**HA K8 Cluster By DevOps Shack**

To set up a highly available Kubernetes cluster with two master nodes and three worker nodes without using a cloud load balancer, you can use a virtual machine to act as a load balancer for the API server. Here are the detailed steps for setting up such a cluster:

**Prerequisites**

* 3 master nodes
* 3 worker nodes
* 1 load balancer node
* All nodes should be running a Linux distribution like Ubuntu

**Step 1: Prepare the Load Balancer Node**

**Install HAProxy**

sudo apt-get update

sudo apt-get install -y haproxy

**Configure HAProxy**

**Edit the HAProxy configuration file (/etc/haproxy/haproxy.cfg):**

sudo nano /etc/haproxy/haproxy.cfg

Add the following configuration:

frontend kubernetes-frontend

bind \*:6443

option tcplog

mode tcp

default\_backend kubernetes-backend

backend kubernetes-backend

mode tcp

balance roundrobin

option tcp-check

server master1 <MASTER1\_IP>:6443 check

server master2 <MASTER2\_IP>:6443 check

**Restart HAProxy**

sudo systemctl restart haproxy

**Step 2: Prepare All Nodes (Masters and Workers)**

**Install Docker, kubeadm, kubelet, and kubectl**

sudo apt-get update

sudo apt install docker.io -y

sudo chmod 666 /var/run/docker.sock

sudo apt-get install -y apt-transport-https ca-certificates curl gnupg

sudo mkdir -p -m 755 /etc/apt/keyrings

curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.30/deb/Release.key | sudo gpg --dearmor -o /etc/apt/keyrings/kubernetes-apt-keyring.gpg

echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.30/deb/ /' | sudo tee /etc/apt/sources.list.d/kubernetes.list

sudo apt update

sudo apt install -y kubeadm=1.30.0-1.1 kubelet=1.30.0-1.1 kubectl=1.30.0-1.1

**Step 3: Initialize the First Master Node**

sudo kubeadm init --control-plane-endpoint "LOAD\_BALANCER\_IP:6443" --upload-certs --pod-network-cidr=10.244.0.0/16

**Set up kubeconfig for the first master node**

mkdir -p $HOME/.kube

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

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

**Install Calico network plugin**

kubectl apply -f https://docs.projectcalico.org/manifests/calico.yaml

**Install Ingress-NGINX Controller**

kubectl apply -f https://raw.githubusercontent.com/kubernetes/ingress-nginx/controller-v0.49.0/deploy/static/provider/baremetal/deploy.yaml

**Step 4: Join the Second & Third Master Node**

kubeadm token create --print-join-command --certificate-key $(kubeadm init phase upload-certs --upload-certs | tail -1)

Run the join command on the second and third master nodes:

sudo kubeadm join LOAD\_BALANCER\_IP:6443 --token <token> --discovery-token-ca-cert-hash sha256:<hash> --control-plane --certificate-key <certificate-key>

**Step 5: Join the Worker Nodes**

kubeadm token create --print-join-command

Run the join command on each worker node:

sudo kubeadm join LOAD\_BALANCER\_IP:6443 --token <token> --discovery-token-ca-cert-hash sha256:<hash>

**Step 6: Verify the Cluster**

kubectl get nodes

kubectl get pods --all-namespaces

**Verification**

**Install etcdctl**

sudo apt-get update

sudo apt-get install -y etcd-client

**Verify Etcd Cluster Health**

ETCDCTL\_API=3 etcdctl --endpoints=https://127.0.0.1:2379 --cacert=/etc/kubernetes/pki/etcd/ca.crt --cert=/etc/kubernetes/pki/etcd/peer.crt --key=/etc/kubernetes/pki/etcd/peer.key endpoint health

**Verify HAProxy Configuration**

Edit /etc/haproxy/haproxy.cfg:

listen stats

bind \*:8404

mode http

stats enable

stats uri /

stats refresh 10s

stats admin if LOCALHOST

Restart HAProxy:

sudo systemctl restart haproxy

Check HAProxy Stats at http://<LOAD\_BALANCER\_IP>:8404.

**Test High Availability**

Simulate Master Node Failure:

sudo systemctl stop kubelet

sudo docker stop $(sudo docker ps -q)

Verify Cluster Functionality:

kubectl get nodes

kubectl get pods --all-namespaces

Ensure HAProxy routes traffic to the remaining master node:

curl -k https://<LOAD\_BALANCER\_IP>:6443/version