# COMO INSTALAR UN CLUSTER KUBERNETES

¿Está buscando una guía sencilla sobre cómo instalar Kubernetes Cluster en Ubuntu 22.04 (Jammy Jellyfish)?

La guía paso a paso en esta página le mostrará cómo instalar el clúster de Kubernetes en Ubuntu 22.04 usando el comando Kubeadm paso a paso.

Kubernetes es una herramienta de orquestación de contenedores gratuita y de código abierto, también conocida como k8s. Con la ayuda de Kubernetes, podemos lograr la implementación, el escalado y la gestión automatizados de aplicaciones en contenedores.

Un clúster de Kubernetes consta de nodos de trabajo en los que se implementa la carga de trabajo de la aplicación y nodos maestros configurados que se utilizan para administrar los nodos de trabajo y los pods en el clúster.

En esta guía, usamos un nodo maestro y dos nodos trabajadores. Los siguientes son los requisitos del sistema en cada nodo,

### Instalación mínima Ubuntu 22.04

Mínimo 2 GB de RAM o más

Mínimo 2 núcleos de CPU o 2 vCPU

20 GB de espacio libre en disco en /var o más

Usuario Sudo con derechos de administrador

Conectividad a Internet en cada nodo

### Configuración de laboratorio

Nodo maestro: 192.168.1.xxx – k8smaster.example.net

Primer nodo trabajador: 192.168.1.xxx – k8sworker1.example.net

Segundo nodo trabajador: 192.168.1.xxx – k8sworker2.example.net

Donde xxx es el número del cuarto octeto de la Ip de vuestras máquinas.

# Paso 1) Establecer nombre de host y agregar entradas en el archivo de hosts

Inicie sesión en el nodo maestro y configure el nombre de host usando el comando hostnamectl,

$ sudo hostnamectl set-hostname "k8smaster.example.net"

$ exec bash

En los nodos worker1 y worker2:

$ sudo hostnamectl set-hostname "k8sworker1.example.net"

$ sudo hostnamectl set-hostname "k8sworker2.example.net

$ exec bash

Agregue las siguientes entradas en el archivo ***/etc/hosts*** en cada nodo, suponiendo como ejemplo, que master es el 173, worker1 es el 174 y worker2 es el 175:

192.168.1.173   k8smaster.example.net k8smaster

192.168.1.174   k8sworker1.example.net k8sworker1

192.168.1.175   k8sworker2.example.net k8sworker2

# paso 2) Deshabilitar el intercambio y agregar la configuración del kernel

Ejecute debajo del comando swapoff y sed para deshabilitar el intercambio. Asegúrese de ejecutar los siguientes comandos en todos los nodos.

$ sudo swapoff -a

$ sudo sed -i '/ swap / s/^\(.\*\)$/#\1/g' /etc/fstab

Cargue los siguientes módulos del kernel en todos los nodos,

$ sudo tee /etc/modules-load.d/containerd.conf <<EOF

overlay

br\_netfilter

EOF

$ sudo modprobe overlay

$ sudo modprobe br\_netfilter

Establezca los siguientes parámetros del Kernel para Kubernetes, ejecute debajo del comando tee:

$sudo tee /etc/sysctl.d/kubernetes.conf <<EOF

net.bridge.bridge-nf-call-ip6tables = 1

net.bridge.bridge-nf-call-iptables = 1

net.ipv4.ip\_forward = 1

EOF

Vuelva a cargar los cambios anteriores, ejecute

$ sudo sysctl --system

# Paso 3) Instale el tiempo de ejecución del contenedor

En esta guía, usamos tiempo de ejecución de contenedores para nuestro clúster de Kubernetes. Entonces, para instalar containerd, primero instale sus dependencias.

$ sudo apt install -y curl gnupg2 software-properties-common apt-transport-https ca-certificates

Habilitar repositorio docker

$ sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmour -o /etc/apt/trusted.gpg.d/docker.gpg

$ sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb\_release -cs) stable"

Ahora, ejecute el siguiente comando apt para instalar containerd

$ sudo apt update

$ sudo apt install -y containerd.io

Configure containerd para que comience a usar systemd como cgroup.

$ containerd config default | sudo tee /etc/containerd/config.toml >/dev/null 2>&1

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

Reinicie el servicio containerd y habilite el servicio de contenedores

$ sudo systemctl restart containerd

$ sudo systemctl enable containerd

# Paso 4) Agregue el repositorio apt para Kubernetes

Ejecute los siguientes comandos para agregar un repositorio apt para Kubernetes

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

$ sudo apt-add-repository "deb http://apt.kubernetes.io/ kubernetes-xenial main"

**Nota**: Al momento de escribir esta guía, Xenial es el repositorio de Kubernetes más reciente, pero cuando el repositorio está disponible para Ubuntu 22.04 (Jammy Jellyfish), debe reemplazar la palabra xenial con "jammy" en el comando "apt-add-repository"

# Paso 5) Instale los componentes de Kubernetes Kubectl, kubeadm y kubelet

Instale componentes de Kubernetes como kubectl, kubelet y la utilidad Kubeadm en todos los nodos. Ejecute el siguiente conjunto de comandos,

$ sudo apt update

$ sudo apt install -y kubelet kubeadm kubectl

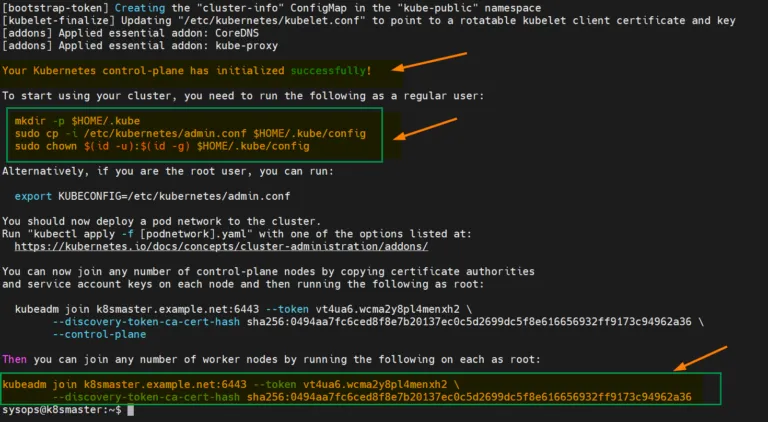
$ sudo apt-mark hold kubelet kubeadm kubectl

# Paso 6) Inicialice el clúster de Kubernetes con el comando Kubeadm

Ahora, estamos listos para inicializar el clúster de Kubernetes. Ejecute el siguiente comando de Kubeadm solo desde el nodo principal.

$ sudo kubeadm init --control-plane-endpoint=k8smaster.example.net

Salida del comando anterior,



Como la salida anterior confirma que el plano de control se ha inicializado con éxito. En la salida también obtenemos un conjunto de comandos para interactuar con el clúster y también el comando para que el nodo trabajador se una al clúster.

Entonces, para comenzar a interactuar con el clúster, ejecute los siguientes comandos desde el nodo maestro,

$ mkdir -p $HOME/.kube

$ sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config

$ sudo chown $(id -u):$(id -g) $HOME/.kube/config

Ahora, intente ejecutar en el master los siguientes comandos de kubectl para ver el estado del clúster y del nodo

$ kubectl cluster-info

$ kubectl get nodes

Úna ambos nodos trabajadores al clúster, el comando ya está allí, simplemente copie y pegue en los nodos trabajadores,

$ sudo kubeadm join k8smaster.example.net:6443 --token vt4ua6.wcma2y8pl4menxh2 \

--discovery-token-ca-cert-hash sha256:0494aa7fc6ced8f8e7b20137ec0c5d2699dc5f8e616656932ff9173c94962a36

**OJO** el discovery token es unico para cada master y hay que tomarlo en el momento de iniciar kubeadm.

Verifique el estado de los nodos desde el nodo maestro usando el comando kubectl,

$ kubectl get nodes

Como podemos ver, el estado de los nodos es 'No listo', así que para activarlo. Debemos instalar CNI (Container Network Interface) o complementos de red como Calico, Flannel y Weave-net.

# Paso 7) Instale el complemento Calico Pod Network

Ejecute los siguientes comandos curl y kubectl para instalar el complemento de red Calico desde el nodo maestro,

$ curl https://projectcalico.docs.tigera.io/manifests/calico.yaml -O

$ kubectl apply -f calico.yaml

Verifique el estado de los pods en el espacio de nombres del sistema kube,

$ kubectl get pods -n kube-system

Perfecto, revisa también el estado de los nodos, deberán aparecer como ready.

$ kubectl get nodes

Genial, lo anterior confirma que los nodos son nodos activos. Ahora, podemos decir que nuestro clúster de Kubernetes es funcional.

# Paso 8) Pruebe la instalación de Kubernetes

Para probar la instalación de Kubernetes, intentemos implementar una aplicación basada en nginx e intentemos acceder a ella.

$ kubectl create deployment nginx-app --image=nginx --replicas=2

$ kubectl get deployment nginx-app

NAME        READY   UP-TO-DATE   AVAILABLE   AGE

nginx-app   2/2     2            2           68s

$

Exponer la implementación como NodePort,

$ kubectl expose deployment nginx-app --type=NodePort --port=80

service/nginx-app exposed

$

Ejecute los siguientes comandos para ver el estado del servicio

$ kubectl get svc nginx-app

$ kubectl describe svc nginx-app

Use el siguiente comando para acceder a la aplicación basada en nginx,

$ curl http://<woker-node-ip-addres>:31246

NOTA: El token dura 24 horas. Es importante copiar la línea kubeadm join completa y guardarla. Caso de unir un worker pasadas 24 horas, tendríamos que generar un nuevo token y un nuevo hash ca cert Discovery asociada a ese token.

Podemos ver la lista de tokens con l comando kubeadm token list

Podemos crear un nuevo token con el comando kubeadm token créate

Podemos crear un nuevo token con la opción --print-join-command