Step 1: Pre-requisites

1.a.. Check the OS, Hardware Configurations & Network connectivity

1.b.. Turn off the swap & firewall

```
swapoff -a
systemctl stop firewalld
systemctl disable firewalld
```

Step 2. Configure the local IP tables to see the Bridged Traffic

2.a.. Enable the bridged traffic

```
lsmod | grep br_netfilter
sudo modprobe br_netfilter
lsmod | grep br_netfilter
```

2.b.. Copy the below contents in this file..

```
cat <<EOF | sudo tee /etc/modules-load.d/k8s.conf
        br netfilter
EOF
```

2.c.. Copy the below contents in this file.. /etc/sysctl.d/k8s.conf

```
cat <<EOF | sudo tee /etc/modules-load.d/k8s.conf
net.bridge.bridge-nf-call-ip6tables = 1
net.bridge.bridge-nf-call-iptables = 1
EOF
sysctl --system
```

Step 3. Install Docker as a Container RUNTIME

3.a.. Uninstall any Older versions

```
# yum remove docker \
            docker-client \
            docker-client-latest \
            docker-common \
            docker-latest \
            docker-latest-logrotate \
            docker-logrotate \
            docker-engine \
            podman \
            runc
```

yum install -y yum-utils

yum-config-manager --add-repo https://download.docker.com/linux/rhel/docker-ce.repo

yum install -y docker-ce docker-ce-cli containerd.io (or) yum install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin

systemctl start docker

systemctl enable docker

systemctl status docker

Step 4. Install kubeadm, kubectl, kubelet

4.a.. Copy the below contents in this file.. /etc/yum.repos.d/kubernetes.repo

```
cat <<EOF | sudo tee /etc/yum.repos.d/kubernetes.repo
[kubernetes]
name=Kubernetes
baseurl=https://pkgs.k8s.io/core:/stable:/v1.31/rpm/
enabled=1
gpgcheck=1
gpgkey=https://pkgs.k8s.io/core:/stable:/v1.31/rpm/repodata/repomd.xml.key
exclude=kubelet kubeadm kubectl cri-tools kubernetes-cni
EOF
```

4.b.. Set SELinux in permissive mode (effectively disabling it)

```
setenforce 0
sed -i 's/^SELINUX=enforcing$/SELINUX=permissive/' /etc/selinux/config
yum install -y kubelet kubeadm kubectl --disableexcludes=kubernetes
systemctl enable --now kubelet
systemctl status  kubelet
```

Step 5. Configuring a cgroup driver

Ignore if docker is used as a CRI

Step 6. Deploy a kubernetes cluster using kubeadm

Run only in Master node

kubeadm init --pod-network-cidr=10.10.0.0/16 --apiserver-advertise-address=(masternode-ip)192.168.0.129

Note: if facing any issue kubeadm ini run below commands to solve erreor

rm /etc/containerd/config.toml

systemctl restart containerd

kubeadm init --pod-network-cidr=10.10.0.0/16 --apiserver-advertise-address=(masternode-ip)192.168.0.129

To start using your cluster, you need to run the following 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

Alternatively, if you are the root user, you can run:

export KUBECONFIG=/etc/kubernetes/admin.conf

You should now deploy a pod network to the cluster.

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

Step 7. Join the worker nodes to the master

Run in Worker Nodes as "Root"(node side only)

kubeadm init --pod-network-cidr=10.10.0.0/16 --apiserver-advertise-address=(masternode-ip)192.168.0.129

(or)

goto master node and type below command to node join token

kubeadm token create --print-join-command

Note1: Incast any issue join the node to master node type below command

kubeadm reset

Note2: In case nodes are not ready type below command on master side

systemctl restart kubelet

Step 8. Access the K8s Cluster & Deploy a POD

kubectl run vsparkz --image nginx