<https://www.linuxtechi.com/setup-highly-available-kubernetes-cluster-kubeadm/> #Reference Site

Installation:

3 Masters \\ 2 worker \ 1 HAProxy

**#Setting The HostNames**

hostnamectl set-hostname k8s-master-1

hostnamectl set-hostname k8s-master-2

hostnamectl set-hostname k8s-master-3

hostnamectl set-hostname k8s-worker-1

hostnamectl set-hostname k8s-worker-2

hostnamectl set-hostname vip-k8s-master

**#setting theHosts**

# cat <<EOF>> /etc/hosts

172.31.3.214 k8s-master-1

172.31.8.151 k8s-master-2

172.31.13.102 k8s-master-3

172.31.9.80 k8s-worker-1

172.31.14.0 k8s-worker-2

172.31.12.184 vip-k8s-master

EOF

**Install and Configure Keepalive and HAProxy on all master / control plane nodes**

sudo yum install haproxy keepalived -y

sudo vi /etc/keepalived/check\_apiserver.sh

#!/bin/sh

APISERVER\_VIP=172.31.12.184

APISERVER\_DEST\_PORT=6443

errorExit() {

echo "\*\*\* $\*" 1>&2

exit 1

}

curl --silent --max-time 2 --insecure https://localhost:${APISERVER\_DEST\_PORT}/ -o /dev/null || errorExit "Error GET https://localhost:${APISERVER\_DEST\_PORT}/"

if ip addr | grep -q ${APISERVER\_VIP}; then

curl --silent --max-time 2 --insecure https://${APISERVER\_VIP}:${APISERVER\_DEST\_PORT}/ -o /dev/null || errorExit "Error GET https://${APISERVER\_VIP}:${APISERVER\_DEST\_PORT}/"

fi

:wq

sudo chmod +x /etc/keepalived/check\_apiserver.sh

**Take the backup of keepalived.conf file and then truncate the file.**

sudo cp /etc/keepalived/keepalived.conf /etc/keepalived/keepalived.conf-org

sudo sh -c '> /etc/keepalived/keepalived.conf'

sudo vi /etc/keepalived/keepalived.conf

! /etc/keepalived/keepalived.conf

! Configuration File for keepalived

global\_defs {

router\_id LVS\_DEVEL

}

vrrp\_script check\_apiserver {

script "/etc/keepalived/check\_apiserver.sh"

interval 3

weight -2

fall 10

rise 2

}

vrrp\_instance VI\_1 {

**state MASTER**

interface **ens5**

virtual\_router\_id 151

**priority 255**

authentication {

auth\_type PASS

auth\_pass P@##D321!

}

virtual\_ipaddress {

172.31.12.184/24

}

track\_script {

check\_apiserver

}

}

:wq

sudo cp /etc/haproxy/haproxy.cfg /etc/haproxy/haproxy.cfg-org

**Remove all lines after default section and add following lines**

sudo vi /etc/haproxy/haproxy.cfg

#---------------------------------------------------------------------

# apiserver frontend which proxys to the masters

#---------------------------------------------------------------------

frontend apiserver

bind \*:8443

mode tcp

option tcplog

default\_backend apiserver

#---------------------------------------------------------------------

# round robin balancing for apiserver

#---------------------------------------------------------------------

backend apiserver

option httpchk GET /healthz

http-check expect status 200

mode tcp

option ssl-hello-chk

balance roundrobin

server k8s-master-1 172.31.3.214:6443 check

server k8s-master-2 172.31.8.151:6443 check

server k8s-master-3 172.31.13.102:6443 check

:wq

**Now copy theses three files (check\_apiserver.sh , keepalived.conf and haproxy.cfg) from k8s-master-1 to k8s-master-2 & 3**

Run the following for loop to scp these files to master 2 and 3

for f in k8s-master-2 k8s-master-3; do scp /etc/keepalived/check\_apiserver.sh /etc/keepalived/keepalived.conf root@$f:/etc/keepalived; scp /etc/haproxy/haproxy.cfg root@$f:/etc/haproxy; done

sudo systemctl enable keepalived --now

sudo systemctl enable haproxy –now

**Disabled the SELINUX**

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

# reboot

sudo modprobe br\_netfilter

sudo sh -c "echo '1' > /proc/sys/net/bridge/bridge-nf-call-iptables"

sudo sh -c "echo '1' > /proc/sys/net/ipv4/ip\_forward"

Install Container Run Time (CRI) Docker on Master & Worker Nodes

need to create a file in all masters

**vi /etc/docker/daemon.json**

**{**

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

**}**

yum remove docker \

docker-client \

docker-client-latest \

docker-common \

docker-latest \

docker-latest-logrotate \

docker-logrotate \

docker-engine

sudo yum install -y yum-utils

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

sudo yum install docker-ce -y

sudo systemctl enable docker –now

**Install Kubeadm, kubelet and kubectl**

cat <<EOF | sudo tee /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

EOF

**sudo yum install -y kubelet kubeadm kubectl --disableexcludes=kubernetes**

systemctl restart kubelet –now

systemctl enable kubelet –now

systemctl restart docker-now

systemctl enable docker –now

sudo kubeadm init --control-plane-endpoint "172.31.12.184:8443" --upload-certs

[kadmin@k8s-master-1 ~]$ mkdir -p $HOME/.kube

[kadmin@k8s-master-1 ~]$ sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config

[kadmin@k8s-master-1 ~]$ sudo chown $(id -u):$(id -g) $HOME/.kube/config

[kadmin@k8s-master-1 ~]$

Now, Let’s deploy pod network (CNI – Container Network Interface), in my case I going to deploy calico addon as pod network, run following kubectl command

[kadmin@k8s-master-1 ~]$ kubectl apply -f https://docs.projectcalico.org/v3.14/manifests/calico.yaml

Once the pod network is deployed successfully, add remaining two master nodes to cluster.

[kadmin@k8s-master-2 ~]$ sudo kubeadm join vip-k8s-master:8443 --token tun848.2hlz8uo37jgy5zqt  --discovery-token-ca-cert-hash sha256:d035f143d4bea38d54a3d827729954ab4b1d9620631ee330b8f3fbc70324abc5 --control-plane --certificate-key a0b31bb346e8d819558f8204d940782e497892ec9d3d74f08d1c0376dc3d3ef4

[kadmin@k8s-master-3 ~]$ sudo kubeadm join vip-k8s-master:8443 --token tun848.2hlz8uo37jgy5zqt  --discovery-token-ca-cert-hash sha256:d035f143d4bea38d54a3d827729954ab4b1d9620631ee330b8f3fbc70324abc5 --control-plane --certificate-key a0b31bb346e8d819558f8204d940782e497892ec9d3d74f08d1c0376dc3d3ef4

kubectl get nodes

[kadmin@k8s-master-1 ~]$ kubectl get nodes

NAME STATUS ROLES AGE VERSION

k8s-master-1 Ready master 31m v1.18.6

k8s-master-2 Ready master 10m v1.18.6

k8s-master-3 Ready master 3m47s v1.18.6

**Join Worker nodes to Kubernetes cluster**

[kadmin@k8s-worker-1 ~]$ sudo kubeadm join vip-k8s-master:8443 --token tun848.2hlz8uo37jgy5zqt --discovery-token-ca-cert-hash sha256:d035f143d4bea38d54a3d827729954ab4b1d9620631ee330b8f3fbc70324abc5

[kadmin@k8s-worker-2 ~]$ sudo kubeadm join vip-k8s-master:8443 --token tun848.2hlz8uo37jgy5zqt --discovery-token-ca-cert-hash sha256:d035f143d4bea38d54a3d827729954ab4b1d9620631ee330b8f3fbc70324abc5

To get status worker nodes,

**[kadmin@k8s-master-1 ~]$ kubectl get nodes**

NAME STATUS ROLES AGE VERSION

k8s-master-1 Ready master 43m v1.18.6

k8s-master-2 Ready master 21m v1.18.6

k8s-master-3 Ready master 15m v1.18.6

k8s-worker-1 Ready <none> 6m11s v1.18.6

k8s-worker-2 Ready <none> 5m22s v1.18.6

[kadmin@k8s-master-1 ~]$

kubectl create deployment nginx-lab --image=nginx

deployment.apps/nginx-lab created

[[kadmin@k8s-master-1 ~]$ kubectl get pods

NAME READY STATUS RESTARTS AGE

nginx-lab-5df4577d49-rzv9q 1/1 Running 0 68s

test-844b65666c-pxpkh 1/1 Running 3 154m

[kadmin@localhost ~]$

[[kadmin@k8s-master-1 ~]$ kubectl scale deployment nginx-lab --replicas=4

deployment.apps/nginx-lab scaled

[[kadmin@k8s-master-1 ~]$

[[kadmin@k8s-master-1 ~]$ kubectl get deployments.apps nginx-lab

NAME READY UP-TO-DATE AVAILABLE AGE

nginx-lab 4/4 4 4 3m10s

[[kadmin@k8s-master-1 ~]$ kubectl expose deployment nginx-lab --name=nginx-lab --type=NodePort --port=80 --target-port=80

service/nginx-lab exposed

To access nginx web **server we can use any master or worker node IP and port as “31766”**

[kadmin@localhost ~]$ curl http:// 172.31.14.0 :31766