**How To Create Kubernetes Cluster With Containerd**

Server List

**Step 1. Install and Configure Containerd**

Do this configuration on all node

Setup host files so node can communicate each other with names

sudo vi /etc/hosts#Add the following list in the end of line  
172.16.4.90 kubmaster.demo  
172.16.4.91 kubworker1.demo  
172.16.4.92 kubworker2.demo #save and exit with :wq

configure modules required by containerd

sudo modprobe overlay  
sudo modprobe br\_netfilter  
cat <<EOF | sudo tee /etc/modules-load.d/containerd.conf  
overlay  
br\_netfilter  
EOF

Install containerd service

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

Configure containerd to use systemd as cgroup driver

sudo mkdir -p /etc/containerd  
sudo containerd config default | sudo tee /etc/containerd/config.toml  
sudo vi /etc/containerd/config.toml#find the [plugins."io.containerd.grpc.v1.cri".containerd.runtimes.runc.options] section and change systemdcgroup to trueSystemdCgroup = true#save and exit with :wq

Restart and enable containerd service

sudo systemctl restart containerd  
sudo systemctl enable containerd

Verify containerd configuration

sudo containerd config dump

**Step 2. install Kubernetes**

Do this configuration on all nodes

Configure sysctl

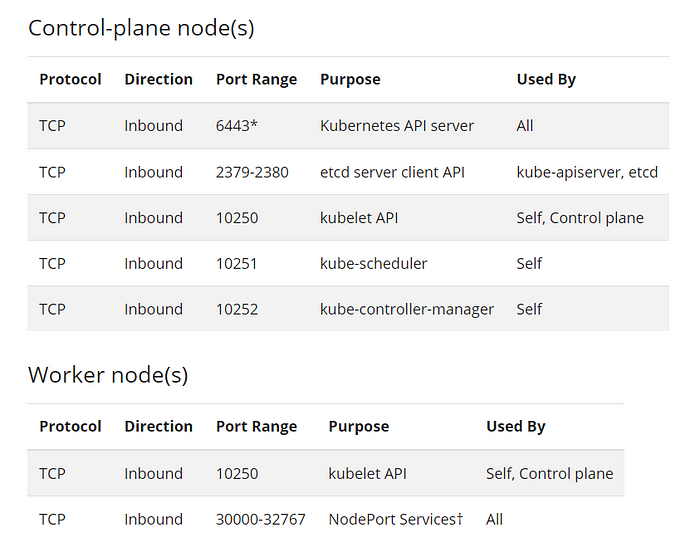
cat <<EOF | sudo tee /etc/sysctl.d/99-kubernetes-cri.conf  
net.bridge.bridge-nf-call-iptables = 1  
net.ipv4.ip\_forward = 1  
net.bridge.bridge-nf-call-ip6tables = 1  
EOF

sudo sysctl --system

Disable Swap

sudo swapoff -a  
sudo vi /etc/fstab#comment swap disk  
#/swap.img none swap sw 0 0#save and exit :wq

Allow required port for kubernetes or Disable Firewall



sudo ufw disable

Setup iptables backend to use iptables-legacy

sudo update-alternatives --config iptablesThere are 2 choices for the alternative iptables (providing /usr/sbin/iptables).Selection Path Priority Status  
------------------------------------------------------------  
 0 /usr/sbin/iptables-nft 20 auto mode  
\* 1 /usr/sbin/iptables-legacy 10 manual mode  
 2 /usr/sbin/iptables-nft 20 manual modePress <enter> to keep the current choice[\*], or type selection number:

Add kubernetes repository

sudo wget <https://packages.cloud.google.com/apt/doc/apt-key.gpg>  
sudo mv apt-key.gpg /etc/apt/trusted.gpg.d/sudo cat <<EOF | sudo tee /etc/apt/sources.list.d/kubernetes.list  
deb <https://apt.kubernetes.io/> kubernetes-xenial main  
EOFsudo apt update  
sudo apt install -y kubeadm kubelet kubectl

Enable kubelet service but dont start yet

sudo systemctl enable kubelet

**Step 3. Kubernetes Cluster Init**

Do this configuration only on master nodes

Initialize cluster using containerd as container runtime

sudo kubeadm init --pod-network-cidr 192.168.0.0/16 --cri-socket /run/containerd/containerd.sock

Note the final result cause we will use it for join worker nodes

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/configAlternatively, if you are the root user, you can run:export KUBECONFIG=/etc/kubernetes/admin.confYou 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 172.16.4.90:6443 --token 7i34jm.q8enu8wxvfic9s8k \  
 --discovery-token-ca-cert-hash sha256:202117e62f133323eff707919ec512eef466a59a29454c4ee320a0626ff42c05

Create kubeconfig file to use kubectl command

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

**Step 4. Join Worker Nodes**

Do this configuration only on worker nodes

Paste kubeadm join from the previous result of cluster init and dont forget to use containerd as container runtime

sudo kubeadm join 172.16.4.90:6443 --token 7i34jm.q8enu8wxvfic9s8k \  
 --discovery-token-ca-cert-hash sha256:202117e62f133323eff707919ec512eef466a59a29454c4ee320a0626ff42c05 --cri-socket /run/containerd/containerd.sock

**Step 5. Install Calico CNI**

Do this configuration on master nodes

install calico cni

kubectl apply -f calico.yaml

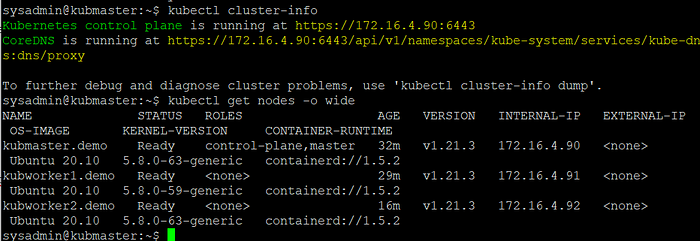
Verify pod status

sysadmin@kubmaster:~$ kubectl get pods -n kube-system  
NAMESPACE NAME READY STATUS RESTARTS AGE  
kube-system calico-kube-controllers-d7c67954f-n5cms 0/1 ContainerCreating 0 74s  
kube-system calico-node-c888q 0/1 PodInitializing 0 74s  
kube-system calico-node-llmjp 0/1 PodInitializing 0 74s  
kube-system calico-node-lq8hc 0/1 Running 0 74s  
kube-system coredns-558bd4d5db-4c7gz 0/1 ContainerCreating 0 20m  
kube-system coredns-558bd4d5db-tg5t2 0/1 ContainerCreating 0 20m  
kube-system etcd-kubmaster.demo 1/1 Running 0 20m  
kube-system kube-apiserver-kubmaster.demo 1/1 Running 0 20m  
kube-system kube-controller-manager-kubmaster.demo 1/1 Running 0 20m  
kube-system kube-proxy-d7cqg 1/1 Running 0 20m  
kube-system kube-proxy-m7pfg 1/1 Running 1 17m  
kube-system kube-proxy-q596m 1/1 Running 0 5m  
kube-system kube-scheduler-kubmaster.demo 1/1 Running 0 20m

Wait until all pod running

sysadmin@kubmaster:~$ kubectl get pods -n kube-system  
NAME READY STATUS RESTARTS AGE  
calico-kube-controllers-d7c67954f-n5cms 1/1 Running 0 2m31s  
calico-node-c888q 1/1 Running 0 2m31s  
calico-node-llmjp 1/1 Running 0 2m31s  
calico-node-lq8hc 1/1 Running 0 2m31s  
coredns-558bd4d5db-4c7gz 1/1 Running 0 21m  
coredns-558bd4d5db-tg5t2 1/1 Running 0 21m  
etcd-kubmaster.demo 1/1 Running 0 21m  
kube-apiserver-kubmaster.demo 1/1 Running 0 21m  
kube-controller-manager-kubmaster.demo 1/1 Running 0 22m  
kube-proxy-d7cqg 1/1 Running 0 21m  
kube-proxy-m7pfg 1/1 Running 1 18m  
kube-proxy-q596m 1/1 Running 0 6m17s  
kube-scheduler-kubmaster.demo 1/1 Running 0 21m  
sysadmin@kubmaster:~$

Verify node status and cluster info



verify cluster

*Thats all the basic configuration cluster using containerd as container runtime, in next article i will share how to configure multimaster kubernetes cluster for high availability using containerd as a container runtime.*

*cheers*