

[docker](#), [kubernetes](#), [centos](#), [instalar](#)

## Instalación de Kubernetes en Centos 7

La Wikipedida define [Kubernetes](#) como (referido en inglés comúnmente como “K8s”) **un sistema de código libre para la automatización del despliegue, ajuste de escala y manejo de aplicaciones en contenedores** .

A esta clase de software se la conoce como orquestadores, existen varios y cada uno tienen sus propias [características](#)

Vamos a crear un cluster con al menos tres nodos , 1 manager y 2 workers (utilizar siempre un número impar de nodos) con unos requisitos mínimos de 2vCPUs y 2 GB de memoria por cada nodo.

Además necesitaremos conectividad de red entre todos los nodos

### Pasos previos

Vamos a realizar los siguientes pasos tanto en el manager como en el resto de nodos Lo primero será deshabilitar la swap

- temporalmente (hasta que reiniciemos )con el comando **swapoff -a**
- Definitivamente editando el fichero `/etc/fstab` y comentando la línea de la partición del swap
- Permitimos a iptable ver el tráfico en modo bridge . Para ello tenemos que cargar el módulo

```
modprobe br_netfilter
```

y activarlo al arrancar con

```
echo '1' > /proc/sys/net/bridge/bridge-nf-call-iptables
```

- creamos un fichero de configuración para que se cargue en `/etc/sysctl.d` incluido el manager

```
cat <<EOF > /etc/sysctl.d/k8s.conf
net.bridge.bridge-nf-call-ip6tables = 1
net.bridge.bridge-nf-call-iptables = 1
EOF
```

y aplicamos los cambios con

```
sysctl --system
```

- deshabilitamos [SELinux](#)
- instalamos los requisitos para instalar docker

```
yum install -y yum-utils device-mapper-persistent-data lvm2
```

- añadimos el repositorio e instalamos docker

```
yum-config-manager --add-repo  
https://download.docker.com/linux/centos/docker-ce.repo
```

```
sudo yum update -y && sudo yum install -y containerd.io-1.2.13 docker-  
ce-19.03.11 docker-ce-cli-19.03.11
```



A día de este tutorial estas son las versiones de docker-ce soportadas por kubernetes

- Lo ponemos para que arranque al inicio

```
systemctl enable docker && systemctl start docker
```

## Instalación de Kubernetes

### Pasos a realizar en todos los nodos (incluido el manager)

```
cat <<EOF > /etc/yum.repos.d/kubernetes.repo  
[kubernetes]  
name=Kubernetes  
baseurl=https://packages.cloud.google.com/yum/repos/kubernetes-el7-x86_64  
enabled=1  
gpgcheck=1  
repo_gpgcheck=1  
gpgkey=https://packages.cloud.google.com/yum/doc/yum-key.gpg  
https://packages.cloud.google.com/yum/doc/rpm-package-key.gpg  
EOF
```

Instalamos los paquetes

```
yum update  
yum install -y kubelet kubeadm kubectl
```

Lo ponemos para que arranque al inicio

```
systemctl enable kubelet && systemctl start kubelet
```

Comprobamos que kubernetes y docker están en el mismo grupo de control ( cgroup)

Para verificar el cgroup de docker

```
docker info | grep -i cgroup
```

Para añadir kubernetes al mismo cgroup

```
sed -i 's/cgroup-driver=systemd/cgroup-driver=cgroupfs/g' /etc/systemd/system/kubelet.service.d/10-kubeadm.conf
```



según la versión de kubernetes y/o SO el fichero de configuración se ha cambiado a /usr/lib/systemd/system/kubelet.service.d/10-kubeadm.conf

```
sed -i 's/cgroup-driver=systemd/cgroup-driver=cgroupfs/g' /usr/lib/systemd/system/kubelet.service.d/10-kubeadm.conf
```

Reiniciamos los servicios

```
systemctl daemon-reload  
systemctl restart kubelet
```

### Pasos a realizar sólo en el Manager

Como el Firewalld está habilitado en CentOS de manera predeterminada, tenemos que abrir los siguientes puertos para permitir la comunicación con los nodos

```
firewall-cmd --permanent --add-port=6443/tcp  
firewall-cmd --permanent --add-port=2379-2380/tcp  
firewall-cmd --permanent --add-port=10250/tcp  
firewall-cmd --permanent --add-port=10251/tcp  
firewall-cmd --permanent --add-port=10252/tcp  
firewall-cmd --permanent --add-port=10255/tcp  
firewall-cmd --reload
```

Iniciamos el cluster

```
kubeadm init
```



hay varias [opciones](#) que podemos usar para definir el interfaz que da servicio o la red para los pods. Por ejemplo `kubeadm init --apiserver-advertise-address=192.168.1.99 --pod-network-cidr=192.168.0.0/16`

Cuando este comando termina nos aparecerá un comando con el token y el hash para unir los nodos. Este comando deberemos de guardarlos para añadir los nodos.

Antes de usar Kubernetes deberemos de ejecutar los siguientes comandos para terminar la configuración.

- Si kubernetes lo vamos a lanzar como root sólo debemos ejecutar el siguiente comando

```
export KUBECONFIG=/etc/kubernetes/admin.conf
```

- Si lo vamos a lanzar como otro usuario deberemos de crear un directorio para la configuración, copiar los archivos necesarios y darle permisos sobre dichos archivos

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

Ahora debemos de definir el modelo de red que queremos usar en kubernetes , hay varias aproximaciones cada una con distitas características  
(<https://kubernetes.io/docs/concepts/cluster-administration/networking/>)

Si por ejemplo vamos a usar [flannel](#) ejecutaríamos en el manager

```
kubectl apply -f  
https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-flannel.yml
```

Si usamos [weave](#) el comando sería

```
kubectl apply -f "https://cloud.weave.works/k8s/net?k8s-version=$(kubectl version | base64 | tr -d '\n')"
```

Si queremos comprobar si se ha instalado correctamente podemos ejecutar

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

nos deberías aparecer un pods llamada weave-net-wvlbx con el estado **running**

### **Paso final en los nodos**

Abrimos los puertos para la comunicación con el resto de nodos

```
firewall-cmd --permanent --add-port=10251/tcp  
firewall-cmd --permanent --add-port=10255/tcp  
firewall-cmd --reload
```

Como último punto tenemos que añadir los nodos al manager. Para ello usaremos el comando que habíamos guardado cuando iniciamos el cluster y lo ejecutamos en los nodos que vamos a unir al cluster

```
kubeadm join ipmanager:6443 --token MITOKEN --discovery-token-ca-cert-hash MIDISCOVERY_TOKEN
```



MITOKEN y MIDISCOVERY\_TOKEN son los que nos dio el manager al iniciarlo

## Funcionamiento

Para ver que todo está funcionando y los nodos están registrados ejecutamos en el manager

```
kubectl get nodes
```

## Problemas

### Cambiar el cgroup driver de Docker a systemd

Editar el fichero `/usr/lib/systemd/system/docker.service` y cambiar la línea

```
ExecStart=/usr/bin/dockerd \  
--exec-opt native.cgroupdriver=systemd
```

## Referencias

- <https://kubernetes.io/docs/setup/production-environment/tools/kubeadm/install-kubeadm/>
- <https://www.techrepublic.com/article/how-to-deploy-nginx-on-a-kubernetes-cluster/>
- <https://www.techrepublic.com/article/how-to-install-a-kubernetes-cluster-on-centos-7/>

From:  
<https://intrusos.info/> - LCWIKI

Permanent link:  
<https://intrusos.info/doku.php?id=virtualizacion:kubernetes:instalacion&rev=1614165859>

Last update: **2023/01/18 14:22**

