TP 1 KUBERNETES
MISE EN PLACE D'UN CLUSTER K8S

INTRODUCTION

Une entreprise souhaite mettre en place un cluster k8s de développement, et cela dans le but de tester cette nouvelle technologie, votre objectif est de mettre en place un cluster On-premise, vos dispose d'une machine

PARTIE 1: Mise en place de l'environnement

1. Créer des machine virtuel pour simuler un cluster:
    a. Pour cela vous devrez installer virtualbox :

```
sudo apt install virtualbox-6.1
```

il sera utilisé en tant que provider

    b. Il faudra aussi installer vagrant:

```
curl -O
https://releases.hashicorp.com/vagrant/2.2.9/vagrant_2.2.9_x86_64.deb
sudo apt install ./vagrant_2.2.9_x86_64.deb
```

vagrant sera utiliser pour instancier les vm

    c. Créer un Vagrantfile qui instancie les machine suivante :
        1. kmaster : 1go de ram , 1 vcpu ,ubuntu 20.04, ip = 172.42.42.100
        2. kwoker1 : 1go de ram , 1 vcpu ,ubuntu 20.04, ip = 172.42.42.101
        3. kworker2 : 1go de ram , 1 vcpu ,ubuntu 20.04, ip = 172.42.42.102

2. Installation des prérequis sur les machines virtuels :
    a. Connecter vous au machine une par une en utilisant la commande:

```
vagrant ssh nom_de_la_machine
```

une connecte à la machine il faudra installer docker, kubelet, kubeadm, kubectl, apt-transport-https et curl

i. Configurer le fichier /etc/hosts :

```
cat >>/etc/hosts<<EOF
172.42.42.100 kmaster
172.42.42.101 kworker1
172.42.42.102 kworker2
EOF
```

pour faciliter l'utilisation des addres ip

ii. installation de Docker

```
apt-get install apt-transport-https ca-certificates curl
software-properties-common -y
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu
$(lsb_release -cs) stable"
apt-get update -y
apt-get install docker-ce -y
```

Docker permet de lance des conteurs il donc primordiale le l'installer

iii. Activation de Docker

```
usermod -aG docker vagrant
systemctl enable docker >/dev/null 2>&1
systemctl start docker
```

iv. Configuration du sysctl

```
cat >>/etc/sysctl.d/kubernetes.conf<<EOF
net.bridge.bridge-nf-call-ip6tables = 1
net.bridge.bridge-nf-call-iptables = 1
EOF
```

```
sysctl --system >/dev/null 2>&1
```

cela permet la communication via un bridge entre les différents réseaux

v. Désactivation des swap

```
sed -i '/swap/d' /etc/fstab
swapoff -a
```

Kubernetes ne peux pas fonctionner correctement si la partition de swap est ON

vi. Installation de curl et apt-transport-https

```
apt-get update && apt-get install -y apt-transport-https curl
curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | apt-key add -
```

apt-transport-https va permettre de sécuriser les comunications au sain du cluster

vii. Installation de Kubernetes

```
cat <<EOF | sudo tee /etc/apt/sources.list.d/kubernetes.list
deb https://apt.kubernetes.io/ kubernetes-xenial main
EOF
ls -ltr /etc/apt/sources.list.d/kubernetes.list
apt-get update -y
apt-get install -y kubelet kubeadm kubectl
```

nous avons installer les tout les packages nécessaire au lancement de K8S

viii. Activation du kubelet

```
systemctl enable kubelet >/dev/null 2>&1
systemctl start kubelet >/dev/null 2>&1
```

3. Crée un script bash bootstrap.sh qui aura pour but d'installer et de configurer tout les machine en l'ajoutant au vagrantfile cela vous évitera de refaire tout ces opération une par une sur tout les machines

PARTIE 2: Initialisation du cluster

1. Initialisation de master :

Pour cela utilise la commande kubeadm init :

```
kubeadm init --apiserver-advertise-address=172.42.42.100
--pod-network-cidr=192.168.0.0/16 >> /root/kubeinit.log 2>/dev/null
```

Cela va initialiser le master ducluster dont l'addres est 172.42.42.100 et indique l'emsembre des adresses 192.168.0.0/16 selon si on utilise calico ou flannel comme pod-network dans notre cas en utilise calico.

a. Copier la configuration de l'admin kube dans le répertoire .kube du user Vagrant :

```
mkdir /home/vagrant/.kube
cp /etc/kubernetes/admin.conf /home/vagrant/.kube/config
chown -R vagrant:vagrant /home/vagrant/.kube
```

Cela va permettre de communiquer avec le cluster.

b. Déploiement du réseau Calico

```
su - vagrant -c "kubectl create -f
https://docs.projectcalico.org/v3.9/manifests/calico.yaml"
```

On crée le réseau en utilisant un manifest du réseau et utilisant le kubectl

c. génération de la commande join :

```
kubeadm token create --print-join-command
```

En lance cette commande kubeadm qui retourne une autre commande qu'il faudra exécuter sur les autres VM pour les ajouter au cluster

2. Ajout des workers au cluster

Il faudra reproduire ces opérations sur chaque node:

Ajout du worker au cluster il faudra exécuter la commande obtenue suite celle ci

```
kubeadm token create --print-join-command
```

une fois nos workers ajouter nous orant un cluster opérationnel

3. Vérification de l'état du cluster :

```
kubectl cluster-info
```

4. Crée deux script bash bootstrap_kmaster.sh et bootstrap_kmaster.sh qui lance le cluster automatiquement.

La correction de ce TP se trouve sur le repo git : https://github.com/AnisBarr/TP1_k8s