**Install Packages**

1. Log in to the control plane node.

**Note:** The following steps must be performed on all three nodes.

1. Create the configuration file for containerd:

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

1. Load the modules:

sudo modprobe overlay sudo modprobe br\_netfilter

1. Set the system configurations for Kubernetes networking:

cat <<EOF | sudo tee /etc/sysctl.d/99-kubernetes-cri.conf net.bridge.bridge-nf-call-iptables = 1 net.ipv4.ip\_forward = 1 net.bridge.bridge-nf-call-ip6tables = 1 EOF

1. Apply the new settings:

sudo sysctl --system

1. Install containerd:

sudo apt-get update && sudo apt-get install -y containerd.io

1. Create the default configuration file for containerd:

sudo mkdir -p /etc/containerd

1. Generate the default containerd configuration, and save it to the newly created default file:

sudo containerd config default | sudo tee /etc/containerd/config.toml

1. Restart containerd to ensure the new configuration file is used:

sudo systemctl restart containerd

1. Verify that containerd is running:

sudo systemctl status containerd

1. Disable swap:

sudo swapoff -a

1. Install the dependency packages:

sudo apt-get update && sudo apt-get install -y apt-transport-https curl

1. Download and add the GPG key:

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

1. Add Kubernetes to the repository list:

cat <<EOF | sudo tee /etc/apt/sources.list.d/kubernetes.list deb https://apt.kubernetes.io/ kubernetes-xenial main EOF

1. Update the package listings:

sudo apt-get update

1. Install Kubernetes packages:

**Note:** If you get a dpkg lock message, just wait a minute or two before trying the command again.

sudo apt-get install -y kubelet=1.27.0-00 kubeadm=1.27.0-00 kubectl=1.27.0-00

1. Turn off automatic updates:

sudo apt-mark hold kubelet kubeadm kubectl

1. Log in to both worker nodes to perform the previous steps.

**Initialize the Cluster**

1. Initialize the Kubernetes cluster on the control plane node using kubeadm:

sudo kubeadm init --pod-network-cidr 192.168.0.0/16 --kubernetes-version 1.27.0

1. Set kubectl access:

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

1. Test access to the cluster:

kubectl get nodes

**Install the Calico Network Add-On**

1. On the control plane node, install Calico Networking:

kubectl apply -f https://raw.githubusercontent.com/projectcalico/calico/v3.25.0/manifests/calico.yaml

1. Check the status of the control plane node:

kubectl get nodes

**Join the Worker Nodes to the Cluster**

1. In the control plane node, create the token and copy the kubeadm join command:

kubeadm token create --print-join-command

**Note:** This output will be used as the next command for the worker nodes.

1. Copy the full output from the previous command used in the control plane node. This command starts with kubeadm join.
2. In both worker nodes, paste the full kubeadm join command to join the cluster. Use sudo to run it as root:

sudo kubeadm join...

1. In the control plane node, view the cluster status:

kubectl get nodes

**Note:** You may have to wait a few moments to allow all nodes to become ready.