Kubernetes cluster installation guide

This is an installation guide for a Kubernetes cluster; this tutorial was tested using Ubuntu 24.04. operating system.

The process documents the configuration of the main node (Control Plane or Master) using containerd as the container runtime and Calico as the network provider (CNI). Upon completing this guide, the cluster will be operational and ready to receive workloads or to join additional worker nodes.

The guide must be followed and execute points 1-4 in all nodes. Points 5, 6 and 8 must be followed only in the master node.

1 - Getting the server ready - All nodes

```
sudo apt-get update
sudo apt-get install -y apt-transport-https ca-certificates curl gpg
sudo mkdir -p /etc/apt/keyrings
curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.32/deb/Release.key | sudo gpg --dearmor -o
/etc/apt/keyrings/kubernetes-apt-keyring.gpg
echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.32/deb/
/' | sudo tee /etc/apt/sources.list.d/kubernetes.list
sudo apt update
sudo apt install -y kubelet kubeadm kubectl
sudo apt-mark hold kubelet kubeadm kubectl
```

2 - Disable swap memory - All nodes

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

3 - Configure network and modules - All nodes

3.1 Configure persistent loading of modules

```
sudo tee /etc/modules-load.d/containerd.conf <<EOF
overlay
br_netfilter
EOF
```

3.2 Load at runtime

```
sudo modprobe overlay
sudo modprobe br_netfilter
```

3.3 Ensure sysctl params are set

```
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
```

3.4 Reload configs

```
sudo sysctl --system
```

4 - Install and configure Containerd - All nodes

4.1 Add docker repository

```
sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg -o /etc/apt/keyrings/docker.asc
sudo chmod a+r /etc/apt/keyrings/docker.asc
echo "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.asc]
https://download.docker.com/linux/ubuntu $(. /etc/os-release && echo "$VERSION_CODENAME") stable" | sudo
tee /etc/apt/sources.list.d/docker.list > /dev/null
```

4.2 Install containerd

```
sudo apt update
sudo apt install -y containerd.io
```

4.3 Configure containerd

```
sudo containerd config default | sudo tee /etc/containerd/config.toml
sudo sed -i 's/SystemdCgroup = false/SystemdCgroup = true/' /etc/containerd/config.toml
```

4.4 Start/restart containerd

```
sudo systemctl restart containerd
sudo systemctl enable containerd
```

4.5 Enable kubelet

```
sudo systemctl enable kubelet
```

5 - Master node configuration - Only master

5.1 Enable and start kubelet

```
sudo kubeadm config images pull
sudo kubeadm init --pod-network-cidr=192.168.0.0/16 --control-plane-endpoint=<IP or DNSNAME> --upload-certs
```

Take in mind that if you use the DNSNAME option, you have to configure a DNS or /etc/hosts

```
sudo kubeadm init --pod-network-cidr=192.168.0.0/16 --control-plane-endpoint=<192.168.122.40/16> --upload-certs


sudo kubeadm init --pod-network-cidr=192.168.0.0/16 --control-plane-endpoint=<soffid-Ubuntu-24-04-PC-Q35-ICH9-2009> --upload-certs
```

Note: If *192.168.0.0/16* is already in use within your network you must select a different pod network CIDR, replacing 192.168.0.0/16 in the above command.

Other options of the init command, but we are not going to use right now are:

--cri-socket ==> Use if have more than one container runtime to set runtime socket path
--apiserver-advertise-address ==> Set advertise address for this particular control-plane node's API server

If everything is ok, you will able to read a message saying "Your Kubernetes control-plane has initialized successfully!". If so, you need to run the following:

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

6 - Installing Calico - Only master

6.1 Installing Calico

```
kubectl apply -f https://raw.githubusercontent.com/projectcalico/calico/v3.28.0/manifests/calico.yaml
```

6.2 Check kubernetes nodes

```
kubectl get nodes
```

7 - Joining WORKER nodes

7.1 For joining other nodes execute the following command - Only MASTER

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

7.2 Example of join token - Only SLAVE

```
kubeadm join 192.168.122.40:6443 --token r705ed.7d47ya6a6f3eecr4 --discovery-token-ca-cert-hash
sha256:0c740869ea8a0e833a0163c5246fb74daed12baf7207e8a4a7f37dda40aaee4e
```

Once joined a message like:

This node has joined the cluster:
* Certificate signing request was sent to apiserver and a response was received.
* The Kubelet was informed of the new secure connection details.

Run 'kubectl get nodes' on the control-plane to see this node join the cluster.

Will appear.

7.3 Check if the node was added - ONLY MASTER

```
kubectl get nodes
NAME                                  STATUS   ROLES          AGE   VERSION
soffid-ubuntu-24-04-pc-q35-ich9-2009       Ready    control-plane   67m   v1.32.13
soffidslave-ubuntu-24-04-pc-q35-ich9-2009   Ready    <none>         71s   v1.32.13
```

8 - Joining other MASTER nodes

Take in mind to have and odd number of master nodes.

```
sudo kubeadm init phase upload-certs --upload-certs
```

This will return a TOKEN:
2a74cf543fcbe46e8e0c9938d65e990f401573df2815945e320946fbe0b638a6

```
sudo kubeadm token create --print-join-command --certificate-key TOKEN


sudo kubeadm token create --print-join-command --certificate-key
2a74cf543fcbe46e8e0c9938d65e990f401573df2815945e320946fbe0b638a6
```

This will return a command like:

```
kubeadm join 192.168.122.40:6443 --token bdvw7e.pceldjy93tb42ggu --discovery-token-ca-cert-hash
sha256:6f4528b9625a9291d286b1525e37e45a9394c9ee98e4e035a7319ac356cebb1b --control-plane --
certificate-key 2a74cf543fcbe46e8e0c9938d65e990f401573df2815945e320946fbe0b638a6
```

Execute it on the new NODE to connect it as a MASTER node

Once done get the authorizations

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

Check the nodes

```
kubectl get nodes
NAME                                  STATUS   ROLES          AGE   VERSION
soffid-ubuntu-24-04-pc-q35-ich9-2009       Ready    control-plane   10m   v1.32.13
soffidslave-ubuntu-24-04-pc-q35-ich9-2009   Ready    control-plane   38s   v1.32.13
```

9 - Untaint master node - Only master

9.1 Check node name

```
kubectl get nodes


NAME                              STATUS   ROLES          AGE   VERSION
soffid-ubuntu-24-04-pc-q35-ich9-2009   Ready    control-plane   15m   v1.32.13
```

9.2 Enable the master node to execute pods

```
kubectl taint node NODE_NAME node-role.kubernetes.io/master:NoSchedule-
```

-Example-

kubectl taint node soffid-ubuntu-24-04-pc-q35-ich9-2009 node-role.kubernetes.io/control-plane:NoSchedule-

9.3 Check master node taints

kubectl describe node NODE_NAME | grep Taints

-Example-

kubectl describe node soffid-ubuntu-24-04-pc-q35-ich9-2009 | grep Taints

---

Revision #6
Created 11 August 2026 08:06:59 by Sebastià Oliver
Updated 11 August 2026 10:58:07 by Sebastià Oliver