Kubernetes Installation on Red Hat Enterprise Linux 8

System: RHEL 8 with kernel 4.18.0-553.50.1.el8_10.x86_64

Prerequisites

Before starting, ensure you have:

- Root or sudo access
- At least 2 CPUs and 2GB RAM
- Network connectivity
- Unique hostname, MAC address, and product_uuid for each node

Step 1: System Preparation and Package Installation

1.1 Update the System

```bash
sudo dnf update -y
```

1.2 Install Essential Packages

```bash
```

```
# Install basic system utilities and tools
sudo dnf install -y \
    curl \
    wget \
    vim \
    nano \
    htop \
    net-tools \
    bind-utils \
    git \
    unzip \
    tar \
    which \
    tree \
    lsof

# Install firewall and network utilities
sudo dnf install -y \
    firewalld \
    iptables-services \
    ipset \
    ipvsadm

# Install container and system utilities
sudo dnf install -y \
    yum-utils \
    device-mapper-persistent-data \
    lvm2 \
    chrony \
    rsync

# Start and enable essential services
sudo systemctl enable --now firewalld
sudo systemctl enable --now chronyd

# Verify firewalld is running
sudo systemctl status firewalld
```

1.3 Set Hostname (if needed)

bash

```
sudo hostnamectl set-hostname k8s-master
# or k8s-worker1, k8s-worker2, etc.
```

1.4 Disable Swap

```
bash

# Temporarily disable swap
sudo swapoff -a

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

1.5 Configure SELinux

```
bash

# Set SELinux to permissive mode
sudo setenforce 0
sudo sed -i 's/^SELINUX=enforcing$/SELINUX=permissive/' /etc/selinux/config
```

1.6 Configure Firewall

```
bash

# For Master Node:
sudo firewall-cmd --permanent --add-port=6443/tcp
sudo firewall-cmd --permanent --add-port=2379-2380/tcp
sudo firewall-cmd --permanent --add-port=10250/tcp
sudo firewall-cmd --permanent --add-port=10251/tcp
sudo firewall-cmd --permanent --add-port=10252/tcp

# For Worker Nodes:
sudo firewall-cmd --permanent --add-port=10250/tcp
sudo firewall-cmd --permanent --add-port=30000-32767/tcp

# Apply firewall rules
sudo firewall-cmd --reload
```

Step 2: Install Container Runtime (containerd)

2.1 Load Required Kernel Modules

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

sudo modprobe overlay
sudo modprobe br_netfilter
```

2.2 Configure sysctl Parameters

```bash
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

sudo sysctl --system
```

2.3 Install containerd

```bash
# Install containerd
sudo dnf config-manager --add-repo https://download.docker.com/linux/centos/docker-ce.repo
sudo dnf install -y containerd.io

# Configure containerd
sudo mkdir -p /etc/containerd
containerd config default | sudo tee /etc/containerd/config.toml

# Enable SystemdCgroup
sudo sed -i 's/SystemdCgroup \= false/SystemdCgroup \= true/g' /etc/containerd/config.toml

# Start and enable containerd
sudo systemctl restart containerd
sudo systemctl enable containerd
```

Step 3: Install Kubernetes Components

3.1 Add Kubernetes Repository

```bash
cat <<EOF | sudo tee /etc/yum.repos.d/kubernetes.repo
[kubernetes]
name=Kubernetes
baseurl=https://pkgs.k8s.io/core:/stable:/v1.28/rpm/
enabled=1
gpgcheck=1
gpgkey=https://pkgs.k8s.io/core:/stable:/v1.28/rpm/repodata/repomd.xml.key
exclude=kubelet kubeadm kubectl cri-tools kubernetes-cni
EOF
```

3.2 Install Kubernetes Components

```bash
sudo dnf install -y kubelet kubeadm kubectl --disableexcludes=kubernetes

# Enable kubelet
sudo systemctl enable --now kubelet
```

Step 4: Initialize Kubernetes Cluster (Master Node Only)

4.1 Initialize the Cluster

```bash
sudo kubeadm init --pod-network-cidr=10.244.0.0/16
```

4.2 Configure kubectl for Regular User

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

4.3 Save the Join Command

After `kubeadm init` completes, save the join command output. It will look like:

```bash
```

```
kubeadm join <master-ip>:6443 --token <token> \
    --discovery-token-ca-cert-hash sha256:<hash>
```

Step 5: Install Pod Network Add-on (Master Node)

5.1 Install Flannel CNI

```bash
kubectl apply -f https://github.com/flannel-io/flannel/releases/latest/download/kube-flannel.yml
```

5.2 Verify Master Node Status

```bash
kubectl get nodes
kubectl get pods --all-namespaces
```

Step 6: Join Worker Nodes to Cluster

6.1 Prepare Worker Nodes

Complete Steps 1-3 on each worker node.

6.2 Join Worker Nodes

Run the join command saved from Step 4.3 on each worker node:

```bash
sudo kubeadm join <master-ip>:6443 --token <token> \
    --discovery-token-ca-cert-hash sha256:<hash>
```

Step 7: Verification

7.1 Check Cluster Status

```bash
kubectl get nodes
kubectl get pods --all-namespaces
kubectl cluster-info
```

7.2 Deploy Test Application

```bash
kubectl create deployment nginx --image=nginx
kubectl expose deployment nginx --port=80 --type=NodePort
kubectl get services
```

Troubleshooting

Common Issues and Solutions

1. **Kubelet not starting:**

```bash
sudo systemctl status kubelet
sudo journalctl -xeu kubelet
```

2. **Container runtime not ready:**

```bash
sudo systemctl restart containerd
sudo systemctl status containerd
```

3. **Network issues:**

```bash
# Check if br_netfilter is loaded
lsmod | grep br_netfilter

# Verify sysctl settings
sysctl net.bridge.bridge-nf-call-iptables
```

4. **Token expired for joining nodes:**

```bash
# Generate new token on master
kubeadm token create --print-join-command
```

Optional: Enable kubectl autocompletion

```bash
```

```
echo 'source <(kubectl completion bash)' >>~/.bashrc
source ~/.bashrc
```

Security Considerations

- Change default passwords and tokens in production
- Configure proper RBAC policies
- Use network policies to restrict pod-to-pod communication
- Regularly update Kubernetes components
- Consider using a service mesh for additional security

Your Kubernetes cluster should now be ready for use!