# **Kubernetes Cluster Setup**

This document provides step-by-step instructions to set up a Kubernetes cluster with a master node and worker nodes. The setup includes configuring the necessary prerequisites, installing the required tools, and initializing the cluster.

## **Master Node Setup**

### **1. Disable Swap**

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

### **2. Enable Bridging and IP Forwarding**

**Load required kernel modules:**

cat <<EOF | sudo tee /etc/modules-load.d/k8s.conf  
overlay  
br\_netfilter  
EOF  
  
sudo modprobe overlay  
sudo modprobe br\_netfilter

**Configure sysctl parameters:**

cat <<EOF | sudo tee /etc/sysctl.d/k8s.conf  
net.bridge.bridge-nf-call-iptables = 1  
net.bridge.bridge-nf-call-ip6tables = 1  
net.ipv4.ip\_forward = 1  
EOF  
  
sudo sysctl --system

**Verify modules and parameters:**

lsmod | grep br\_netfilter  
lsmod | grep overlay  
sysctl net.bridge.bridge-nf-call-iptables net.bridge.bridge-nf-call-ip6tables net.ipv4.ip\_forward

### **3. Install Container Runtime (Containerd)**

**Download and install Containerd:**

curl -LO <https://github.com/containerd/containerd/releases/download/v1.7.14/containerd-1.7.14-linux-amd64.tar.gz>sudo tar Cxzvf /usr/local containerd-1.7.14-linux-amd64.tar.gz  
curl -LO <https://raw.githubusercontent.com/containerd/containerd/main/containerd.service>sudo mkdir -p /usr/local/lib/systemd/system/  
sudo mv containerd.service /usr/local/lib/systemd/system/

**Configure Containerd:**

sudo mkdir -p /etc/containerd  
containerd config default | sudo tee /etc/containerd/config.toml  
sudo sed -i 's/SystemdCgroup \= false/SystemdCgroup \= true/g' /etc/containerd/config.toml

**Start and enable the Containerd service:**

sudo systemctl daemon-reload  
sudo systemctl enable --now containerd

**Verify the service:**

systemctl status containerd

### **4. Install runc and CNI Plugins**

**Install runc:**

curl -LO <https://github.com/opencontainers/runc/releases/download/v1.1.12/runc.amd64>sudo install -m 755 runc.amd64 /usr/local/sbin/runc

**Install CNI plugins:**

curl -LO <https://github.com/containernetworking/plugins/releases/download/v1.5.0/cni-plugins-linux-amd64-v1.5.0.tgz>sudo mkdir -p /opt/cni/bin  
sudo tar Cxzvf /opt/cni/bin cni-plugins-linux-amd64-v1.5.0.tgz

### **5. Install Kubernetes Components**

**Add Kubernetes repository and GPG key:**

sudo apt-get update  
sudo apt-get install -y apt-transport-https ca-certificates curl gpg  
  
curl -fsSL <https://pkgs.k8s.io/core:/stable:/v1.29/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:/v1.29/deb/> /' | sudo tee /etc/apt/sources.list.d/kubernetes.list

**Install kubeadm, kubelet, and kubectl:**

sudo apt-get update  
sudo apt-get install -y kubelet=1.29.6-1.1 kubeadm=1.29.6-1.1 kubectl=1.29.6-1.1 --allow-downgrades --allow-change-held-packages  
sudo apt-mark hold kubelet kubeadm kubectl

**Verify installation:**

kubeadm version  
kubelet --version  
kubectl version --client

### **6. Configure crictl**

sudo crictl config runtime-endpoint unix:///var/run/containerd/containerd.sock

### **7. Initialize Control Plane**

**Run the following command to initialize the control plane:**

sudo kubeadm init --pod-network-cidr=192.168.0.0/16 --apiserver-advertise-address=<private\_ip\_of\_control\_plane> --node-name master

**Note:** Save the output join command for later use.

### **8. Configure kubectl**

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

### **9. Install Calico Network Plugin**

kubectl create -f <https://raw.githubusercontent.com/projectcalico/calico/v3.28.0/manifests/tigera-operator.yaml>curl <https://raw.githubusercontent.com/projectcalico/calico/v3.28.0/manifests/custom-resources.yaml> -O  
  
kubectl apply -f custom-resources.yaml

## **Worker Node Setup**

### **1. Repeat Master Node Steps (1-8)**

Follow the same steps as the master node, excluding control plane initialization.

### **2. Join the Cluster**

**Use the kubeadm join command generated during master initialization:**

sudo kubeadm join <MASTER\_IP>:6443 --token <TOKEN> --discovery-token-ca-cert-hash sha256:<HASH>

If the command is lost, regenerate it on the master node:

kubeadm token create --print-join-command

This completes the Kubernetes cluster setup. You now have a fully operational Kubernetes cluster with one master and multiple worker nodes.