**********KUBERNETES_INSTALLATION**********

Install Kubernetes Cluster on CentOS 7 with kubeadm and CRI-O

 # create 3 vm of centos image

Server Type	Server Hostname	Specs
Master	master-node	4GB Ram, 2vcpus
Worker	node1	2GB Ram, 1vcpus
Worker	node2	2GB Ram, 1vcpus

 # edit host file and write all ip-address of all machine on all system

vi /etc/hosts

```
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6
192.168.174.166 master
192.168.174.167 node1
192.168.174.168 node2

~
```

 # make sure all 3 pc ping to each other

 # create keygen on master node

Ssh-keygen

 # copy keygen to all client node

Ssh-copy-id

 # Update all system

yum update

 # disable selinux on all system

setenforce 0

'or'

vi /etc/selinux/config

SELINUX=disabled

```
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#     enforcing - SELinux security policy is enforced.
#     permissive - SELinux prints warnings instead of enforcing.
#     disabled - No SELinux policy is loaded.
SELINUX=disabled
# SELINUXTYPE= can take one of three values:
#     targeted - Targeted processes are protected,
#     minimum - Modification of targeted policy. Only selected processes are protected.
#     mls - Multi Level Security protection.
SELINUXTYPE=targeted
```

 # firewall disable on all system
systemctl disable firewalld
systemctl stop firewalld

 # restat all system on all machine
init 6

```
[root@node2 ~]# modprobe overlay
[root@node2 ~]# modprobe br_netfilter
[root@node2 ~]# tee /etc/sysctl.d/kubernetes.conf<<EOF
> net.bridge.bridge-nf-call-ip6tables = 1
> net.bridge.bridge-nf-call-iptables = 1
> net.ipv4.ip_forward = 1
> EOF
net.bridge.bridge-nf-call-ip6tables = 1
net.bridge.bridge-nf-call-iptables = 1
net.ipv4.ip_forward = 1
[root@node2 ~]# sysctl --system
* Applying /usr/lib/sysctl.d/00-system.conf ...
net.bridge.bridge-nf-call-ip6tables = 0
net.bridge.bridge-nf-call-iptables = 0
net.bridge.bridge-nf-call-arptables = 0
* Applying /usr/lib/sysctl.d/10-default-yama-scope.conf ...
kernel.yama.ptrace_scope = 0
* Applying /usr/lib/sysctl.d/50-default.conf ...
kernel.sysrq = 16
kernel.core_uses_pid = 1
kernel.kptr_restrict = 1
net.ipv4.conf.default.rp_filter = 1
net.ipv4.conf.all.rp_filter = 1
net.ipv4.conf.default.accept_source_route = 0
net.ipv4.conf.all.accept_source_route = 0
net.ipv4.conf.default.promote_secondaries = 1
net.ipv4.conf.all.promote_secondaries = 1
fs.protected_hardlinks = 1
fs.protected_symlinks = 1
* Applying /usr/lib/sysctl.d/60-libvirtd.conf ...
fs.aio-max-nr = 1048576
* Applying /etc/sysctl.d/99-sysctl.conf ...
* Applying /etc/sysctl.d/kubernetes.conf ...
net.bridge.bridge-nf-call-ip6tables = 1
net.bridge.bridge-nf-call-iptables = 1
net.ipv4.ip_forward = 1
* Applying /etc/sysctl.conf ...
[root@node2 ~]#
```

 # add repo file on all machine

tee /etc/yum.repos.d/kubernetes.repo<<EOF
[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

```
[root@master ~]# tee /etc/yum.repos.d/kubernetes.repo<<EOF
> [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
[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/r
pm-package-key.gpg
[root@master ~]#
```

 # install packages on all machine
yum clean all && yum -y makecache
yum -y install epel-release git curl wget kubelet kubeadm kubectl --disableexcludes=kubernetes

 # start kubelet service on all machine
systemctl start kubelet

 # enable kubelet service on all machine
Systemctl enable kubelet

 # Add CRI-O repo on all machine
OS=CentOS_7
VERSION=1.22
curl -L -o /etc/yum.repos.d/devel:kubic:libcontainers:stable.repo
https://download.opensuse.org/repositories/devel:/kubic:/libcontainers:/stable/$OS/devel:kubic:libcontainers:stable.repo

curl -L -o /etc/yum.repos.d/devel:kubic:libcontainers:stable:cri-o:$VERSION.repo
https://download.opensuse.org/repositories/devel:kubic:libcontainers:stable:cri-o:$VERSION/$OS/devel:kubic:libcontainers:stable:cri-o:$VERSION.repo

```
[root@node2 ~]# OS=CentOS_7
[root@node2 ~]# VERSION=1.22
[root@node2 ~]# curl -L -o /etc/yum.repos.d/devel:kubic:libcontainers:stable.repo https://download.opens
use.org/repositories/devel:/kubic:/libcontainers:/stable/$OS/devel:kubic:libcontainers:stable.repo
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   359  100   359    0     0    188      0  0:00:01  0:00:01 --:--:--   188
[root@node2 ~]# curl -L -o /etc/yum.repos.d/devel:kubic:libcontainers:stable:cri-o:$VERSION.repo https:/
/download.opensuse.org/repositories/devel:kubic:libcontainers:stable:cri-o:$VERSION/$OS/devel:kubic:libc
ontainers:stable:cri-o:$VERSION.repo
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   425  100   425    0     0    694      0 --:--:-- --:--:-- --:--:--   695
100   426  100   426    0     0    574      0 --:--:-- --:--:-- --:--:--     0
100   427  100   427    0     0    492      0 --:--:-- --:--:-- --:--:--   492
100   428  100   428    0     0    431      0 --:--:-- --:--:-- --:--:--   431
100   429  100   429    0     0    379      0  0:00:01  0:00:01 --:--:--   379
100   381  100   381    0     0    112      0  0:00:03  0:00:03 --:--:--   301
[root@node2 ~]#
```

#Install CRI-O on all machine
yum remove docker-ce docker-ce-cli containerd.io
yum install cri-o -y

 # Start and enable Service on all machine
systemctl daemon-reload
systemctl start crio
systemctl enable crio

<u>Only on master:-</u>

 # check status of br_netfilter
lsmod | grep br_netfilter

```
[root@master ~]# lsmod | grep br_netfilter
br_netfilter           22256  0
bridge                155432  1 br_netfilter
```

 # Pull image of cubeadm
kubeadm config images pull

```
[root@master ~]# kubeadm config images pull
[config/images] Pulled registry.k8s.io/kube-apiserver:v1.26.0
[config/images] Pulled registry.k8s.io/kube-controller-manager:v1.26.0
[config/images] Pulled registry.k8s.io/kube-scheduler:v1.26.0
[config/images] Pulled registry.k8s.io/kube-proxy:v1.26.0
[config/images] Pulled registry.k8s.io/pause:3.9
[config/images] Pulled registry.k8s.io/etcd:3.5.6-0
[config/images] Pulled registry.k8s.io/coredns/coredns:v1.9.3
```

create cluster of cubernete
kubeadm init --pod-network-cidr=10.85.0.0/16 --upload-certs --control-plane-endpoint=master

outpt:-

```
Please note that the certificate-key gives access to cluster sensitive data, keep it secret!
As a safeguard, uploaded-certs will be deleted in two hours; If necessary, you can use
"kubeadm init phase upload-certs --upload-certs" to reload certs afterward.

Then you can join any number of worker nodes by running the following on each as root:

kubeadm join master:6443 --token ifs1ly.62vcct4kwzxhon9c \
        --discovery-token-ca-cert-hash sha256:b6c4ba907e6d3a580c30678acc602ed010f0f92e5d9b22134243efe2c6
```

Copy token (last 2nd line)

 # create a file and paste token

nano join-token

```
 GNU nano 2.3.1                    File: join-token

kubeadm join master:6443 --token ifs1ly.62vcct4kwzxhon9c \
        --discovery-token-ca-cert-hash sha256:b6c4ba907e6d3a580c30678acc602ed010f0f92e5d9b22134243efe2c6
```

 # Give permission to this file and check

chmod u+x join-token

```
[root@master ~]# chmod u+x join-token
[root@master ~]# ls
anaconda-ks.cfg  Documents  initial-setup-ks.cfg  Music     Public     Videos
Desktop          Downloads  join-token            Pictures  Templates
```

 # copy the file on all node

```
[root@master ~]# rsync join-token root@node1:/root/
[root@master ~]# rsync join-token root@node2:/root/
```

 # Run token on both node

./join-token

 # check node on master node

kubectl get nodes

```
[root@master ~]# kubectl get nodes
NAME     STATUS   ROLES           AGE   VERSION
master   Ready    control-plane   34m   v1.26.0
node1    Ready    <none>          17s   v1.26.0
node2    Ready    <none>          9s    v1.26.0
```

Install network plugin:-

Install Calico:

 # Install the Tigera Calico operator and custom resource definitions.

kubectl create -f
https://raw.githubusercontent.com/projectcalico/calico/v3.24.5/manifests/tigera-operator.yaml

Install Calico by creating the necessary custom resource
kubectl create -f
https://raw.githubusercontent.com/projectcalico/calico/v3.24.5/manifests/custom-resources.yaml

show all podes
kubectl get pods --all-namespaces

```
[root@master ~]# kubectl get pods --all-namespaces
NAMESPACE          NAME                                READY   STATUS    RESTARTS   AGE
kube-system        coredns-787d4945fb-dx5zn            1/1     Running   0          43m
kube-system        coredns-787d4945fb-p7657            1/1     Running   0          43m
kube-system        etcd-master                         1/1     Running   0          43m
kube-system        kube-apiserver-master               1/1     Running   0          43m
kube-system        kube-controller-manager-master      1/1     Running   0          43m
kube-system        kube-proxy-9sb2x                    1/1     Running   0          43m
kube-system        kube-proxy-mxdkg                    1/1     Running   0          8m53s
kube-system        kube-proxy-qmt7n                    1/1     Running   0          8m45s
kube-system        kube-scheduler-master               1/1     Running   0          43m
tigera-operator    tigera-operator-7795f5d79b-pvvl2    1/1     Running   0          3m34s
```

show podes in detail information
kubectl get pods --all-namespaces -o wide

```
[root@master ~]# kubectl get pods --all-namespaces -o wide
NAMESPACE          NAME                                READY   STATUS    RESTARTS   AGE    IP
   NODE     NOMINATED NODE   READINESS GATES
kube-system        coredns-787d4945fb-dx5zn            1/1     Running   0          44m    10.85.0.3
   master   <none>           <none>
kube-system        coredns-787d4945fb-p7657            1/1     Running   0          44m    10.85.0.2
   master   <none>           <none>
kube-system        etcd-master                         1/1     Running   0          44m    192.168.174.168
   master   <none>           <none>
kube-system        kube-apiserver-master               1/1     Running   0          44m    192.168.174.168
   master   <none>           <none>
kube-system        kube-controller-manager-master      1/1     Running   0          44m    192.168.174.168
   master   <none>           <none>
kube-system        kube-proxy-9sb2x                    1/1     Running   0          44m    192.168.174.168
   master   <none>           <none>
kube-system        kube-proxy-mxdkg                    1/1     Running   0          10m    192.168.174.167
   node1    <none>           <none>
kube-system        kube-proxy-qmt7n                    1/1     Running   0          10m    192.168.174.166
   node2    <none>           <none>
kube-system        kube-scheduler-master               1/1     Running   0          44m    192.168.174.168
   master   <none>           <none>
tigera-operator    tigera-operator-7795f5d79b-pvvl2    1/1     Running   0          5m5s   192.168.174.166
   node2    <none>           <none>
```

$$$$$ BASIC_COMMANDS $$$$$$

 # for create container
kubectl create deployment nginx-image --image=nginx --replicas=1
replicas =>how many pod want to create

 # create container with port
kubectl expose deployment nginx-image --port=9000 --target-port=80 --name=web1
Target port =>port of container
Port =>port of pod
By default port is cluster ip

 # create container with port and type
kubectl expose deployment nginx-image --port=9000 --target-port=80 --name=web2 --type=NodePort

for delete the services
Kubectl delete services 'service_name'

For delete deployment
Kubectl delet deployment nginx-image

For show pods
Kubectl get pods

****

Types of services
=> cluster ip
=> Node port
=> Load balencer

* If we want in internal changes the we want need to make changes in '.yaml' file

Worker node(s)

Protocol	Direction	Port Range
TCP	**Inbound**	**10250**
TCP	Inbound	30000-32767