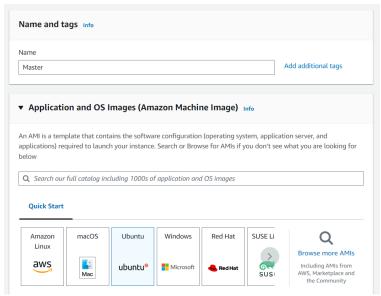
### **Experiment No 3**

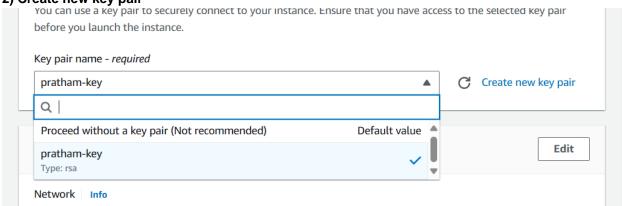
# Prathamesh Palve D15A 32 Batch B

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

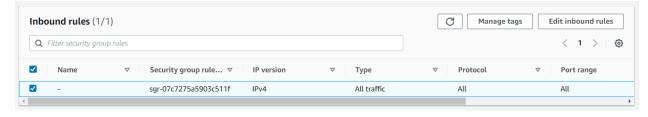
### 1) Launch 2 EC2 instance and select Ubuntu in AMI



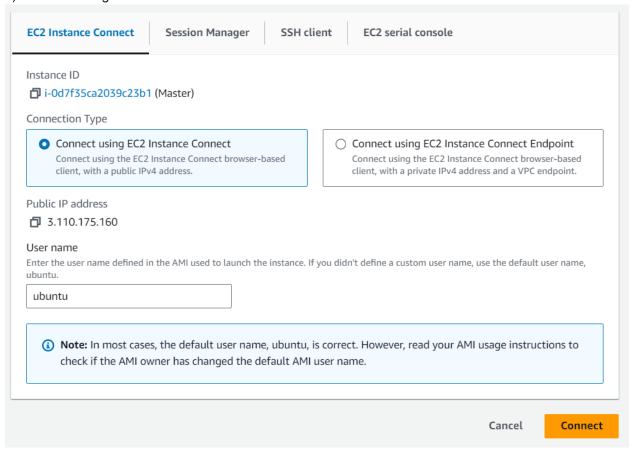
2) Create new key pair



- 3) In Security group select all checkbox and launch instance
- 4) Go to security group and edit inbound rules of both instance
- 5) Delete all the rules and add new rule with All traffic and Anywhere-IPv4



6) Now in running instances click on master instance and click on connect



- 8) Similarly connect the worker
- 9) Set hostname to master and worker respectively

ubuntu@ip-172-31-12-130:~\$ sudo hostnamectl set-hostname master ubuntu@ip-172-31-12-130:~\$ ubuntu@worker:~\$

10) Use command sudo apt-get update on both master and worker CLI sudo apt-get update on both

```
Get:24 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-backports/main Translation-en [10.5 kB]
Get:25 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-backports/main amd64 c-n-f Metadata [388 B]
Get:26 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-backports/restricted amd64 c-n-f Metadata [1
Get:27 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-backports/universe amd64 Packages [24.3 kB]
Get:28 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-backports/universe Translation-en [16.4 kB]
Get:29 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-backports/universe amd64 c-n-f Metadata [644
Get:30 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-backports/multiverse amd64 c-n-f Metadata [1
Get:31 http://security.ubuntu.com/ubuntu jammy-security/main amd64 Packages [765 kB]
Get:32 http://security.ubuntu.com/ubuntu jammy-security/main Translation-en [165 kB]
Get:33 http://security.ubuntu.com/ubuntu jammy-security/main amd64 c-n-f Metadata [11.3 kB]
Get:34 http://security.ubuntu.com/ubuntu jammy-security/restricted amd64 Packages [826 kB]
Get:35 http://security.ubuntu.com/ubuntu jammy-security/restricted Translation-en [133 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 [781 kB]
Get:38 http://security.ubuntu.com/ubuntu jammy-security/universe Translation-en [143 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 27.1 MB in 5s (5664 kB/s)
Reading package lists... Done
ubuntu@master:~$
```

### 11) Installing Docker on both CLI

sudo apt-get install docker.io on both

```
Done.

Created symlink /etc/systemd/system/multi-user.target.wants/docker.service -- /lib/systemd/system/docker.service.

Created symlink /etc/systemd/system/sockets.target.wants/docker.socket -- /lib/systemd/system/docker.socket.

Processing triggers for dbus (1.12.20-2ubuntu4.1) ...

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:-$ |
```

### 12) Enable Docker on both CLI and check its status

sudo systemctl enable docker

sudo systermctl status docker on both

### **DOCKER INSTALLED SUCCESSFULLY**

#### 13) Now for Installing Kubernetes (On both CLI)

```
ubuntu@master:~$ sudo apt-get update
Hit:1 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy InRelease
Hit:2 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:3 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu jammy-security InRelease
Reading package lists... Done
ubuntu@master:~$
```

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

```
Updating certificates in /etc/ssl/certs..
rehash: warning: skipping ca-certificates.crt,it does not contain exactly one certificate or CRL
19 added, 6 removed; done.
Setting up libcurl4:amd64 (7.81.0-lubuntu1.13) ...
Setting up curl (7.81.0-lubuntul.13) ...
Processing triggers for man-db (2.10.2-1) ...
Processing triggers for libc-bin (2.35-Oubuntu3.1) ...
Processing triggers for ca-certificates (20230311ubuntu0.22.04.1) ...
Updating certificates in /etc/ssl/certs...
0 added, 0 removed; done.
Running hooks in /etc/ca-certificates/update.d...
done.
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 quests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@master:~$
```

### 14) Download Google cloud public signing key

sudo curl -fsSLo /usr/share/keyrings/kubernetes-archive-keyring.gpg https://dl.k8s.io/apt/doc/apt-key.gpg

No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@master:~\$ sudo curl -fsSLo /usr/share/keyrings/kubernetes-archive-keyring.gpg https://dl.k8s.io/apt/doc/apt-key.gpg
ubuntu@master:~\$ []

## 15) Adding kubernetes apt repository

echo "deb [signed-by=/usr/share/keyrings/kubernetes-archive-keyring.gpg] https://apt.kubernetes.io/kubernetes-xenial main" | sudo tee /etc/apt/sources.list.d/kubernetes.list

ubuntu@master:-\$ echo "deb [signed-by=/usr/share/keyrings/kubernetes-archive-keyring.gpg] https://apt.kubernetes.io/ kubernetes-xenial main" | sudo tee /etc/apt/sources.lis t.d/kubernetes.list deb [signed-by=/usr/share/keyrings/kubernetes-archive-keyring.gpg] https://apt.kubernetes.io/ kubernetes-xenial main ubuntu@master:-\$ []

# 16) Run this 3 commands

sudo apt-get update

```
ubuntu@master:~$ sudo apt-get update
Hit:1 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy InRelease
Hit:2 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:3 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu jammy-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu jammy-security InRelease
Get:5 https://packages.cloud.google.com/apt kubernetes-xenial InRelease [8993 B]
Get:6 https://packages.cloud.google.com/apt kubernetes-xenial/main amd64 Packages [69.9 kB]
Fetched 78.9 kB in 1s (53.8 kB/s)
Reading package lists... Done
ubuntu@master:~$
```

sudo apt-get install -y kubelet kubeadm kubectl

```
Setting up conntrack (1:1.4.6-2build2) ...
Setting up kubectl (1.28.2-00) ...
Setting up ebtables (2.0.11-4build2) ...
Setting up socat (1.7.4.1-3ubuntu4) ...
Setting up cri-tools (1.26.0-00) ...
Setting up kubernetes-cni (1.2.0-00) ...
Setting up kubelet (1.28.2-00) ...
Created symlink /etc/systemd/system/multi-user.target.wants/kubelet.service 
ightarrow /lib/systemd/system/kubelet.service.
Setting up kubeadm (1.28.2-00) ...
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:~$ 🗍
sudo apt-mark hold kubelet kubeadm kubectl
ubuntu@master:~$ sudo apt-mark hold kubelet kubeadm kubectl
kubelet set on hold.
kubeadm set on hold.
kubectl set on hold.
ubuntu@master:~$ ||
```

### **KUBERNETES INSTALLED SUCCESSFULLY**

### 17) Kubernetes Deployment

sudo swapoff -a

```
ubuntu@master:~$ sudo swapoff -a ubuntu@master:~$ [
```

### 18) Initialize kubernetes on Master

```
sudo touch "/etc/docker/daemon.json"
sudo nano "/etc/docker/daemon.json" Run this command and copy paste this
{
"exec-opts": ["native.cgroupdriver=systemd"]
}
Then press ctrl + O and enter then ctrl + X
sudo cat "/etc/docker/daemon.json"
sudo systemctl daemon-reload
sudo systemctl restart docker
sudo systemctl restart kubelet
sudo kubeadm reset
```

```
ubuntu@master:~$ sudo touch "/etc/docker/daemon.json" ubuntu@master:~$ sudo nano "/etc/docker/daemon.json"
```

```
ubuntu@master:~$ sudo cat "/etc/docker/daemon.json"
        "exec-opts": ["native.cgroupdriver=systemd"],
        "log-driver": "json-file",
        "log-opts": {
        "max-size": "100m"
    },
         "storage-driver": "overlay2"
ubuntu@master:~$ 🛚
ubuntu@master:~$ sudo systemct1 daemon-reload
ubuntu@master:~$ sudo systemctl restart docker
ubuntu@master:~$ sudo systemctl restart kubelet
ubuntu@master:~$ sudo kubeadm reset
W0917 18:15:57.371540
                           10839 preflight.go:56] [reset] WARNING:
[reset] Are you sure you want to proceed? [y/N]: y
[preflight] Running pre-flight checks
[reset] Deleted contents of the etcd data directory: /var/lib/etc
[reset] Stopping the kubelet service
[reset] Unmounting mounted directories in "/var/lib/kubelet"
$ sudo kubeadm init --pod-network-cidr=10.244.0.0/16 --ignore-preflight-errors=all
ubuntu@master:~$ sudo kubeadm init --pod-network-cidr=10.244.0.0/16 --ignore-preflight-er
[init] Using Kubernetes version: v1.28.2
[preflight] Running pre-flight checks
       [WARNING NumCPU]: the number of available CPUs 1 is less than the required 2
       [WARNING Mem]: the system RAM (965 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 conn
[preflight] You can also perform this action in beforehand using 'kubeadm config images p
W0917 18:17:31.346348 10860 checks.go:835] detected that the sandbox image "registry.k8
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 kub
172.31.12.130]
[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] and IPs [172.
[certs] Generating "etcd/peer" certificate and key
```

kubeadm join 172.31.12.130:6443 --token 67nba2.98zekjx1ogwtrr29 --discovery-token-ca-cert-hash sha256:5d3403f5221016f77cbed1757a266467af45dc41b765ebe535f15ee058baf883

[certs] Generating "etcd/healthcheck-client" certificate and key
[certs] Generating "apiserver-etcd-client" certificate and key

[certs] Generating "sa" key and public key

[certs] etcd/peer serving cert is signed for DNS names [localhost master] and IPs [172.31

```
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) $HOME/.kube/config
Alternatively, if you are the root user, you can run:
 export KUBECONFIG=/etc/kubernetes/admin.conf
You should now deploy a pod network to the cluster.
Run "kubectl apply -f [podnetwork].yaml" with one of the options listed at:
 https://kubernetes.io/docs/concepts/cluster-administration/addons/
Then you can join any number of worker nodes by running the following on each as root:
kubeadm join 172.31.12.130:6443 --token 67nba2.98zekjx1ogwtrr29 \
        -discovery-token-ca-cert-hash sha256:5d3403f5221016f77cbed1757a266467af45dc4<u>1b765ebe535f15ee058baf883</u>
ubuntu@master:~$
ubuntu@master:~$ mkdir -p $HOME/.kube
ubuntu@master:~$ sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
ubuntu@master:~$ sudo chown $(id -u):$(id -g) $HOME/.kube/config
ubuntu@master:~$ ||
ubuntu@master:~$ kubectl apply -f https://github.com/flannel-io/flannel/releases/latest/download/kube-flannel.yml
namespace/kube-flannel created
serviceaccount/flannel created
clusterrole.rbac.authorization.k8s.io/flannel created
clusterrolebinding.rbac.authorization.k8s.io/flannel created
configmap/kube-flannel-cfg created
daemonset.apps/kube-flannel-ds created
ubuntu@master:~$ 🗍
[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-start] Waiting for the kubelet to perform the TLS Bootstrap...
10917 18:31:31.967826 11348 kubelet.go:220] [kubelet-start] preserving the crisocket information for th
10917 18:31:31.968119 11348 patchnode.go:31] [patchnode] Uploading the CRI Socket information "unix:///
as an annotation
10917 18:31:31.968407 11348 cert_rotation.go:137] Starting client certificate rotation controller
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:~$ ∏
ubuntu@master:~$ kubectl get pods --all-namespaces
The connection to the server 172.31.12.130:6443 was refused - did you specify the right host or port?
ubuntu@master:~$
ubuntu@master:~$ kubectl get nodes
NAME
               STATUS
                               ROLES
                                                                      VERSION
                                                           \mathbf{AGE}
               Ready
                                                                      v1.28.2
                               control-plane
                                                           24m
master
               Ready
                                                                      v1.28.2
worker
                               <none>
                                                            11m
ubuntu@master:~$
```

```
Expanded Security Maintenance for Applications is not enabled.

128 updates can be applied immediately.
68 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

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

Last login: Sun Sep 17 18:56:17 2023 from 13.233.177.5

ubuntu@master:-$ kubectl get pods --all-namespaces

NAMES PACE

NAMES PACE

NAMES PACE

NAME READY STATUS

RESTARTS

ACE

NAME-clannel da-fidef | 1/1 Running | 12 (80s ago) | 36m |

Rube-system coredns-5dd5756b68-5kjv4 | 1/1 Running | 8 (80s ago) | 40m |

Rube-system coredns-5dd5756b68-5kjv4 | 1/1 Running | 18 (80s ago) | 40m |

Rube-system kube-apisterer kube-clannelser | 1/1 Running | 18 (29s ago) | 41m |

Rube-system kube-proxy-mm6xl | 1/1 Running | 18 (29s ago) | 40m |

Rube-system kube-proxy-mm6xl | 1/1 Running | 18 (29s ago) | 40m |

Rube-system kube-proxy-mm6xl | 1/1 Running | 18 (29s ago) | 40m |

Rube-system kube-proxy-mm6xl | 1/1 Running | 18 (29s ago) | 40m |

Rube-system kube-proxy-mm6xl | 1/1 Running | 18 (29s ago) | 40m |

Rube-system kube-proxy-mm6xl | 1/1 Running | 18 (29s ago) | 40m |

Rube-system kube-proxy-mm6xl | 1/1 Running | 18 (29s ago) | 40m |

Rube-system kube-proxy-mm6xl | 1/1 Running | 18 (29s ago) | 40m |

Rube-system kube-proxy-mm6xl | 1/1 Running | 18 (29s ago) | 40m |

Rube-system kube-proxy-mm6xl | 1/1 Running | 18 (29s ago) | 40m |

Rube-system kube-proxy-mm6xl | 1/1 Running | 18 (29s ago) | 40m |

Rube-system kube-proxy-mm6xl | 1/1 Running | 18 (29s ago) | 40m |

Rube-system kube-proxy-mm6xl | 1/1 Running | 18 (29s ago) | 40m |

Rube-system kube-scheduler-master | 1/1 Running | 18 (29s ago) | 40m |

Rube-system kube-scheduler-master | 1/1 Running | 18 (29s ago) | 40m |

Rube-system kube-scheduler-master | 1/1 Running | 18 (29s ago) | 40m |

Rube-system kube-scheduler-master | 1/1 Running | 18 (29s ago) | 40m |

Rube-system kube-scheduler-master | 1/1 Running | 18 (29s ago) | 40m |

Rube-system kube-
```

#### Conclusion:

Thus we have understood the Kubernetes Cluster Architecture, installed and spun a Kubernetes Cluster on AWS Cloud Platform.

### Error:

The status of all namespaces was not showing running in the first try, but when I rebooted the master and worker instances it was successful.