Kubernetes Installation using Kubeadm

System Requirements

- **OS:** Ubuntu 24.04 LTS server
- **RAM:** 2GB or more
- **Disk:** 20GB+
- **CPU:** 2 cores or more
- **Swap Memory:** Disable

1.Disable Swap Temporarily

➢ **sudo swapoff -a**

2. Disable Swap Permanently

➢ **sudo vi /etc/fstab**

```
# Comment the below line
        UUID=<some-uuid> none swap sw 0
```

If not show above line, No problem u can move next step

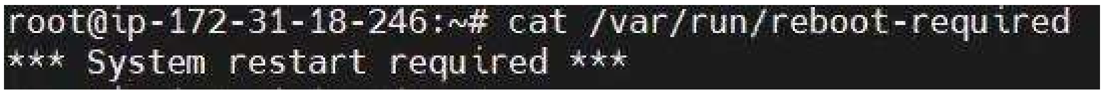
3. Run Below Commands on Both Master and Worker Nodes

➢ **apt update && apt upgrade -y**

#Check if the System Needs Reboot

➢ **cat /var/run/reboot-required**

Once put this command we will get this



```
root@ip-172-31-18-246:~# cat /var/run/reboot-required
*** System restart required ***
```

We should restart system. After that run this below commands,

➢
```
cat > /etc/modules-load.d/containerd.conf <<EOF
overlay
br_netfilter
EOF
```

➢ **modprobe overlay**

- modprobe br_netfilter

overlay: Enables the overlayfs kernel module, which is required for container images.

br_netfilter: Enables the bridge netfilter kernel module, which is necessary for Kubernetes networking features like inter-pod communication.

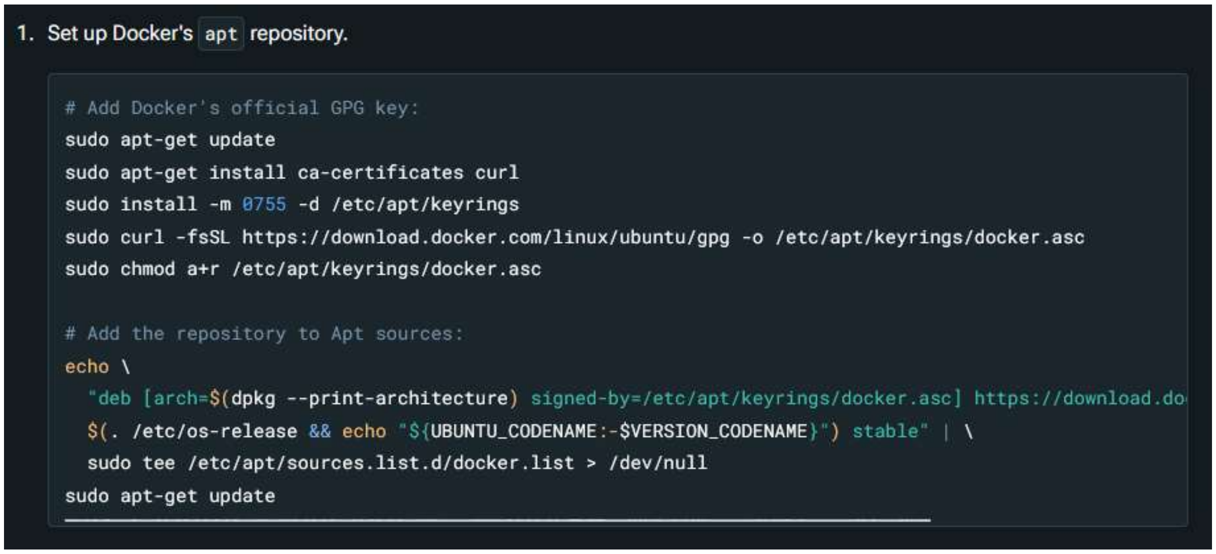
- cat > /etc/sysctl.d/99-kubernetes-cri.conf <<EOF
```
net.bridge.bridge-nf-call-iptables  = 1
net.ipv4.ip_forward                 = 1
net.bridge.bridge-nf-call-ip6tables = 1
EOF
```

- sysctl --system

4.Install containerd on both Master and worker

https://docs.docker.com/engine/install/ubuntu/

This is installation link of containerd, you should run commands, commands are like this



1. Set up Docker's `apt` repository.

```
# Add Docker's official GPG key:
sudo apt-get update
sudo apt-get install ca-certificates curl
sudo install -m 0755 -d /etc/apt/keyrings
sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg -o /etc/apt/keyrings/docker.asc
sudo chmod a+r /etc/apt/keyrings/docker.asc

# Add the repository to Apt sources:
echo \
  "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.asc] https://download.do
  $(. /etc/os-release && echo "${UBUNTU_CODENAME:-$VERSION_CODENAME}") stable" | \
  sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
sudo apt-get update
```

- sudo apt-get install containerd.io -y

Check if containerd installed or not,

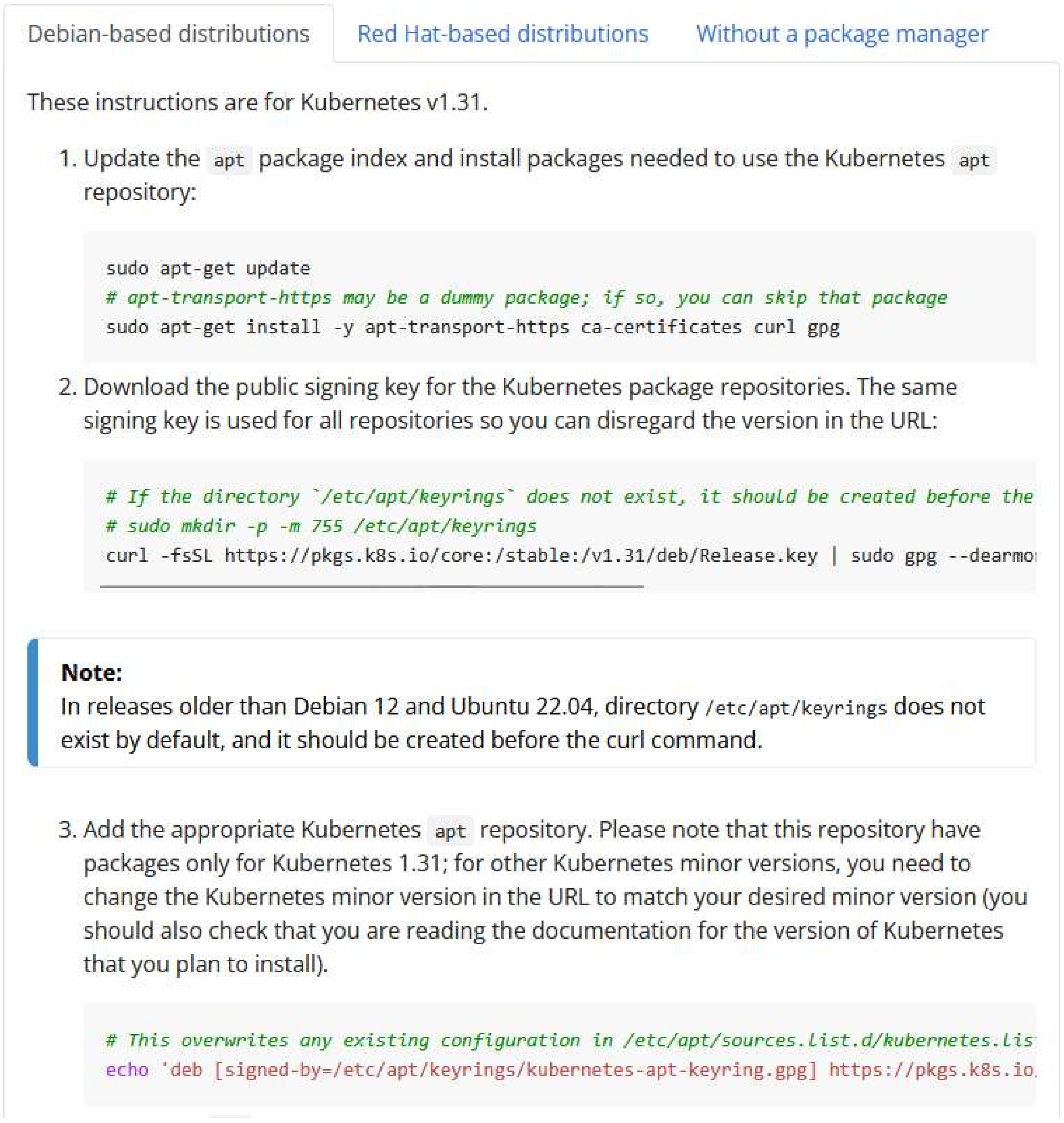
- Systemctl status containerd

- **mkdir -p /etc/containerd**
- **containerd config default > /etc/containerd/config.toml**
- **sed -i 's/SystemdCgroup = false/SystemdCgroup = true/g' /etc/containerd/config.toml**
- **systemctl restart containerd**

Create the crictl configuration file

- **cat > /etc/crictl.yaml <<EOF**
  runtime-endpoint: unix:///run/containerd/containerd.sock
  image-endpoint: unix:///run/containerd/containerd.sock
  timeout: 2
  EOF

5.Install Kubernetes Packages on All Nodes


Debian-based distributions | Red Hat-based distributions | Without a package manager

These instructions are for Kubernetes v1.31.

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

```
sudo apt-get update
# apt-transport-https may be a dummy package; if so, you can skip that package
sudo apt-get install -y apt-transport-https ca-certificates curl gpg
```

2. Download the public signing key for the Kubernetes package repositories. The same signing key is used for all repositories so you can disregard the version in the URL:

```
# If the directory `/etc/apt/keyrings` does not exist, it should be created before the
# sudo mkdir -p -m 755 /etc/apt/keyrings
curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.31/deb/Release.key | sudo gpg --dearmo
```

Note:
In releases older than Debian 12 and Ubuntu 22.04, directory `/etc/apt/keyrings` does not exist by default, and it should be created before the curl command.

3. Add the appropriate Kubernetes `apt` repository. Please note that this repository have packages only for Kubernetes 1.31; for other Kubernetes minor versions, you need to change the Kubernetes minor version in the URL to match your desired minor version (you should also check that you are reading the documentation for the version of Kubernetes that you plan to install).

```
# This overwrites any existing configuration in /etc/apt/sources.list.d/kubernetes.lis
echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io
```

- **sudo apt-get update**
- **sudo apt-get install -y kubelet kubeadm kubectl**
- **sudo apt-mark hold kubelet kubeadm kubectl**

https://kubernetes.io/docs/setup/production-environment/tools/kubeadm/install-kubeadm/

This is installation link of Kubernetes package

- You can choose kubeadm version and scroll down you can see link as shown in the above image.
- Why i put the images instead of giving commands, because version problem will happen that's why I do this.

6.Run Below Commands on Master Node Alone

Enable IP Forwarding

1.**Check the current value of ip_forward:**

```
cat /proc/sys/net/ipv4/ip_forward
```

2.**Enable IP forwarding:**

Edit the /etc/sysctl.conf file:

```
sudo nano /etc/sysctl.conf
```

Find the following line and uncomment it (or add it if it doesn't exist):

```
net.ipv4.ip_forward=1
```

Save and exit the file.

3.**Apply the changes:**

```
sudo sysctl -p
```

After enabling IP forwarding, you can run kubeadm init command:

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

You will receive a token to join worker nodes to the master node. Copy and paste this token into a notepad.

Example : Dont copy this **kubeadm join 172.31.30.229:6443 --token cz9cwu.2muc1x6gqz4d7d6f **

```
    --discovery-token-ca-cert-hash
sha256:1836752e4fbf1411263019edaecd13a53241f73eab06b5440bffe476f3dcb986
```

After that you can run below commands,

```
kubectl create -f https://docs.projectcalico.org/manifests/tigera-operator.yaml
wget https://docs.projectcalico.org/manifests/custom-resources.yaml

kubectl apply -f custom-resources.yaml
```

Verify

```
    watch kubectl get pods --all-namespaces
```

Now we can check Kubernetes work or not.

Deploy Nginx deployment:

```
kubectl create deployment nginx-deployment --image=nginx --replicas=1 --port=80
```

Explanation:

- **kubectl create deployment: Creates a deployment.**
- **nginx-deployment: The name of the deployment.**
- **--image=nginx: Specifies the Docker image to use (in this case, nginx).**
- **--replicas=1: Sets the number of replicas (pods) to 1.**
- **--port=80: Exposes port 80 on the container.**

To expose the Nginx deployment as a service, you can use the following command:

```
kubectl expose deployment nginx-deployment --type=NodePort --port=80
```

Explanation:

- **kubectl expose deployment: Exposes the deployment as a service.**
- **nginx-deployment: The name of the deployment to expose.**
- **--type=NodePort: Exposes the service on a port on each node in the cluster.**
- **--port=80: Specifies the port the service should expose.**