********** Install Kubernetes on Master Node **********

-> Create one security group with below inbound rules

Rule-1 : "Type : SSH (22), Source : Anywhere"

Rule-2 : "Type : ALL TCP , Source: Anywhere"

-> Create 3 virtual machines using Ubuntu 20.04 version AMI by selecting above security group

1 Master Node : t2.medium

2 Worker Nodes : t2.micro

********** Note: Use UBUNTU 20.04 version AMI for all 3 machines
****

========== Master & Worker Nodes Common Commands Execution start ==========

```
# Upgrade apt packages
$ sudo apt-get update

#Create configuration file for containerd:
$ cat <<EOF | sudo tee /etc/modules-load.d/containerd.conf overlay br_netfilter
EOF

#Load modules:

$ sudo modprobe overlay
$ sudo modprobe br_netfilter


#Set system configurations for Kubernetes networking:

$ cat <<EOF | sudo tee /etc/sysctl.d/99-kubernetes-cri.conf
net.bridge.bridge-nf-call-iptables = 1
net.ipv4.ip_forward = 1
net.bridge.bridge-nf-call-ip6tables = 1
EOF


#Apply new settings:
$ sudo sysctl --system

#Install containerd:

$ sudo apt-get update && sudo apt-get install -y containerd


# Create default configuration file for containerd:

$ sudo mkdir -p /etc/containerd


#Generate default containerd configuration and save to the newly created default file:

$ sudo containerd config default | sudo tee /etc/containerd/config.toml


#Restart containerd to ensure new configuration file usage:
```

```
$ sudo systemctl restart containerd


#Verify that containerd is running (optional)

$ sudo systemctl status containerd


#Disable swap:

$ sudo swapoff -a


#Disable swap on startup in /etc/fstab:

$ sudo sed -i '/ swap / s/^\(.*\)$/#\1/g' /etc/fstab


#Install dependency packages:

$ sudo apt-get update && sudo apt-get install -y apt-transport-https curl


# Download and add GPG key:

$ curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | sudo apt-key add -


# Add Kubernetes to repository list:

$ cat <<EOF | sudo tee /etc/apt/sources.list.d/kubernetes.list
deb https://apt.kubernetes.io/ kubernetes-xenial main
EOF


Update package listings:

$ sudo apt-get update


# Install Kubernetes packages (Note: If you get a dpkg lock message, just wait a minute or two before
trying the command again):

$ sudo apt-get install -y  kubelet kubeadm kubectl kubernetes-cni nfs-common


# Turn off automatic updates:

$ sudo apt-mark hold kubelet kubeadm kubectl kubernetes-cni nfs-common


======================Common Commands Execution End=========================================


=================Only Master Node Commands Execution Start====================================


Initialize the Cluster-

Initialize the Kubernetes cluster on the control plane node using kubeadm
(Note: This is only performed on the Control Plane Node):

$ sudo kubeadm init

Note: if we will get an error as "[ERROR NumCPU]: the number of available CPUs 1 is less than the
```

```
required 2"
Kubeadm runs a series of pre-flight checks to validate the system state before making changes.

This error means the host don't have minimum requirement of 2 CPU.
You can ignore the error if you still want to go ahead and install kubernetes on this host.

sudo kubeadm init --ignore-preflight-errors=NumCPU



# Set kubectl access:

$ mkdir -p $HOME/.kube

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

$ sudo chown $(id -u):$(id -g) $HOME/.kube/config


# Test access to cluster:

$ kubectl get nodes


# Install the Calico Network Add-On -

# On the Control Plane Node, install Calico Networking:

$ kubectl apply -f https://docs.projectcalico.org/manifests/calico.yaml

$ kubectl get nodes


Join the Worker Nodes to the Cluster

In the Control Plane Node, create the token and copy the kubeadm join command (NOTE:The join command
can also be found in the output from kubeadm init command):

$ kubeadm token create --print-join-command

Note : In both Worker Nodes, paste the kubeadm join command to join the cluster. Use sudo to run it
as root:

sudo kubeadm join ...

In the Control Plane Node, view cluster status (Note: You may have to wait a few moments to allow all
nodes to become ready):

#command to join other nodes as master


======================================================
Validate the setup by executing below command in master-node
======================================================
$  kubectl get nodes
```