Kubernetes Cluster Setup With kubeadm And Monitoring Tools (Prometheus, Grafana)

Setup Guide

JUNE 23, 2025

FP

Requirements

- Two Virtual Machines (CentOS 8/9):
  - 1 Master Node
  - 1 Worker Node
- Internet access

Part 1: Master Node Setup

Step 1: Run Setup Script

Create and run the following script on the master node:

install-kubeadm-master.sh

```
#!/bin/bash
set -e

# Firewall Rules
cat <<EOF | sudo tee /usr/lib/firewalld/services/kubernetes-control-plane.xml
<?xml version="1.0" encoding="utf-8"?>
<service>
  <short>Kubernetes Control Plane</short>
  <description>Ports required for Kubernetes control plane components.</description>
  <port protocol="tcp" port="6443"/>
  <port protocol="tcp" port="2379-2380"/>
  <port protocol="tcp" port="10250"/>
  <port protocol="tcp" port="10259"/>
  <port protocol="tcp" port="10257"/>
</service>
EOF

sudo firewall-cmd --reload
sudo firewall-cmd --permanent --add-service=kubernetes-control-plane
sudo firewall-cmd --reload

# Disable SELinux and Swap
setenforce 0 || true
sed -i 's/^SELINUX=enforcing/SELINUX=permissive/' /etc/selinux/config
swapoff -a
sed -i '/swap/d' /etc/fstab

# Kernel Modules and Sysctl
cat <<EOF | tee /etc/modules-load.d/k8s.conf
br_netfilter
EOF
modprobe br_netfilter

cat <<EOF | tee /etc/sysctl.d/k8s.conf
net.bridge.bridge-nf-call-ip6tables = 1
net.bridge.bridge-nf-call-iptables = 1
net.ipv4.ip_forward = 1
EOF
```

```
sysctl --system

# Install containerd
sudo dnf install -y yum-utils device-mapper-persistent-data lvm2
sudo dnf config-manager --add-repo=https://download.docker.com/linux/centos/docker-
ce.repo
sudo dnf install -y containerd.io

mkdir -p /etc/containerd
containerd config default | tee /etc/containerd/config.toml
sed -i 's/SystemdCgroup = false/SystemdCgroup = true/' /etc/containerd/config.toml
systemctl enable --now containerd

# Kubernetes Repo
cat <<EOF | sudo tee /etc/yum.repos.d/kubernetes.repo
[kubernetes]
name=Kubernetes
baseurl=https://pkgs.k8s.io/core:/stable:/v1.32/rpm/
enabled=1
gpgcheck=1
gpgkey=https://pkgs.k8s.io/core:/stable:/v1.32/rpm/repodata/repomd.xml.key
exclude=kubelet kubeadm kubectl cri-tools kubernetes-cni
EOF

# Install Kubernetes Tools
sudo dnf install -y kubelet kubeadm kubectl --disableexcludes=kubernetes
sudo systemctl enable --now kubelet
```

Step 2: Initialize Cluster

```
sudo kubeadm init --pod-network-cidr=192.168.0.0/16
```

Step 3: 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
```

Step 4: Install Calico Network Plugin

```
kubectl apply -f
https://raw.githubusercontent.com/projectcalico/calico/v3.27.0/manifests/calico.yaml
```

Step 5: Get Join Command for Worker

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

Part 2: Worker Node Setup

install-kubeadm-worker.sh

```
#!/bin/bash
set -e

# Firewall Rules
cat <<EOF | sudo tee /usr/lib/firewalld/services/kubernetes-worker-node.xml
<?xml version="1.0" encoding="utf-8"?>
<service>
  <short>Kubernetes Worker Node</short>
  <description>Ports required for Kubernetes worker node components.</description>
  <port protocol="tcp" port="10250"/>
  <port protocol="tcp" port="10256"/>
  <port protocol="tcp" port="30000-32767"/>
</service>
EOF

sudo firewall-cmd --reload
sudo firewall-cmd --permanent --add-service=kubernetes-worker-node
sudo firewall-cmd --reload

# Disable swap
swapoff -a
sed -i '/swap/d' /etc/fstab

# Kernel Modules
modprobe overlay
modprobe br_netfilter
cat <<EOF | sudo tee /etc/modules-load.d/k8s.conf
overlay
br_netfilter
EOF

# Sysctl Settings
cat <<EOF | sudo tee /etc/sysctl.d/k8s.conf
net.bridge.bridge-nf-call-ip6tables = 1
net.bridge.bridge-nf-call-iptables = 1
net.ipv4.ip_forward = 1
EOF
sysctl --system

# Install containerd
sudo dnf install -y yum-utils device-mapper-persistent-data lvm2 curl
sudo dnf config-manager --add-repo https://download.docker.com/linux/centos/docker-
ce.repo
sudo dnf install -y containerd.io

mkdir -p /etc/containerd
containerd config default | tee /etc/containerd/config.toml
sed -i 's/SystemdCgroup = false/SystemdCgroup = true/' /etc/containerd/config.toml
systemctl daemon-reexec
systemctl enable --now containerd
```

```
# Kubernetes Repo
cat <<EOF | tee /etc/yum.repos.d/kubernetes.repo
[kubernetes]
name=Kubernetes
baseurl=https://pkgs.k8s.io/core:/stable:/v1.29/rpm/
enabled=1
gpgcheck=1
repo_gpgcheck=1
gpgkey=https://pkgs.k8s.io/core:/stable:/v1.29/rpm/repodata/repomd.xml.key
EOF

# Install Kubernetes Tools
sudo dnf install -y kubelet kubeadm kubectl --disableexcludes=kubernetes
sudo systemctl enable --now kubelet
```

Step 2: Join Cluster (run join command from master)

```
kubeadm join <MASTER-IP>:6443 --token <token> --discovery-token-ca-cert-hash
sha256:<hash>
```

Monitoring Tools

Prometheus Server On Master

install-prometheus.sh

```
#!/bin/bash
dnf install -y wget tar

useradd -M -r -s /bin/false prometheus
mkdir -p /etc/prometheus /var/lib/prometheus

cd /tmp
wget https://github.com/prometheus/prometheus/releases/download/v2.40.7/prometheus-
2.40.7.linux-amd64.tar.gz
tar -xzf prometheus-2.40.7.linux-amd64.tar.gz

cp prometheus-2.40.7.linux-amd64/prometheus /usr/local/bin/
cp prometheus-2.40.7.linux-amd64/promtool /usr/local/bin/

cp -r prometheus-2.40.7.linux-amd64/consoles /etc/prometheus/
cp -r prometheus-2.40.7.linux-amd64/console_libraries /etc/prometheus/
cp prometheus-2.40.7.linux-amd64/prometheus.yml /etc/prometheus/

chown -R prometheus:prometheus /etc/prometheus /var/lib/prometheus
/usr/local/bin/prometheus /usr/local/bin/promtool

cat <<EOF > /etc/systemd/system/prometheus.service
[Unit]
Description=Prometheus Server
Wants=network-online.target
After=network-online.target

[Service]
User=prometheus
Group=prometheus
Type=simple
ExecStart=/usr/local/bin/prometheus \
  --config.file /etc/prometheus/prometheus.yml \
  --storage.tsdb.path=/var/lib/prometheus \
  --web.console.templates=/etc/prometheus/consoles \
  --web.console.libraries=/etc/prometheus/console_libraries

[Install]
WantedBy=multi-user.target
EOF


systemctl daemon-reload
systemctl enable --now prometheus
firewall-cmd --add-port=9090/tcp --permanent
firewall-cmd –reload
```

Node Exporter On Worker

install-node-exporter.sh

```
#!/bin/bash

dnf install -y wget tar

useradd -M -r -s /bin/false node_exporter
cd /tmp
wget https://github.com/prometheus/node_exporter/releases/download/v1.5.0/node_exporter-1.5.0.linux-amd64.tar.gz
tar xvf node_exporter-1.5.0.linux-amd64.tar.gz
cp node_exporter-1.5.0.linux-amd64/node_exporter /usr/local/bin/
chown node_exporter:node_exporter /usr/local/bin/node_exporter

cat <<EOF > /etc/systemd/system/node_exporter.service
[Unit]
Description=Prometheus Node Exporter
Wants=network-online.target
After=network-online.target

[Service]
User=node_exporter
Group=node_exporter
Type=simple
ExecStart=/usr/local/bin/node_exporter
[Install]
WantedBy=multi-user.target
EOF

systemctl daemon-reload
systemctl start node_exporter
systemctl enable node_exporter
firewall-cmd --add-port=9100/tcp --permanent
firewall-cmd –reload
```

Configure Prometheus to Scrape Node Exporter on Worker

After installing Node Exporter on the worker node, edit Prometheus config file on the master node:

/etc/prometheus/prometheus.yml

```
# For worker1
  - job_name: "worker1"
    static_configs:
      - targets: ["workerIP:9100"]
```

Then reload Prometheus:

```
systemctl restart prometheus
```

This will enable Prometheus to collect metrics from the Node Exporter running on the worker node.

Grafana

install-grafana.sh

```
#!/bin/bash
cat <<EOF > /etc/yum.repos.d/grafana.repo
[grafana]
name=grafana
baseurl=https://packages.grafana.com/oss/rpm
repo_gpgcheck=1
enabled=1
gpgcheck=1
gpgkey=https://packages.grafana.com/gpg.key
sslverify=1
sslcacert=/etc/pki/tls/certs/ca-bundle.crt
EOF

dnf install -y grafana
systemctl daemon-reload
systemctl enable --now grafana-server
firewall-cmd --add-port=3000/tcp --permanent
firewall-cmd --reload
```
