Prácticas Kubernetes

Instalar un cluster con Kubeadm

1. Preparar los servidores.

Características

- 3 máquinas virtuales con al menos 2,5G de RAM y 20HDD de espacio en disco
- Debemos configurarla con al menos 2 procesadores, de lo contrario no funciona la instalación

Preparación inicial del servidor

- Debemos trabajar como root
- En primer lugar debemos deshabilitar el **SWAP**. De lo contrario no funciona el servidor
- para comprobar las áreas de su app que tienes en tu sistema puedes utilizar el comando

```
swapon -s
```

- Podemos deshabilitarlo con el comando

```
swapoff -a
```

- Sin embargo, debemos deshabilitarlo del sistema para que no se active al rebotar el servidor
- Para ello modificamos el fichero /etc/fstab.
- Debemos comentar la línea donde aparece el swap
- Debería ser similar a la siguiente. Hay que tener en cuenta que puedes tener más de un área de swap y por lo tanto tienes que deshabilitar las todas

```
# /etc/fstab: static file system information.
#
# Use 'blkid' to print the universally unique identifier for a
# device; this may be used with UUID= as a more robust way to name devices
# that works even if disks are added and removed. See fstab(5).
#
```

```
# <file system> <mount point>   <type>  <options>       <dump>  <pass>
# / was on /dev/sda1 during installation
UUID=e51dac01-e63f-4cb9-b10d-6b9d7b07a53b   /                       ext4
errors=remount-ro 0       1
/swapfile                 none            swap    sw              0       0
```

- También debemos asegurarnos de que se pueda acceder por el puerto 6443.
- Debemos también instalar algún container runtime para que pueda ser utilizado por el clúster de kubernetes.
- podéis utilizar por supuesto cualquiera aunque las recomendaciones serían las siguientes:
    - containerd
    - CRI-O
    - Docker Engine
- Debemos conectarnos como “root” o bien como usuario que pueda hacer “sudo”.

Instalar Kubectl, Kubeadm y Kubelet con un gestor de paquetes. Ejemplo con Centos/RedHat

- En primer lugar configuramos el repositorio de Kubernetes:

```
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
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
```

- Ahora desactivamos SELinux, que es una funcionalidad de los Linux que añade una capa de seguridad más estricta al sistema. kubernetes no funciona si está activada

```
# Set SELinux in permissive mode (effectively disabling it)
```

```
sudo setenforce 0
sudo sed -i 's/^SELINUX=enforcing$/SELINUX=permissive/' /etc/selinux/config
```

- Instalamos los 3 componentes necesarios: kubeadm, kubelet, y kubectl:

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

- Solo queda arrancar el agente Kubelet

```
sudo systemctl enable --now kubelet
```

Instalar Kubectl, Kubeadm y Kubelet con un gestor de paquetes. Ejemplo con Debian/Ubuntu

- Actualizar el índice de paquetes apt e instalar los paquetes necesarios para usar el repositorio apt de Kubernetes:

```
sudo apt-get update
sudo apt-get install -y apt-transport-https ca-certificates curl
```

- Descargar la clave de firma pública de Google Cloud

```
sudo curl -fsSLo /usr/share/keyrings/kubernetes-archive-keyring.gpg https://packages.cloud.google.com/apt/doc/apt-key.gpg
```

- Instalar repositorio de Kubernetes

```
echo "deb [signed-by=/usr/share/keyrings/kubernetes-archive-keyring.gpg] https://apt.kubernetes.io/ kubernetes-xenial main" | sudo tee /etc/apt/sources.list.d/kubernetes.list
```

- Instalamos los 3 componentes necesarios: kubeadm, kubelet, y kubectl:

```
sudo apt-get update
sudo apt-get install -y kubelet kubeadm kubectl
sudo apt-mark hold kubelet kubeadm kubectl
```

Crear el cluster

- Creamos el cluster con una configuración que depende del tipo de plugin Network que hayamos seleccionado
- Por ejemplo, si queremos usar una POD Network Calico, necesitamos añadir el parámetro –pod-network-cidr
- Por ejemplo. es importante comprobar que el direccionamiento IP que le estamos poniendo al clúster no coincide con alguno de los que tengamos dentro de la red local

```
kubeadm init --pod-network-cidr=192.168.0.0/16
```

- Abrimos una nueva pestaña o un nuevo terminal y nos conectamos como el usuario con el que vamos a trabajar, en este caso sin ser root
- Ejecutamos los siguientes comandos para cargar la configuración

```
mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config
```

- Instalamos el operador tigera calico (los operadores es un tema que veremos posteriormente pero son componentes que ayudan a la instalación de productos complejos)

```
kubectl create -f https://raw.githubusercontent.com/projectcalico/calico/v3.24.0/manifests/tigera-operator.yaml
```

- Instalamos el plugin de calico. Como puedes observar se trata de un YAML que genera los PODS necesarios

```
kubectl create -f https://raw.githubusercontent.com/projectcalico/calico/v3.24.0/manifests/custom-resources.yaml
```

- Comprobamos que los pods de Calico están ejecutándose

```
watch kubectl get pods -n calico-system
```

- Hacemos un “untaint” del nodo para que puede realizar scheduling de los workloads

```
kubectl taint nodes --all node-role.kubernetes.io/control-plane- node-role.kubernetes.io/master-
```

- Probamos el cluster

```
kubectl get nodes
```

Añadir nodos al cluster

- Nos conectamos al nodo que queremos incorporar al cluster. Es importante que también tenga el software instalado
- Ejecutamos el join que se ha indicado en el momento de hacer el “init”.
- Por ejemplo (el vuestro será distinto evidentemente)

```
kubeadm join 192.168.1.101:6443 --token tokentoken.lalalalaqyd3kavez --discovery-token-ca-cert-hash sha256:complexshaoverhere
```

- Probamos que el cluster tiene los nodos añadidos

```
kubectl get nodes
```