## Configure Host

sudo blkid /dev/nvme0n1p14

sudo mkfs.ext4 /dev/nvme0n1p14

sudo mkdir -p /data/pv-local

sudo chmod 755 /data/pv-local

sudo mount /dev/nvme0n1p14 /data/pv-local

echo '/dev/nvme0n1p14 /data/pv-local ext4 defaults 0 0' | sudo tee -a /etc/fstab

sudo swapoff -a

sudo sed -i 's|^\(/swap.img\s\+none\s\+swap\s\+sw\s\+0\s\+0\)|#\1|' /etc/fstab

sudo modprobe overlay

sudo modprobe br\_netfilter

sudo sysctl -w net.ipv4.ip\_forward=1

sudo sysctl -w net.bridge.bridge-nf-call-iptables=1

sudo sysctl -w net.bridge.bridge-nf-call-ip6tables=1

echo -e "overlay\nbr\_netfilter" | sudo tee /etc/modules-load.d/k8s.conf

sudo chmod 644 /etc/modules-load.d/k8s.conf

sudo apt install apt-transport-https ca-certificates curl gnupg lsb-release gpg python3-pip -y

sudo mkdir -p /etc/apt/keyrings

sudo chmod 0755 /etc/apt/keyrings

sudo curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.33/deb/Release.key -o /etc/apt/keyrings/kubernetes-Release.key

gpg --dearmor < /etc/apt/keyrings/kubernetes-Release.key | sudo tee /etc/apt/keyrings/kubernetes-apt-keyring.gpg > /dev/null

sudo chmod 0644 /etc/apt/keyrings/kubernetes-apt-keyring.gpg

echo "deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.33/deb/ /" | sudo tee /etc/apt/sources.list.d/kubernetes.list > /dev/null

sudo apt update && sudo apt install -y kubelet kubeadm kubectl containerd

sudo kubectl completion bash > /etc/bash\_completion.d/kubectl

sudo chmod 0644 /etc/bash\_completion.d/kubectl

sudo mkdir -p /etc/containerd

containerd config default | sudo tee /etc/containerd/config.toml > /dev/null

sudo sed -i 's/^.\*SystemdCgroup = false/ SystemdCgroup = true/' /etc/containerd/config.toml

sudo systemctl enable containerd

sudo systemctl restart containerd

## Install K8s

sudo kubeadm init --pod-network-cidr=10.0.0.0/16

# add the nodes

sudo mkdir -p /home/ubuntu/.kube

sudo cp -i /etc/kubernetes/admin.conf /home/ubuntu/.kube/config

sudo chown ubuntu:ubuntu /home/ubuntu/.kube/config

kubectl label node ip-10-10-50-49 node-role.kubernetes.io/worker=

kubectl label node ip-10-10-9-126 node-role.kubernetes.io/worker=

# untaint the node:

kubectl taint nodes ip-10-10-225-40 node-role.kubernetes.io/control-plane-

# Exemplo mínimo (podes adaptar):

kubectl create -f https://raw.githubusercontent.com/cilium/cilium/v1.14/install/kubernetes/quick-install.yaml

CILIUM\_CLI\_VERSION=$(curl -s https://raw.githubusercontent.com/cilium/cilium-cli/main/stable.txt)

CLI\_ARCH=amd64

if [ "$(uname -m)" = "aarch64" ]; then CLI\_ARCH=arm64; fi

curl -L --fail --remote-name-all https://github.com/cilium/cilium-cli/releases/download/${CILIUM\_CLI\_VERSION}/cilium-linux-${CLI\_ARCH}.tar.gz{,.sha256sum}

sha256sum --check cilium-linux-${CLI\_ARCH}.tar.gz.sha256sum

sudo tar xzvfC cilium-linux-${CLI\_ARCH}.tar.gz /usr/local/bin

rm cilium-linux-${CLI\_ARCH}.tar.gz{,.sha256sum}

cilium install

kubectl apply -f https://raw.githubusercontent.com/metallb/metallb/v0.13.10/config/manifests/metallb-native.yaml

# wait

cat <<EOF | kubectl apply -f -

apiVersion: metallb.io/v1beta1

kind: IPAddressPool

metadata:

name: my-pool

namespace: metallb-system

spec:

addresses:

- 10.10.230.240-10.10.230.250

---

apiVersion: metallb.io/v1beta1

kind: L2Advertisement

metadata:

name: l2adv

namespace: metallb-system

EOF

## NGinx

kubectl run nginx --image=nginx

kubectl get pods

# ubuntu@ip-10-10-225-40:~$ kubectl get pods

# NAME READY STATUS RESTARTS AGE

# nginx 1/1 Running 0 8s

scale to 5

kubectl create deployment nginx-deploy --image=nginx --replicas=2

# ubuntu@ip-10-10-225-40:~$ kubectl get pods

# NAME READY STATUS RESTARTS AGE

# nginx 1/1 Running 0 64s

# nginx-deploy-c9d9f6c6c-4lbh8 1/1 Running 0 8s

# nginx-deploy-c9d9f6c6c-qmf5l 1/1 Running 0 8s

kubectl scale deployment nginx-deploy --replicas=5

# ubuntu@ip-10-10-225-40:~$ kubectl get pods

# NAME READY STATUS RESTARTS AGE

# nginx 1/1 Running 0 99s

# nginx-deploy-c9d9f6c6c-4lbh8 1/1 Running 0 43s

# nginx-deploy-c9d9f6c6c-5t78b 1/1 Running 0 2s

# nginx-deploy-c9d9f6c6c-6jgcp 0/1 ContainerCreating 0 2s

# nginx-deploy-c9d9f6c6c-7tqpk 1/1 Running 0 2s

# nginx-deploy-c9d9f6c6c-qmf5l 1/1 Running 0 43s

kubectl get deployments

# NAME READY UP-TO-DATE AVAILABLE AGE

# nginx-deploy 5/5 5 5 77s

# nodeport service for external access

kubectl expose deployment nginx-deploy --type=NodePort --port=80

kubectl get svc

## Config Map

# index.html

echo "<h1>Hello from ConfigMap!</h1>" > index.html

kubectl create configmap custom-index --from-file=index.html

# Pod que monta ConfigMap em nginx

cat <<EOF | kubectl apply -f -

apiVersion: v1

kind: Pod

metadata:

name: nginx-configmap

labels:

app: nginx-configmap

spec:

containers:

- name: nginx

image: nginx

volumeMounts:

- name: html

mountPath: /usr/share/nginx/html

volumes:

- name: html

configMap:

name: custom-index

EOF

kubectl expose pod nginx-configmap --type=NodePort --port=80

kubectl get svc

## Secrets

# secret with password

kubectl create secret generic db-pass --from-literal=password=SuperSecret123

cat <<EOF | kubectl apply -f -

apiVersion: v1

kind: Pod

metadata:

name: secret-check

spec:

containers:

- name: alpine

image: alpine

command: ["/bin/sh", "-c", "echo 'Password:' && cat /etc/secret-volume/password && sleep 3600"]

volumeMounts:

- name: secret-vol

mountPath: /etc/secret-volume

readOnly: true

volumes:

- name: secret-vol

secret:

secretName: db-pass

EOF

kubectl exec -it secret-check -- cat /etc/secret-volume/password

## Persistance

# pvc.yaml

apiVersion: v1

kind: PersistentVolume

metadata:

name: ebs-pv

spec:

capacity:

storage: 1Gi

accessModes:

- ReadWriteOnce

hostPath:

path: "/mnt/data" # caminho do EBS montado

---

apiVersion: v1

kind: PersistentVolumeClaim

metadata:

name: ebs-pvc

spec:

accessModes:

- ReadWriteOnce

resources:

requests:

storage: 1Gi

---

apiVersion: v1

kind: Pod

metadata:

name: pod-ebs

spec:

containers:

- name: busybox

image: busybox

command: ["/bin/sh", "-c", "while true; do echo hello >> /data/out.txt; sleep 10; done"]

volumeMounts:

- mountPath: "/data"

name: ebs-vol

volumes:

- name: ebs-vol

persistentVolumeClaim:

claimName: ebs-pvc

Screenshot 5: kubectl exec e ver conteúdo de /data/out.txt após recriar o pod.

## Namespaces

kubectl create namespace support-team

kubectl create deployment nginx-support --image=nginx -n support-team

kubectl get pods -n support-team