**Installation of Kubernetes Cluster on Master & Worker Nodes**

The following steps will run on the Master and worker nodes.

**Step 1: Prepare Hostname, Firewall, and SELinux**

On your master and worker node, set the hostname and if you don’t have a DNS server, then also update your **/etc/hosts** file.

For Example:

# hostnamectl set-hostname master-node

# cat <<EOF>> /etc/hosts

10.128.0.27 master-node

10.128.0.29 node-1 worker-node-1

10.128.0.30 node-2 worker-node-2

EOF

You can ping **worker-node-1** and **worker-node-2** to test if your updated host file is fine using the [ping command](https://www.tecmint.com/linux-ping-command-examples/).

# ping 10.128.0.29

# ping 10.128.0.30

Next, disable **SELinux** and update your firewall rules.

# setenforce 0

# sed -i --follow-symlinks 's/SELINUX=enforcing/SELINUX=disabled/g' /etc/sysconfig/selinux

# reboot

Set the following firewall rules on ports. Make sure that each [firewall-cmd command](https://www.tecmint.com/install-configure-firewalld-in-centos-ubuntu/), returns a success.

# firewall-cmd --permanent --add-port=6443/tcp

# firewall-cmd --permanent --add-port=2379-2380/tcp

# firewall-cmd --permanent --add-port=10250/tcp

# firewall-cmd --permanent --add-port=10251/tcp

# firewall-cmd --permanent --add-port=10252/tcp

# firewall-cmd --permanent --add-port=10255/tcp

# firewall-cmd --permanent --add-port=30000-32767/tcp

# firewall-cmd –reload

# modprobe br\_netfilter

# echo '1' > /proc/sys/net/bridge/bridge-nf-call-iptables

**Step 2: Setup the Kubernetes Repo:**

You will need to add **Kubernetes** repositories manually on master and worker nodes as they do not come installed by default on **CentOS 7**.

cat <<EOF > /etc/yum.repos.d/Kubernetes.repo

[kubernetes]

name=Kubernetes

baseurl=https://packages.cloud.google.com/yum/repos/kubernetes-el7-x86\_64

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

EOF

**Step 3: Install Kubeadm and Docker:**

Before installation of Kubernetes, we need to install the Docker and Docker-Compose on master and worker nodes.

**Docker Installation from the following link**.

curl -fsSL https://get.docker.com/ | sh

sudo systemctl start docker.

sudo systemctl status docker

sudo systemctl enable docker.

**Docker Compose installation**:

curl -SL https://github.com/docker/compose/releases/download/v2.7.0/docker-compose-linux-x86\_64 -o /usr/local/bin/docker-compose

sudo chmod +x /usr/local/bin/docker-compose

docker-compose --version

**Kubernetes Installation:**

Install kubeadm and other important packages.

yum install -y kubelet kubeadm kubectl --disableexcludes=Kubernetes

When the installation completes successfully, enable and start both services.

# systemctl enable kubelet

# systemctl start kubelet

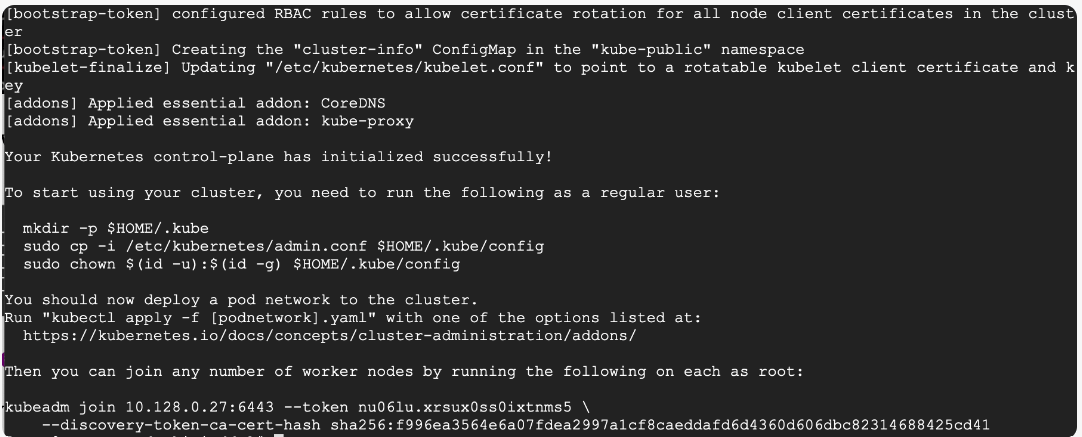
**Step 4: Initialize Kubernetes Master and Setup Default User**

Now we are ready to initialize Kubernetes master, but before that, you need to disable swap to run the “kubeadm init“ command.

# swapoff -a

Initializing Kubernetes master is a fully automated process managed by the “kubeadm init“ command you will run. Only on Master Node

# kubeadm init --pod-network-cidr 10.244.0.0/16 --apiserver-advertise-address=master node IP



# yum install bash-completion echo "source <(kubectl completion bash)" >> ~/.bashrc

To start using your cluster, you need to run the following commands on mater nodes as a regular user:

# mkdir -p $HOME/.kube

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

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

**Now check to see if the kubectl** **command is activated.**

# kubectl get nodes

Text

Description automatically generated

At this point, you will also notice that the status of the master node is ‘NotReady’. This is because we are yet to deploy the pod network to the cluster.

The pod Network is the overlay network for the cluster, that is deployed on top of the present node network. It is designed to allow connectivity across the pod.

**Step 5: Setup Your Pod Network**

Weave Net can be installed onto your CNI-enabled Kubernetes cluster with a single command:

kubectl apply -f https://github.com/weaveworks/weave/releases/download/v2.8.1/weave-daemonset-k8s.yaml

**Text

Description automatically generated**

Now if you check the status of your **master node**, it should be ‘**Ready**’.

# kubectl get nodes

Text

Description automatically generated

**Setting Up Worker Nodes to Join Kubernetes Cluster:**

You will find the kubeadm join token from the master node. You can copy and paste it to your **worker nodes** if you had copied it somewhere. Otherwise, you can generate it again with the following command.

# kubeadm token create --print-join-command

Text

Description automatically generated

As suggested in the last line, go back to your **master node** and check if **worker node-1** and **worker node-2** have joined the cluster using the following command.

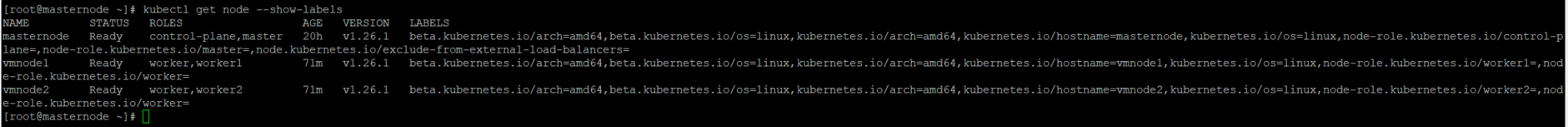
# kubectl get nodes

Table

Description automatically generated with medium confidence

You will see the ROLES with the following command.

# kubectl get node --show-labels

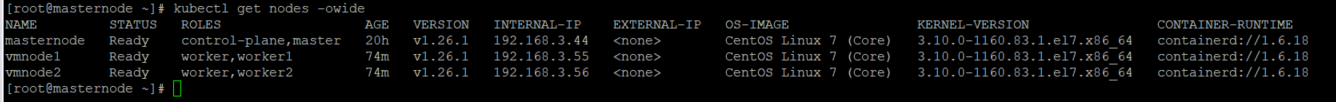


With the following command, you can assign the ROLES to the worker node as workers.

# kubectl label node vmnode2 node-role.kubernetes.io/worker=

After running the kubectl label command you can run the following command to see the worker node roles assigned.

# kubectl get nodes -owide



**kubeadm init error: CRI v1 runtime API is not implemented**

To resolve the issue remove the file config.toml from the following path.

rm -rf /etc/containerd/config.toml

then restart the containerd services

systemctl restart containerd