Prácticas Kubernetes

Instalar un cluster con Kubeadm

1. Preparar los servidores. Ubuntu 18.04

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
- 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

```
# /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
```

- Ahora configurarmos iptables para poder recibir tráfico de tipo bridge.

- Para ello editamos el fichero “sysctl.conf”
- Añadimos las siguientes líneas

```
net/bridge/bridge-nf-call-ip6tables = 1
net/bridge/bridge-nf-call-iptables = 1
net/bridge/bridge-nf-call-arptables = 1
```

- Luego debemos instalar algunos paquetes (si no los tenemos ya en el sistema).

```
apt-get install ebtables ethtool
```

Instalar Docker, Kubectl, Kubeadm y Kubelet

- Instalamos docker Docker. Primero hacemos un apt-get update para actualizar el sistema

```
apt-get update
apt-get install -y docker.io
```

- Comprobamos que se ha instalado y la versión

```
systemctl Docker status
docker version
```

- Instalamos soporte para HTTPS

```
apt-get update
apt-get install -y apt-transport-https
```

- Instalamos “curl” si no lo tenemos instalado

```
apt-get install curl
```

- Recuperamos la clave para el repositorio de Kubernetes y lo añadimos con apt-key:

```
curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | apt-key add -
```

- Añadimos el repositorio a nuestro sistes

```
cat <<EOF >/etc/apt/sources.list.d/kubernetes.list
deb http://apt.kubernetes.io/ kubernetes-xenial main
EOF
```

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

```
apt-get update
apt-get install -y kubelet kubeadm kubect
```

Crear el cluster

- Creamos el cluster. Por ejemplo, si queremos usar una POD Network Calico, necesitamos añadir el parámetro –pod-network-cidr switch
- Por ejemplo

```
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 mi caso “kubernetes”
- 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 plugin de calico

```
kubectl apply -f https://docs.projectcalico.org/v3.11/manifests/calico.yaml
```

- Comprobamos que los pods del Sistema están ejecutándose

```
kubectl get pods --all-namespaces
```

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

```
kubectl taint nodes --all 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
```