**K8S MULTI NODE CLUSTER SETUP**

**Master node configuration**

**To give root power to user:**

sudo su - root

**Both the nodes (master and slave) require docker containers. Master node needs containers for running different services, while worker node needs it for the client to launch the pods.**

yum install docker -y

systemctl enable docker --now

**Now the next step is to install the software for K8s setup which is kubeadm, but by default kubeadm is not provided by the repos configured in the yum, so we need to configure yum first before downloading it.**

vi /etc/yum.repos.d/kubernetes.repo

[kubernetes]

name=Kubernetes

baseurl=https://packages.cloud.google.com/yum/repos/kubernetes-el7-\$basearch

enabled=1

gpgcheck=1

repo\_gpgcheck=1

gpgkey=https://packages.cloud.google.com/yum/doc/yum-key.gpg https://packages.cloud.google.com/yum/doc/rpm-package-key.gpg

exclude=kubelet kubeadm kubectl

**Next step is to install kubeadm using yum.**

yum install kubeadm --disableexcludes=Kubernetes

**All the services – kubelet, kubeadm and kubectl is installed now, the next step is to enable kubelet.**

systemctl enable kubelet --now

**Now for running the service we need separate containers, either we can download it one by one manually or we can directly use the below command to pull all the necessary images from the docker hub.**

kubeadm config images pull

**One of the requirements is to setup the driver to systemd (by default driver is cgroup, you can check it using docker info command), use below command to do the same.**

vim /etc/docker/daemon.json

{

"exec-opts": ["native.cgroupdriver=systemd"]

}

**After making changes, restart the docker service**

systemctl restart docker

**One more requirement for networking is to have iproute-tc software, download it using yum command.**

yum install iproute-tc -y

**For network configuration, we need to run the following commands:**

cat <<EOF | sudo tee /etc/sysctl.d/k8s.conf

net.bridge.bridge-nf-call-ip6tables = 1

net.bridge.bridge-nf-call-iptables = 1

EOF

**To start the service:**

sudo sysctl --system

**Now we also need to specify the range of IP address we want to assign to the pods or containers, also because we are doing the setup on top AWS, we might not have enough RAM or CPU in the system so we can ignore these warnings/errors using “--ignore-preflight-errors” keyword.**

kubeadm init --pod-network-cidr=10.240.0.0/16 --ignore-preflight-errors=NumCPU --ignore-preflight-errors=Mem

**Now system is configured as a master.**

**Normally we have a separate client who will use kubectl command on the master, but just for testing, we can make master as the client/user. Now if you run “kubectl” command, it will fail (we already have kubectl software in the system). It will fail because client should always know where master is running (the IP of the master), so client should know the port number of API, and username and password of master, so to use this cluster as a normal user, you can copy below files in the HOME location, the files contain all the credentials of 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

**Now system is also configured as a client.**

**To configure some extra networks, just run the below commands for flannel configuration.**

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

**To generate a token so that worker nodes can join:**

kubeadm token create --print-join-command

**Worker node configuration**

**In worker nodes also, we need to do the same setup as master node except few commands like pulling docker images because it’s not required in the worker node.**

sudo su – root

yum install docker -y

systemctl enable docker --now

vi /etc/yum.repos.d/kubernetes.repo

[kubernetes]

name=Kubernetes

baseurl=https://packages.cloud.google.com/yum/repos/kubernetes-el7-\$basearch

enabled=1

gpgcheck=1

repo\_gpgcheck=1

gpgkey=https://packages.cloud.google.com/yum/doc/yum-key.gpg https://packages.cloud.google.com/yum/doc/rpm-package-key.gpg

exclude=kubelet kubeadm kubectl

yum install kubeadm --disableexcludes=Kubernetes

systemctl enable kubelet --now

vim /etc/docker/daemon.json

{

"exec-opts": ["native.cgroupdriver=systemd"]

}

systemctl restart docker

yum install iproute-tc -y

cat <<EOF | sudo tee /etc/sysctl.d/k8s.conf

net.bridge.bridge-nf-call-ip6tables = 1

net.bridge.bridge-nf-call-iptables = 1

EOF

sudo sysctl --system

**To join to the master node:**

kubeadm join 172.31.32.190:6443 --token lfjs0d.ljkluvbh2f6nu389 --discovery-token-ca-cert-hash sha256:88ca836d0ccfc7db998f06bb0934ae7bcf13bd28089885c4fd9a49a2977a6a64

**Now the system is successfully configured as a slave/worker node.**