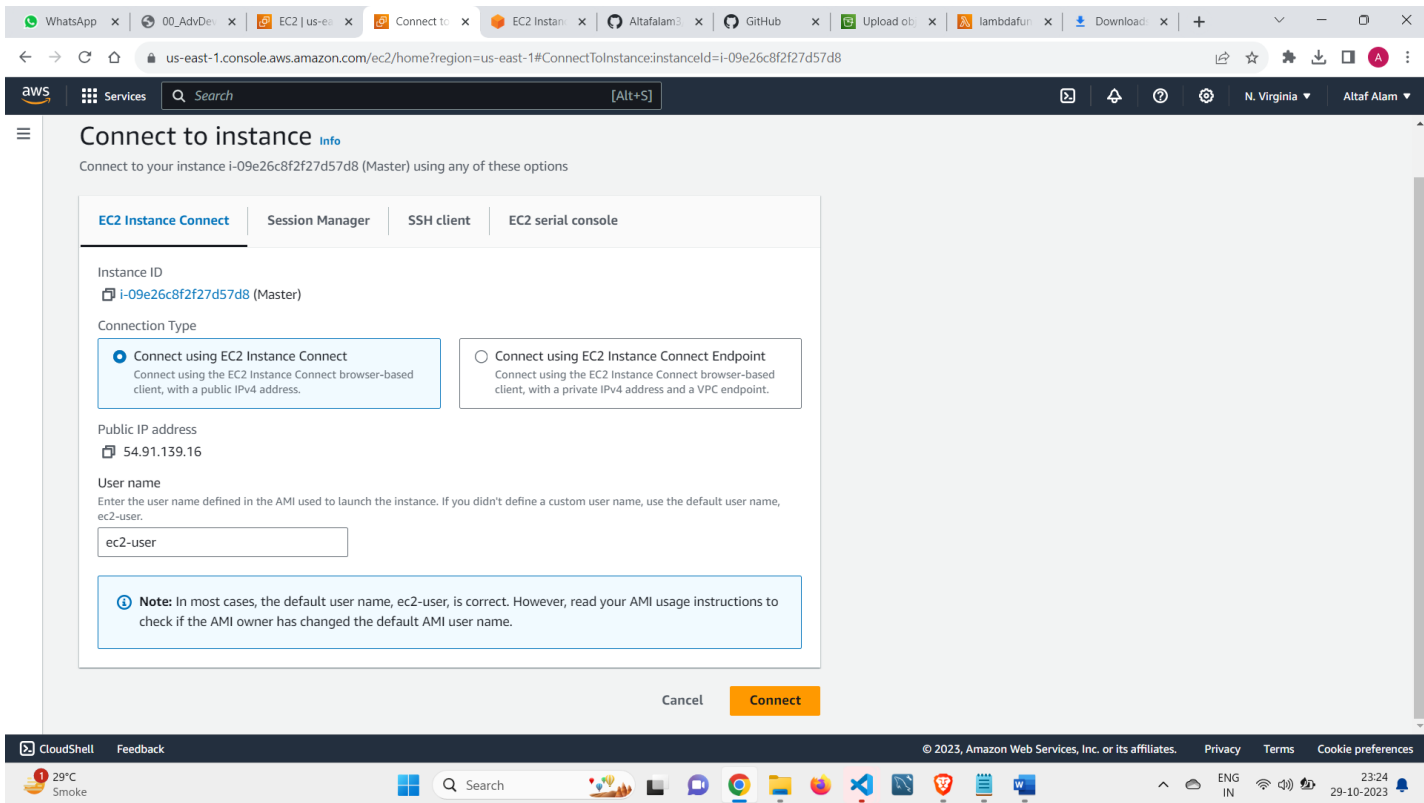
[Find Instance by attribute or tag \(case-sensitive\)](#)

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
Master1	i-00fc716d0f399724c	Running	t2.micro	Initializing	No alarms	us-east-1b	ec2-18-205-19-2
Worker-node1	i-0cf7b956ad06879b6	Running	t2.micro	Initializing	No alarms	us-east-1b	ec2-3-84-135-17

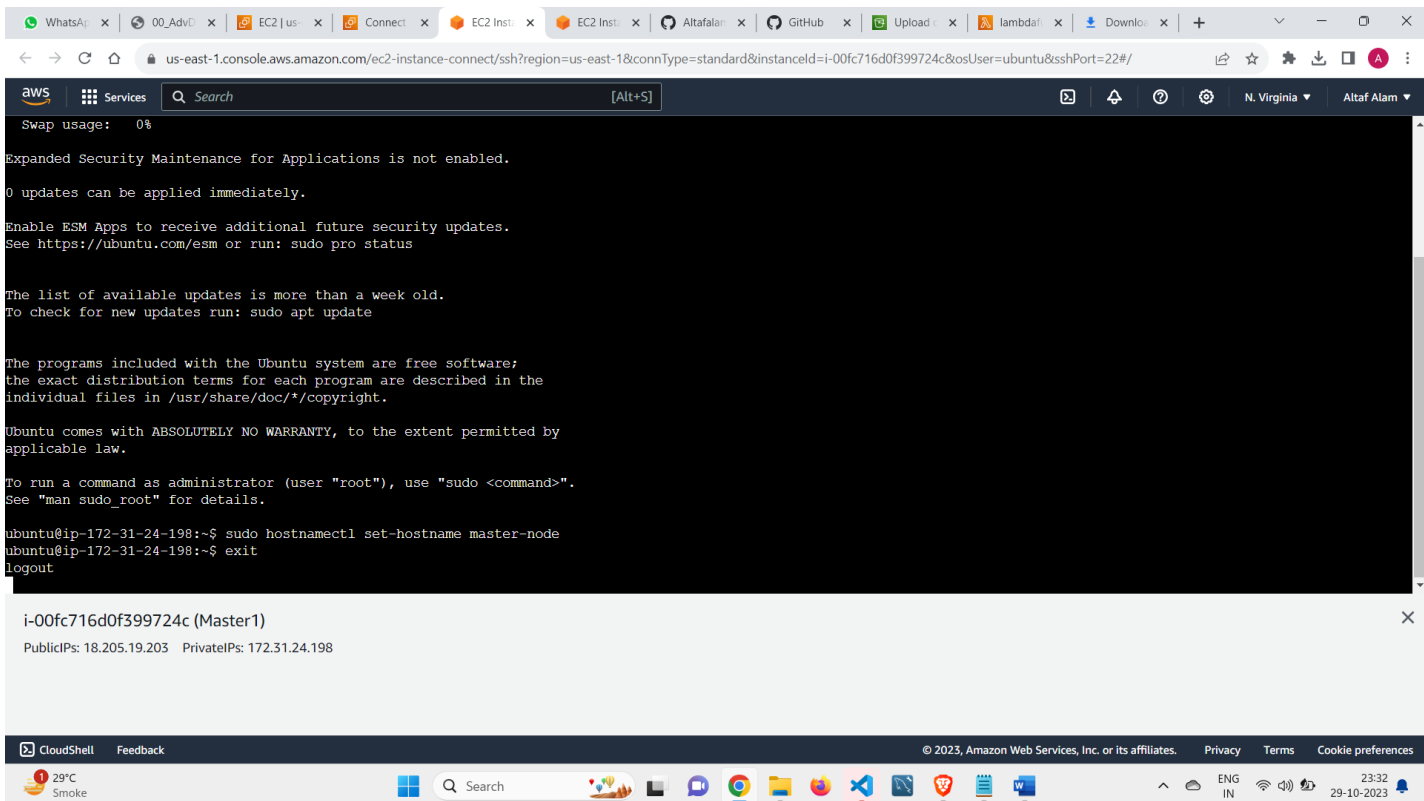
Select an instance

T11 ALTAF ALAM 02

Connect to master instance

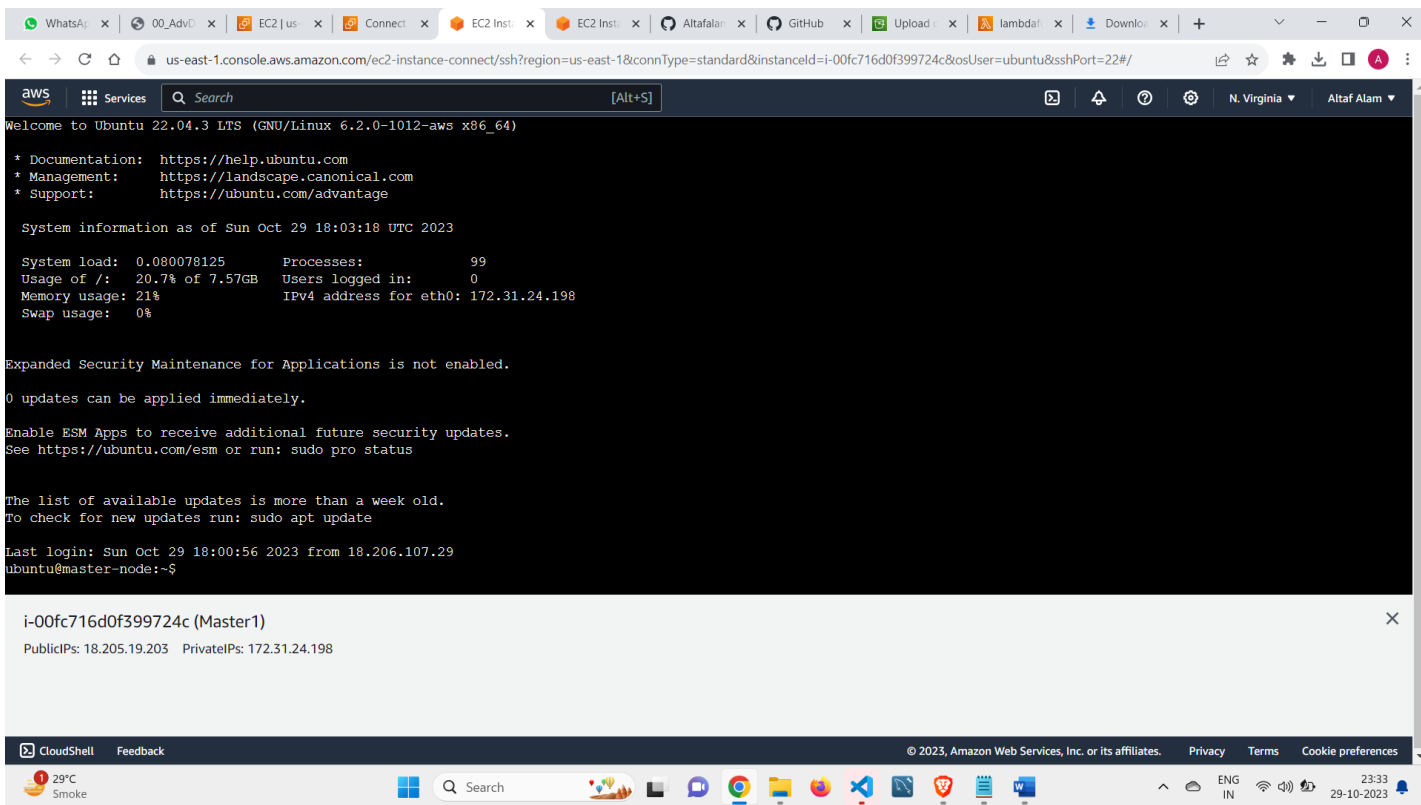


Here open the terminal and sethostname for both master and worker node



T11 ALTAF ALAM 02

Refresh it



The screenshot shows a web browser with multiple tabs open, including 'WhatsApp', '00_Adv...', 'EC2 | us...', 'Connect', 'EC2 Inst...', 'EC2 Inst...', 'Altafalan', 'GitHub', 'Upload', 'lambda', and 'Downlo...'. The active tab is 'us-east-1.console.aws.amazon.com/ec2-instance-connect/ssh?region=us-east-1&connType=standard&instancetype=i-00fc716d0f399724c&osUser=ubuntu&sshPort=22#/'. The terminal window displays the following text:

```
Welcome to Ubuntu 22.04.3 LTS (GNU/Linux 6.2.0-1012-aws x86_64)

* Documentation:  https://help.ubuntu.com
* Management:    https://landscape.canonical.com
* Support:        https://ubuntu.com/advantage

System information as of Sun Oct 29 18:03:18 UTC 2023

System load:  0.080078125      Processes:           99
Usage of /:   20.7% of 7.57GB   Users logged in:    0
Memory usage: 21%             IPv4 address for eth0: 172.31.24.198
Swap usage:   0%

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

Last login: Sun Oct 29 18:00:56 2023 from 18.206.107.29
ubuntu@master-node:~$
```

Below the terminal output, a box displays the instance ID 'i-00fc716d0f399724c (Master1)' and IP addresses: 'PublicIPs: 18.205.19.203' and 'PrivateIPs: 172.31.24.198'.

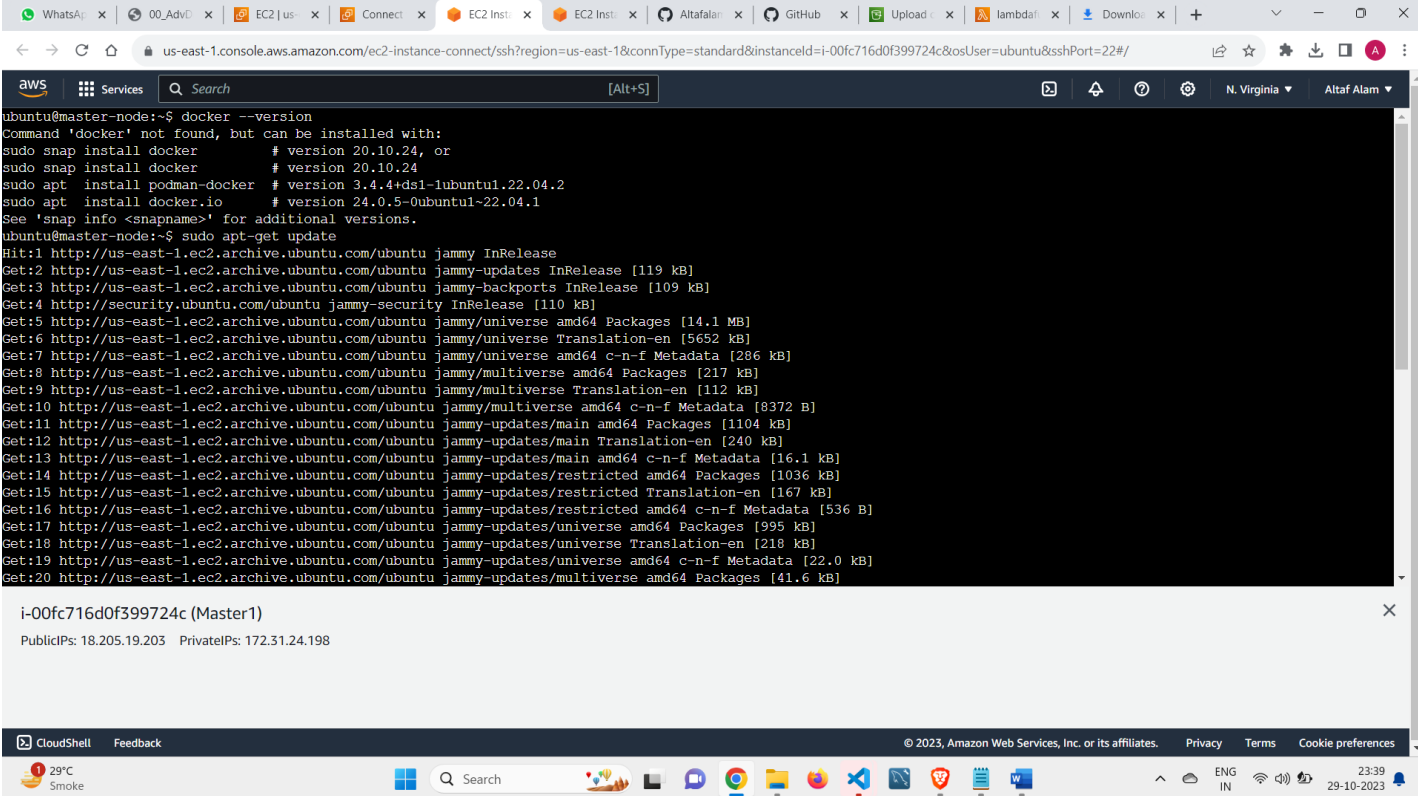
The bottom of the screenshot shows the Windows taskbar with a temperature display of 29°C, a search bar, and various application icons. The system clock shows 23:33 on 29-10-2023.

Similarly do for worker1

T11 ALTAF ALAM 02

Come back to master-node execute and same on worker

Install docker



```
ubuntu@master-node:~$ docker --version
Command 'docker' not found, but can be installed with:
sudo snap install docker          # version 20.10.24, or
sudo snap install docker          # version 20.10.24
sudo apt install podman-docker     # version 3.4.4+ds1-1ubuntu1.22.04.2
sudo apt install docker.io        # version 24.0.5-0ubuntu1-22.04.1
See 'snap info <snapname>' for additional versions.
ubuntu@master-node:~$ sudo apt-get update
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates InRelease [119 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-backports InRelease [109 kB]
Get:4 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/universe amd64 Packages [14.1 MB]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/universe Translation-en [5652 kB]
Get:7 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/universe amd64 c-n-f Metadata [286 kB]
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/multiverse amd64 Packages [217 kB]
Get:9 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/multiverse Translation-en [112 kB]
Get:10 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/multiverse amd64 c-n-f Metadata [8372 B]
Get:11 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 Packages [1104 kB]
Get:12 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main Translation-en [240 kB]
Get:13 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 c-n-f Metadata [16.1 kB]
Get:14 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/restricted amd64 Packages [1036 kB]
Get:15 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/restricted Translation-en [167 kB]
Get:16 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/restricted amd64 c-n-f Metadata [536 B]
Get:17 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 Packages [995 kB]
Get:18 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/universe Translation-en [218 kB]
Get:19 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 c-n-f Metadata [22.0 kB]
Get:20 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/multiverse amd64 Packages [41.6 kB]
```

i-00fc716d0f399724c (Master1)
PublicIPs: 18.205.19.203 PrivateIPs: 172.31.24.198

sudo apt-get install docker.io

T11 ALTAF ALAM 02

```
Get:32 http://security.ubuntu.com/ubuntu jammy-security/main Translation-en [180 kB]
Get:33 http://security.ubuntu.com/ubuntu jammy-security/main amd64 c-n-f Metadata [11.4 kB]
Get:34 http://security.ubuntu.com/ubuntu jammy-security/restricted amd64 Packages [1016 kB]
Get:35 http://security.ubuntu.com/ubuntu jammy-security/restricted Translation-en [164 kB]
Get:36 http://security.ubuntu.com/ubuntu jammy-security/restricted amd64 c-n-f Metadata [536 B]
Get:37 http://security.ubuntu.com/ubuntu jammy-security/universe amd64 Packages [793 kB]
Get:38 http://security.ubuntu.com/ubuntu jammy-security/universe Translation-en [145 kB]
Get:39 http://security.ubuntu.com/ubuntu jammy-security/universe amd64 c-n-f Metadata [16.7 kB]
Get:40 http://security.ubuntu.com/ubuntu jammy-security/multiverse amd64 Packages [36.5 kB]
Get:41 http://security.ubuntu.com/ubuntu jammy-security/multiverse Translation-en [7060 B]
Get:42 http://security.ubuntu.com/ubuntu jammy-security/multiverse amd64 c-n-f Metadata [260 B]
Fetched 27.9 MB in 5s (5515 kB/s)
Reading package lists... Done
ubuntu@master-node:~$ ^C
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 debotstrap 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 41 not upgraded.
Need to get 69.7 MB of archives.
After this operation, 267 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
```

i-00fc716d0f399724c (Master1)

PublicIPs: 18.205.19.203 PrivateIPs: 172.31.24.198

```
No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@master-node:~$ docker --version
Docker version 24.0.5, build 24.0.5-0ubuntu1-22.04.1
ubuntu@master-node:~$ sudo systemctl enable docker
ubuntu@master-node:~$ sudo systemctl status docker
● docker.service - Docker Application Container Engine
   Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: enabled)
   Active: active (running) since Sun 2023-10-29 18:12:09 UTC; 1min 54s ago
   TriggeredBy: ● docker.socket
     Docs: https://docs.docker.com
    Main PID: 2950 (dockerd)
      Tasks: 8
     Memory: 34.4M
        CPU: 302ms
    CGroup: /system.slice/docker.service
            └─2950 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock

Oct 29 18:12:08 master-node systemd[1]: Starting Docker Application Container Engine...
Oct 29 18:12:08 master-node dockerd[2950]: time="2023-10-29T18:12:08.513848877Z" level=info msg="Starting up"
Oct 29 18:12:08 master-node dockerd[2950]: time="2023-10-29T18:12:08.515869816Z" level=info msg="detected 127.0.0.53 nameserver, assuming systemd-resolved, so using resolvconf"
Oct 29 18:12:08 master-node dockerd[2950]: time="2023-10-29T18:12:08.638382094Z" level=info msg="Loading containers: start."
Oct 29 18:12:08 master-node dockerd[2950]: time="2023-10-29T18:12:08.947998027Z" level=info msg="Loading containers: done."
Oct 29 18:12:09 master-node dockerd[2950]: time="2023-10-29T18:12:09.012935950Z" level=info msg="Docker daemon" commit="24.0.5-0ubuntu1-22.04.1" graphdriver="overlay2" version="24.0.5"
Oct 29 18:12:09 master-node dockerd[2950]: time="2023-10-29T18:12:09.013434361Z" level=info msg="Daemon has completed initialization"
Oct 29 18:12:09 master-node systemd[1]: Started Docker Application Container Engine.
lines 1-21/21 (END)
```

i-00fc716d0f399724c (Master1)

PublicIPs: 18.205.19.203 PrivateIPs: 172.31.24.198

T11 ALTAF ALAM 02

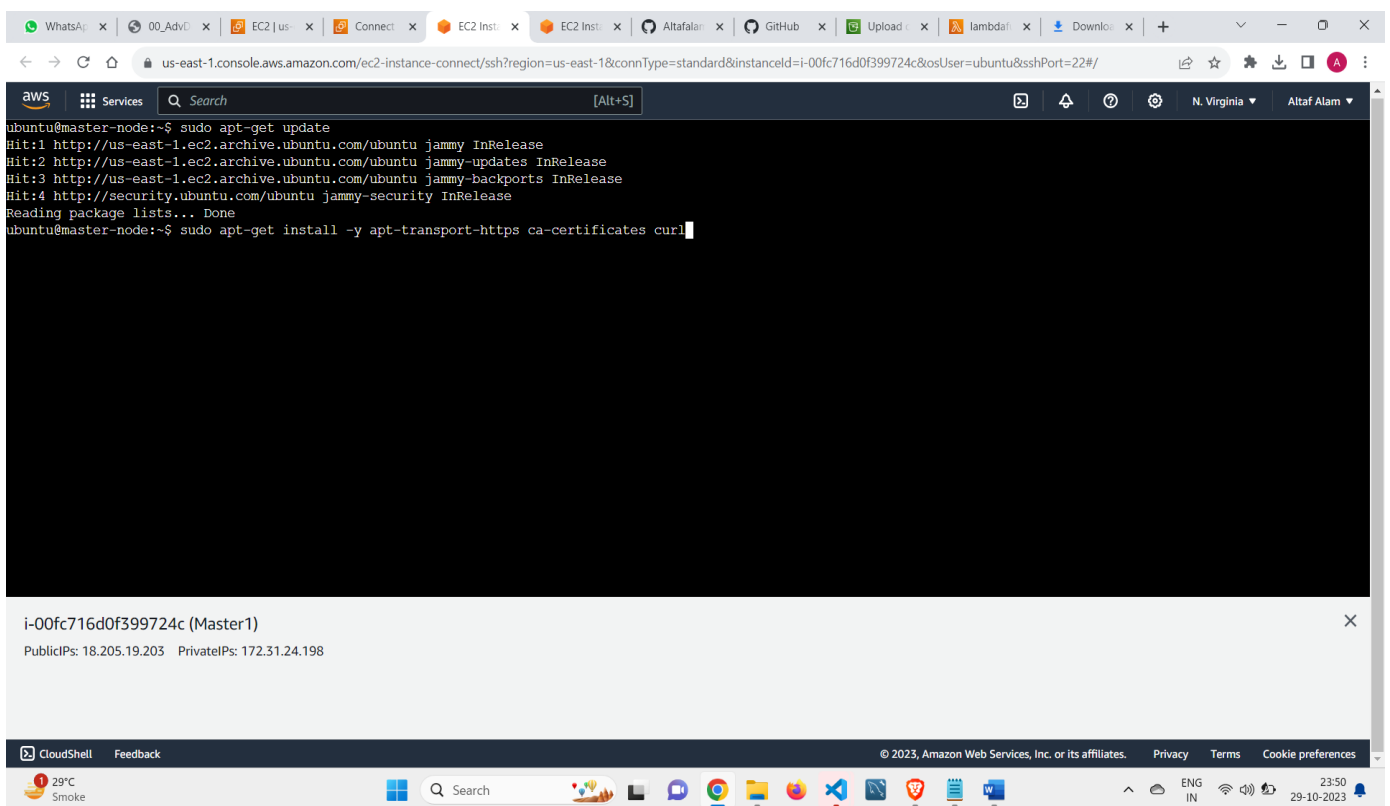
If above step failed then manually do start by this

```
--- sudo systemctl start docker
```

Now installing kubernetes

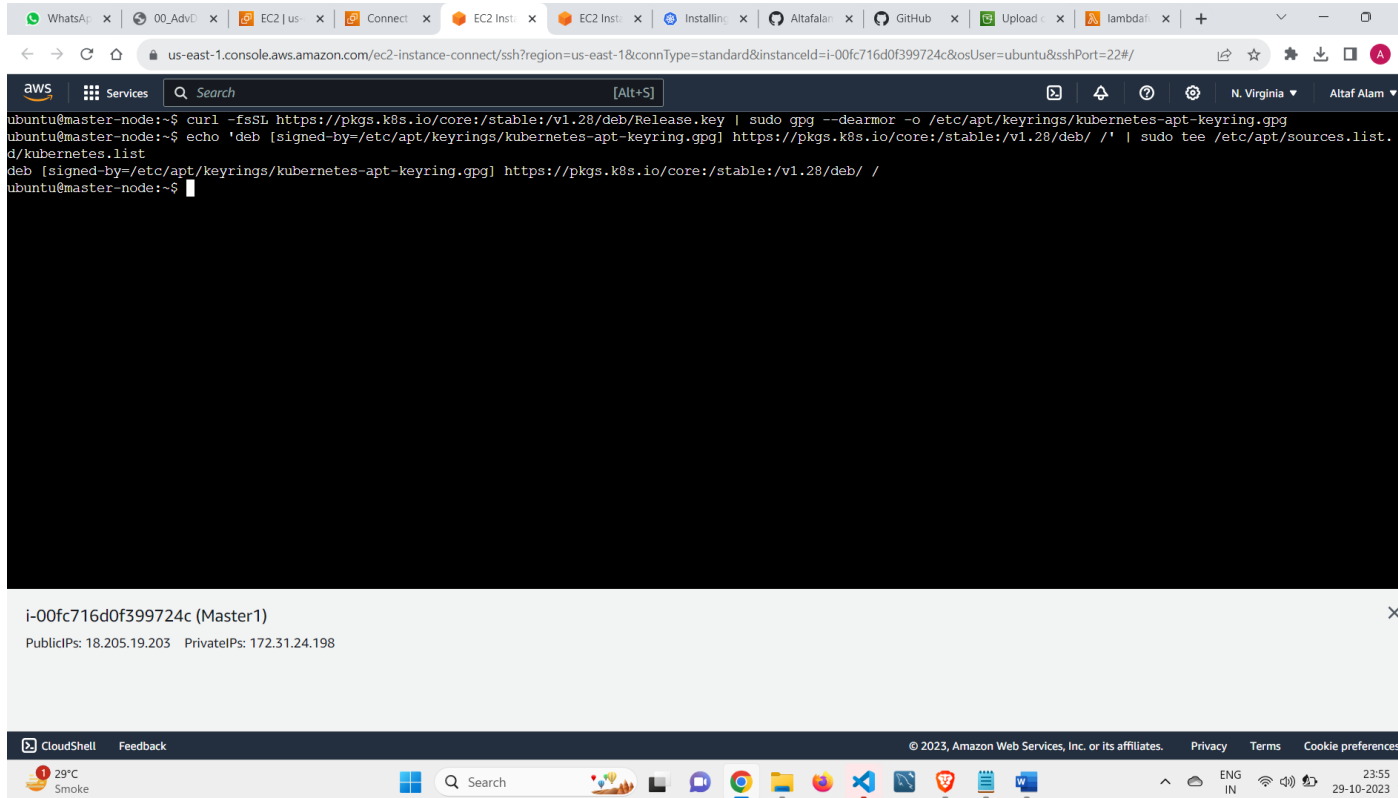
<https://kubernetes.io/docs/setup/production-environment/tools/kubeadm/install-kubeadm/>

```
sudo apt-get install -y apt-transport-https ca-certificates curl
```



T11 ALTAF ALAM 02

1. `curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.28/deb/Release.key | sudo gpg --dearmor -o /etc/apt/keyrings/kubernetes-apt-keyring.gpg`
2. `echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.28/deb/ /' | sudo tee /etc/apt/sources.list.d/kubernetes.list`



Run these commands in both

1. `sudo apt-get update`
2. `sudo apt-get install -y kubelet kubeadm kubectl`
3. `sudo apt-mark hold kubelet kubeadm kubectl`

T11 ALTAF ALAM 02

```
ubuntu@master-node:~$ sudo apt-get update
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu jammy-security InRelease
Hit:5 https://prod-cdn.packages.k8s.io/repositories/istio/kubernetes/core/stable/v1.28/deb InRelease
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 ebtables kubernetes-cni socat
The following NEW packages will be installed:
  conntrack cri-tools ebtables kubeadm kubectl kubelet kubernetes-cni socat
0 upgraded, 8 newly installed, 0 to remove and 39 not upgraded.
Need to get 87.6 MB of archives.
After this operation, 337 MB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/main amd64 conntrack amd64 1:1.4.6-2build2 [33.5 kB]
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/main amd64 ebtables amd64 2.0.11-4build2 [84.9 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/main amd64 socat amd64 1.7.4.1-3ubuntu4 [349 kB]
Get:4 https://prod-cdn.packages.k8s.io/repositories/istio/kubernetes/core/stable/v1.28/deb cri-tools 1.28.0-1.1 [19.6 MB]
Get:5 https://prod-cdn.packages.k8s.io/repositories/istio/kubernetes/core/stable/v1.28/deb kubernetes-cni 1.2.0-2.1 [27.6 MB]
Get:6 https://prod-cdn.packages.k8s.io/repositories/istio/kubernetes/core/stable/v1.28/deb kubelet 1.28.3-1.1 [19.5 MB]
Get:7 https://prod-cdn.packages.k8s.io/repositories/istio/kubernetes/core/stable/v1.28/deb kubectl 1.28.3-1.1 [10.3 MB]
Get:8 https://prod-cdn.packages.k8s.io/repositories/istio/kubernetes/core/stable/v1.28/deb kubeadm 1.28.3-1.1 [10.1 MB]
Fetched 87.6 MB in 1s (60.6 MB/s)
Selecting previously unselected package conntrack.
```

i-00fc716d0f399724c (Master1)
PublicIPs: 18.205.19.203 PrivateIPs: 172.31.24.198

```
Preparing to unpack .../7-kubeadm 1.28.3-1.1_amd64.deb ...
Unpacking kubeadm (1.28.3-1.1) ...
Setting up conntrack (1:1.4.6-2build2) ...
Setting up kubectl (1.28.3-1.1) ...
Setting up ebtables (2.0.11-4build2) ...
Setting up socat (1.7.4.1-3ubuntu4) ...
Setting up cri-tools (1.28.0-1.1) ...
Setting up kubernetes-cni (1.2.0-2.1) ...
Setting up kubelet (1.28.3-1.1) ...
Setting up kubeadm (1.28.3-1.1) ...
Processing triggers for man-db (2.10.2-1) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
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:~$
```

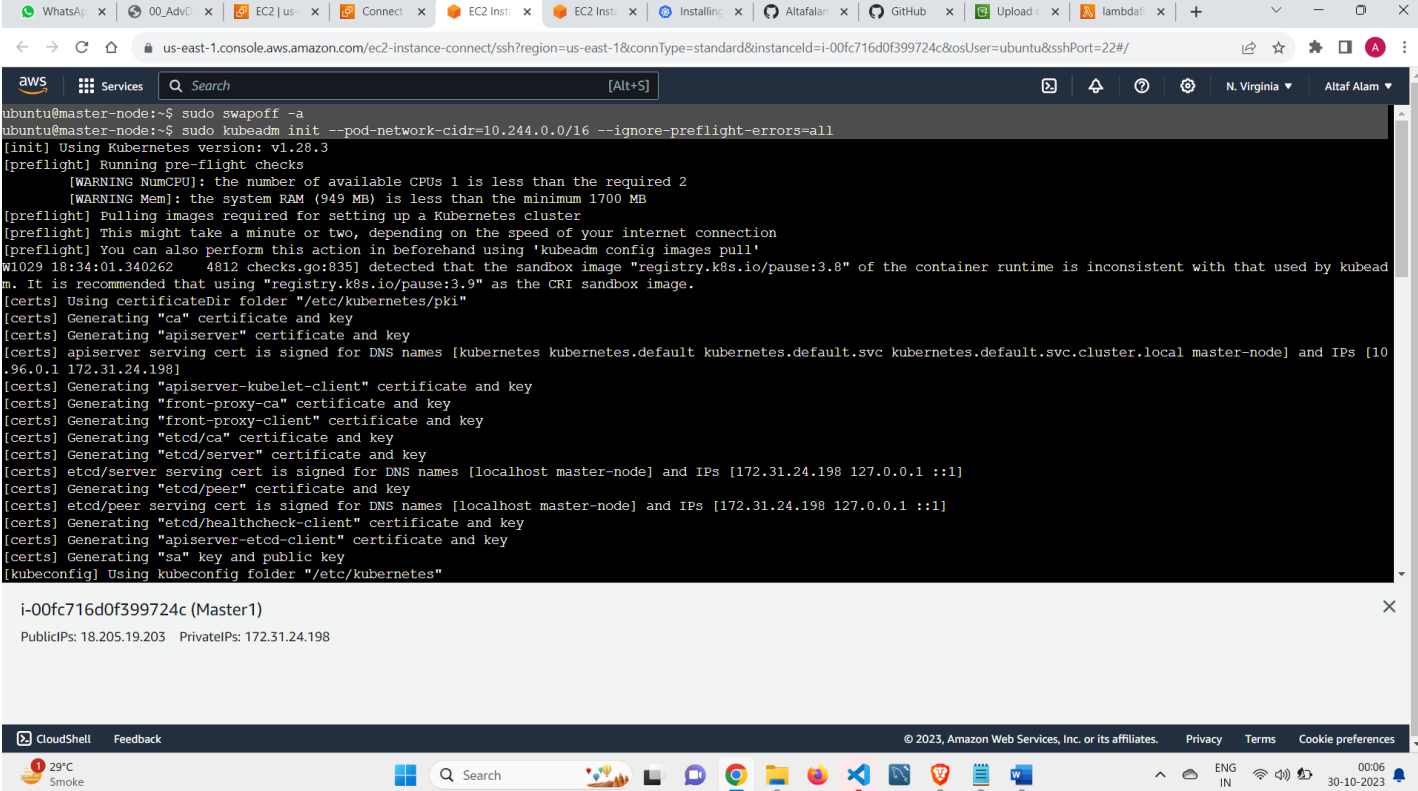
i-00fc716d0f399724c (Master1)
PublicIPs: 18.205.19.203 PrivateIPs: 172.31.24.198

T11 ALTAF ALAM 02

Installation done , now

`sudo swapoff -a`

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



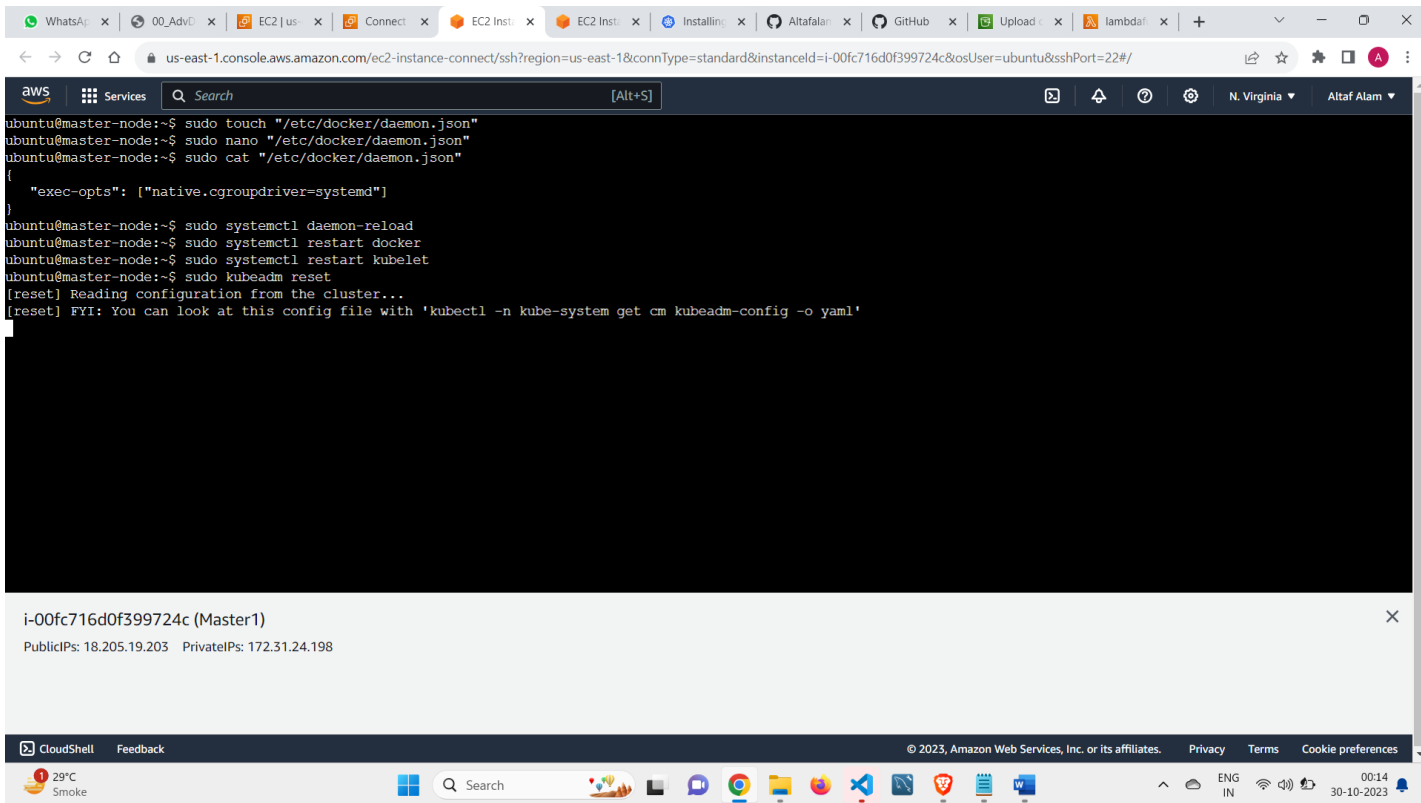
```
ubuntu@master-node:~$ sudo swapoff -a
ubuntu@master-node:~$ sudo kubeadm init --pod-network-cidr=10.244.0.0/16 --ignore-preflight-errors=all
[init] Using Kubernetes version: v1.28.3
[preflight] Running pre-flight checks
        [WARNING NumCPU]: the number of available CPUs 1 is less than the required 2
        [WARNING Mem]: the system RAM (949 MB) is less than the minimum 1700 MB
[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 in beforehand using 'kubeadm config images pull'
W1029 18:34:01.340262 4812 checks.go:835] 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 that using "registry.k8s.io/pause:3.9" 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 [kubernetes kubernetes.default kubernetes.default.svc kubernetes.default.svc.cluster.local master-node] and IPs [10.96.0.1 172.31.24.198]
[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 [localhost master-node] and IPs [172.31.24.198 127.0.0.1 ::1]
[certs] Generating "etcd/peer" certificate and key
[certs] etcd/peer serving cert is signed for DNS names [localhost master-node] and IPs [172.31.24.198 127.0.0.1 ::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"
```

i-00fc716d0f399724c (Master1)
PublicIPs: 18.205.19.203 PrivateIPs: 172.31.24.198

NOW, execute this line by line:

```
$ sudo touch "/etc/docker/daemon.json"
$ sudo nano "/etc/docker/daemon.json"
$ c
{
  "exec-opts": ["native.cgroupdriver=systemd"]
}
$ sudo systemctl daemon-reload
$ sudo systemctl restart docker
$ sudo systemctl restart kubelet
$ sudo kubeadm reset
```

T11 ALTAF ALAM 02



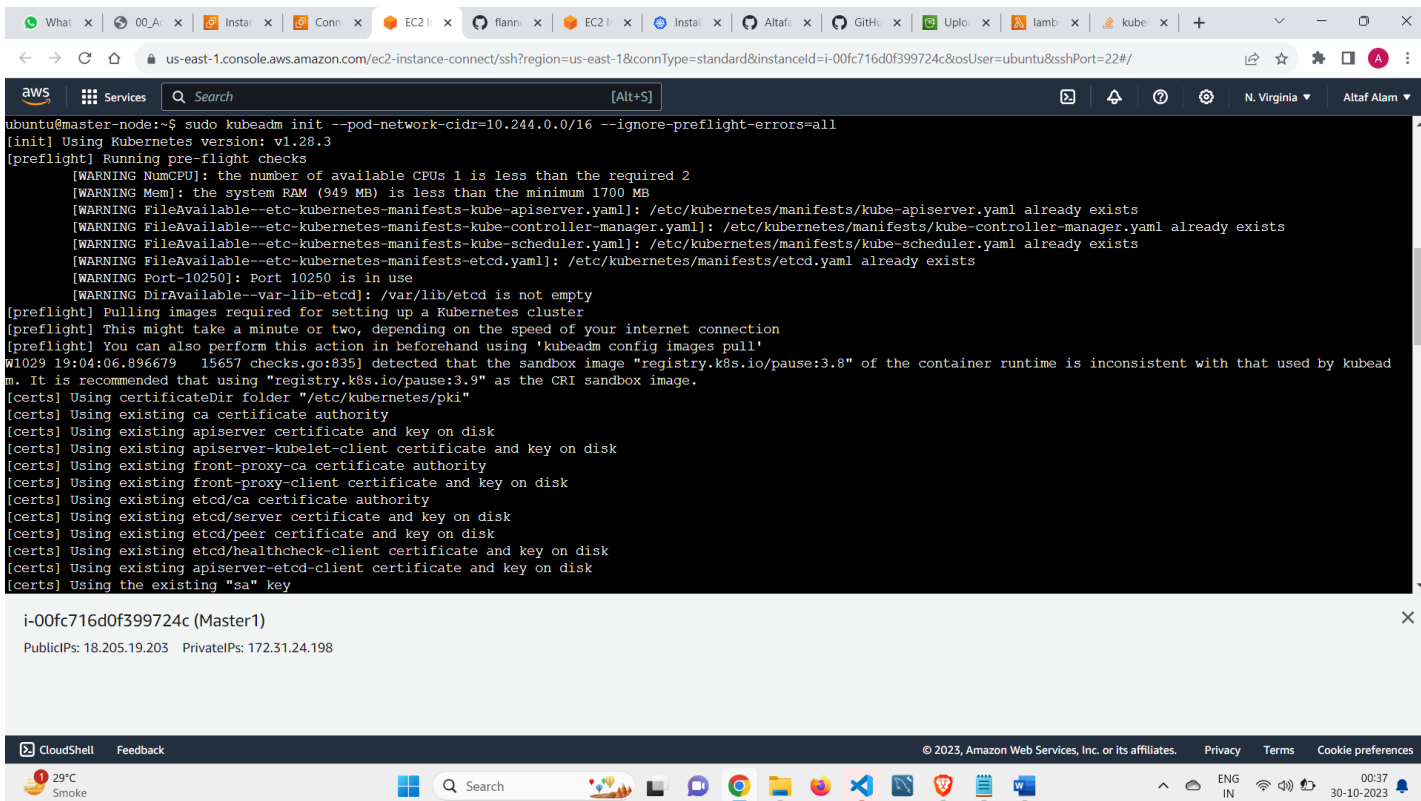
The screenshot shows the AWS CloudShell interface with a terminal window. The terminal output shows the following commands and their results:

```
ubuntu@master-node:~$ sudo touch "/etc/docker/daemon.json"
ubuntu@master-node:~$ sudo nano "/etc/docker/daemon.json"
ubuntu@master-node:~$ sudo cat "/etc/docker/daemon.json"
{
  "exec-opts": ["native.cgroupdriver=systemd"]
}
ubuntu@master-node:~$ sudo systemctl daemon-reload
ubuntu@master-node:~$ sudo systemctl restart docker
ubuntu@master-node:~$ sudo systemctl restart kubelet
ubuntu@master-node:~$ sudo kubeadm reset
[reset] Reading configuration from the cluster...
[reset] FYI: You can look at this config file with 'kubectl -n kube-system get cm kubeadm-config -o yaml'
```

Below the terminal window, the instance details are shown: i-00fc716d0f399724c (Master1), PublicIPs: 18.205.19.203, PrivateIPs: 172.31.24.198.

Now, again run the command:

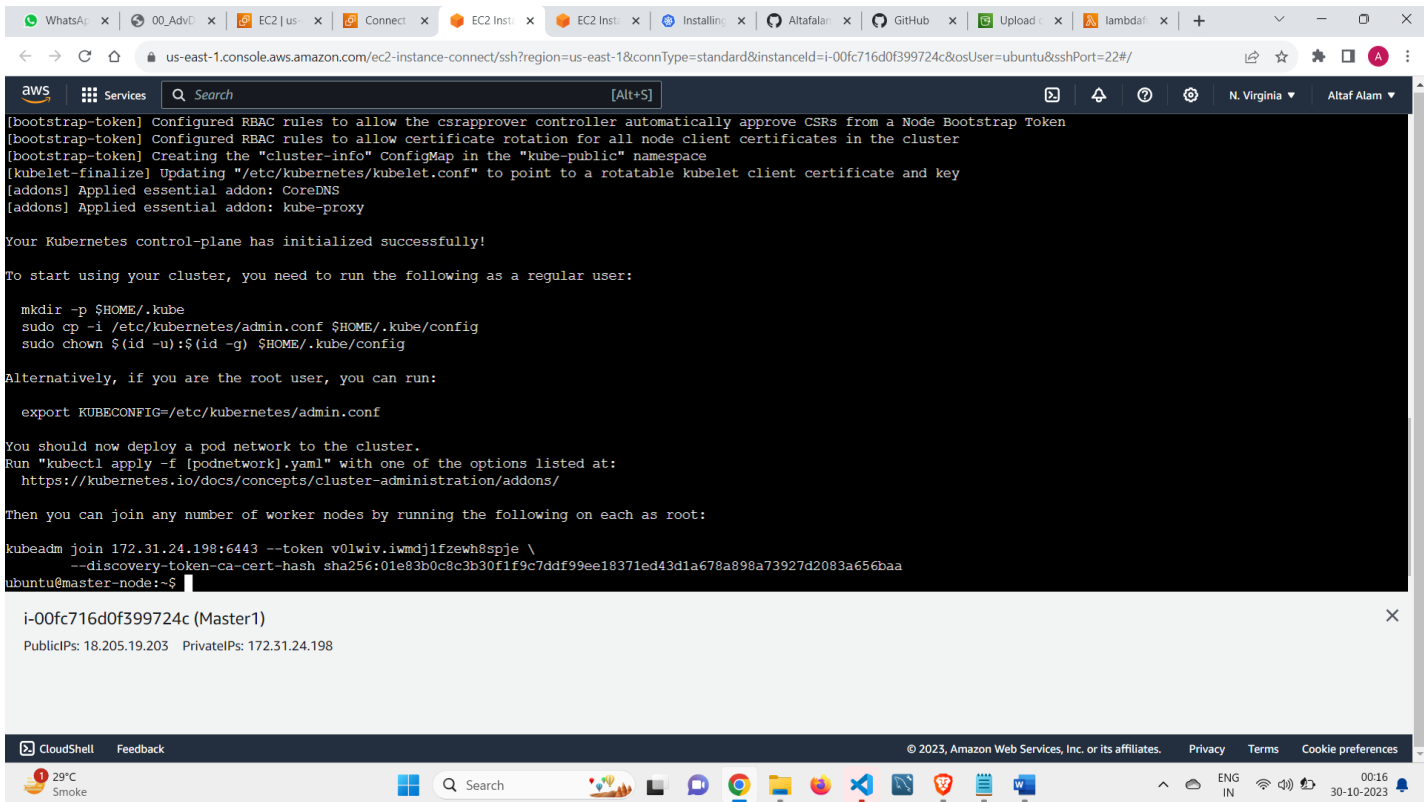
```
sudo kubeadm init --pod-network-cidr=10.244.0.0/16 --ignore-preflight-errors=all
```



The screenshot shows the AWS CloudShell interface with a terminal window. The terminal output shows the following commands and their results:

```
ubuntu@master-node:~$ sudo kubeadm init --pod-network-cidr=10.244.0.0/16 --ignore-preflight-errors=all
[init] Using Kubernetes version: v1.28.3
[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 FileAvailable--etc-kubernetes-manifests-kube-apiserver.yaml]: /etc/kubernetes/manifests/kube-apiserver.yaml already exists
[WARNING FileAvailable--etc-kubernetes-manifests-kube-controller-manager.yaml]: /etc/kubernetes/manifests/kube-controller-manager.yaml already exists
[WARNING FileAvailable--etc-kubernetes-manifests-kube-scheduler.yaml]: /etc/kubernetes/manifests/kube-scheduler.yaml already exists
[WARNING FileAvailable--etc-kubernetes-manifests-etcd.yaml]: /etc/kubernetes/manifests/etcd.yaml already exists
[WARNING Port-10250]: Port 10250 is in use
[WARNING DirAvailable--var-lib-etcd]: /var/lib/etcd is not empty
[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 in beforehand using 'kubeadm config images pull'
W1029 19:04:06.896679 15657 checks.go:835] 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 that using "registry.k8s.io/pause:3.9" as the CRI sandbox image.
[certs] Using certificateDir folder "/etc/kubernetes/pki"
[certs] Using existing ca certificate authority
[certs] Using existing apiserver certificate and key on disk
[certs] Using existing apiserver-kubelet-client certificate and key on disk
[certs] Using existing front-proxy-ca certificate authority
[certs] Using existing front-proxy-client certificate and key on disk
[certs] Using existing etcd/ca certificate authority
[certs] Using existing etcd/server certificate and key on disk
[certs] Using existing etcd/peer certificate and key on disk
[certs] Using existing etcd/healthcheck-client certificate and key on disk
[certs] Using existing apiserver-etcd-client certificate and key on disk
[certs] Using the existing "sa" key
```

Below the terminal window, the instance details are shown: i-00fc716d0f399724c (Master1), PublicIPs: 18.205.19.203, PrivateIPs: 172.31.24.198.



```
[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:

kubeadm join 172.31.24.198:6443 --token v0lwiv.iwmdj1fzewish8spje \
--discovery-token-ca-cert-hash sha256:01e83b0c8c3b30f1f9c7ddf99ee18371ed43d1a678a898a73927d2083a656baa
ubuntu@master-node:~$
```

i-00fc716d0f399724c (Master1)
PublicIPs: 18.205.19.203 PrivateIPs: 172.31.24.198

Copy the last join command and keep,

Now , run these commands which in above **SS ONLY IN MASTER NODE**

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

ALSO RUN this in **MASTER NODE**,

kubectl apply -f <https://raw.githubusercontent.com/flannel-io/flannel/master/Documentation/kube-flannel.yml>


```

aws
Services
Search [Alt+S]
N. Virginia
Altaf Alam

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 172.31.24.198:6443 --token g542vs.kirh0ca3lw916i6x \
--discovery-token-ca-cert-hash sha256:01e83b0c8c3b30f1f9c7ddf99ee18371ed43d1a678a898a73927d2083a656baa
ubuntu@master-node:~$ mkdir -p $HOME/.kube
ubuntu@master-node:~$ sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
cp: overwrite '/home/ubuntu/.kube/config'?
ubuntu@master-node:~$ sudo chown $(id -u):$(id -g) $HOME/.kube/config
ubuntu@master-node:~$ kubectl apply -f https://raw.githubusercontent.com/flannel-io/flannel/master/Documentation/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
ubuntu@master-node:~$

i-00fc716d0f399724c (Master1)
PublicIPs: 18.205.19.203 PrivateIPs: 172.31.24.198

CloudShell Feedback
© 2023, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences
29°C Smoke
Search
ENG IN
00:39 30-10-2023

```

(stop and restart around it above if not working)

AND EXECUTE in **MASTER NODE** (wait until all ticks checked):

kubectl get pods --all-namespaces

Once all ticks checked then go to worker node

For joining the worker node with masternode, execute only in **WORKER NODE**:

kubeadm join 172.31.24.198:6443 --token g542vs.kirh0ca3lw916i6x \

--discovery-token-ca-cert-hash

sha256:01e83b0c8c3b30f1f9c7ddf99ee18371ed43d1a678a898a73927d2083a656baa

Remove the "\", Also append "--ignore-preflight-errors=all" at the end of above command and do using "sudo".

'''

```
sudo kubeadm join 172.31.24.198:6443 --token g542vs.kirh0ca31w916i6x --discovery-token-ca-cert-hash sha256:01e83b0c8c3b30f1f9c7ddf99ee18371ed43d1a678a898a73927d2083a656baa --ignore-preflight-errors=all
```

'''

AFTER THIS DO IN MASTER NODE :

(wait for finish in this command to see worker)

```
kubectl get nodes
```

CONCLUSION:

Here we studied Kubernetes cluster architecture in detail. Also we installed Kubernetes in ubuntu machine and created a sample deployment.

