Kubernetes installation:

Control plane

| **Protocol** | **Direction** | **Port Range** | **Purpose** | **Used By** |
| --- | --- | --- | --- | --- |
| TCP | Inbound | 6443 | Kubernetes API server | All |
| TCP | Inbound | 2379-2380 | etcd server client API | kube-apiserver, etcd |
| TCP | Inbound | 10250 | Kubelet API | Self, Control plane |
| TCP | Inbound | 10259 | kube-scheduler | Self |
| TCP | Inbound | 10257 | kube-controller-manager | Self |

| **Protocol** | **Direction** | **Port Range** | **Purpose** | **Used By** |
| --- | --- | --- | --- | --- |
| TCP | Inbound | 10250 | Kubelet API | Self, Control plane |
| TCP | Inbound | 30000-32767 | NodePort Services† | All |

Worker node(s)

🡪 Swap disabled. You **MUST** disable swap in order for the kubelet to work properly

🡪 sudo swapoff -a

🡪 vi /etc/hosts --🡪

65.0.176.183 master

3.110.54.18 worker1

3.110.204.55 worker2

🡪 hostnamectl set-hostname master (for nodes also)

**Containerd (both master and worker nodes)**

This section contains the necessary steps to use containerd as CRI runtime.

Use the following commands to install Containerd on your system:

Install and configure prerequisites:

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

overlay

br\_netfilter

EOF

sudo modprobe overlay

sudo modprobe br\_netfilter

*# Setup required sysctl params, these persist across reboots.*

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

*# Apply sysctl params without reboot*

sudo sysctl –system

**Install containerd:**

1.Install the containerd.io package from the official Docker repositories. Instructions for setting up the Docker repository for your respective Linux distribution and installing the containerd.io package can be found at [Install Docker Engine](https://docs.docker.com/engine/install/#server).

-🡪 sudo apt-get update

🡪 sudo apt-get install containerd

2. Configure containerd:

sudo mkdir -p /etc/containerd

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

3. Restart containerd:

sudo systemctl restart containerd

script to install and configure containerd:

install.sh

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

overlay

br\_netfilter

EOF

sudo modprobe overlay

sudo modprobe br\_netfilter

*# Setup required sysctl params, these persist across reboots.*

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

*# Apply sysctl params without reboot*

sudo sysctl --system

sudo apt-get update

sudo apt-get install conatinerd -y

sudo mkdir -p /etc/containerd

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

sudo systemctl restart containerd

service containerd status

**Installing kubeadm, kubelet and kubectl(both master and workernodes)**

Update the apt package index and install packages needed to use the Kubernetes apt repository:

sudo apt-get update

sudo apt-get install -y apt-transport-https ca-certificates curl

Download the Google Cloud public signing key:

sudo curl -fsSLo /usr/share/keyrings/kubernetes-archive-keyring.gpg https://packages.cloud.google.com/apt/doc/apt-key.gpg

Add the Kubernetes apt repository:

echo "deb [signed-by=/usr/share/keyrings/kubernetes-archive-keyring.gpg] https://apt.kubernetes.io/ kubernetes-xenial main" | sudo tee /etc/apt/sources.list.d/kubernetes.list

Update apt package index, install kubelet, kubeadm and kubectl, and pin their version:

sudo apt-get update

sudo apt-get install -y kubelet=1.21.0-00 kubeadm=1.21.0-00 kubectl=1.21.0-00

sudo apt-mark hold kubelet kubeadm kubectl

ref: <https://kubernetes.io/docs/setup/production-environment/tools/kubeadm/install-kubeadm/#check-required-ports>

[kubeadm init](https://kubernetes.io/docs/reference/setup-tools/kubeadm/kubeadm-init) to bootstrap a Kubernetes control-plane node

initialize cluster:

-🡪sudo kubeadm init

Access master: (thereis only one entry-point i.e api server)

1. Using kubeconfig file (contains api address, certificate and key)
2. Kubectl

🡪 kubectl get node (get error)

🡪 sudo kubectl get node –kubeconfig /etc/kubernates/admin.conf

Create admin.conf as environment variable

🡪sudo -i

-🡪 export KUBECONFIG=/etc/kubernetes/admin.conf

🡪 kubectl get node

🡪exit

ls ~/.kube

mkdir ~/.kube

sudo cp -i /etc/kubernetes/admin.conf

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

ls -l ~/.kube/config

sudo chown ubuntu:ubuntu /home/ubuntu/.kube/config

ls -l ~/.kube/config

namespace:

🡪 kubectl get namespace

Configuring networking: weave net

Ref : <https://www.weave.works/docs/net/latest/kubernetes/kube-addon/>

<https://www.weave.works/docs/net/latest/tasks/ipam/configuring-weave/>

🡪 wget "https://cloud.weave.works/k8s/net?k8s-version=$(kubectl version | base64 | tr -d '\n')" -O weave.yaml

🡪 vi weave.yaml

containers:

- name: weave

command:

- /home/weave/launch.sh

- --ipalloc-range=100.32.0.0/12 --- add this line

🡪 kubectl apply -f weave.yaml

🡪 kubectl get node --- now master running

🡪 kubectl get pod -n kube-system

🡪 kubectl get pod -n kube-system -o wide --- checking ports

Add worker nodes:

🡪 sudo swapoff -a ( worker nodes)

🡪 kubeadm token create –print-join-command (master)

🡪 kubeadm join 172.31.2.216:6443 --token sld7qr.j8uwiu6apysd2inc --discovery-token-ca-cert-hash sha256:8825bcfc2a69f67b589b4cc4b6b155d4bb0a8c52549f8eb76f3bf72dcf5c0057 (worker node)