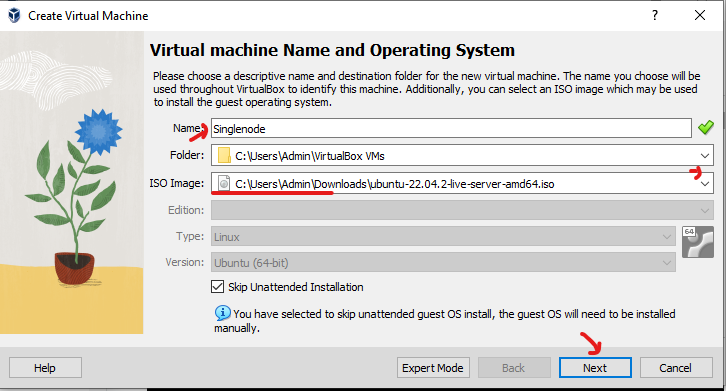
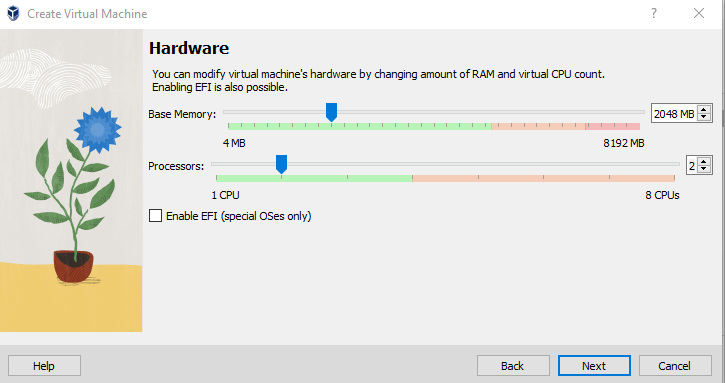
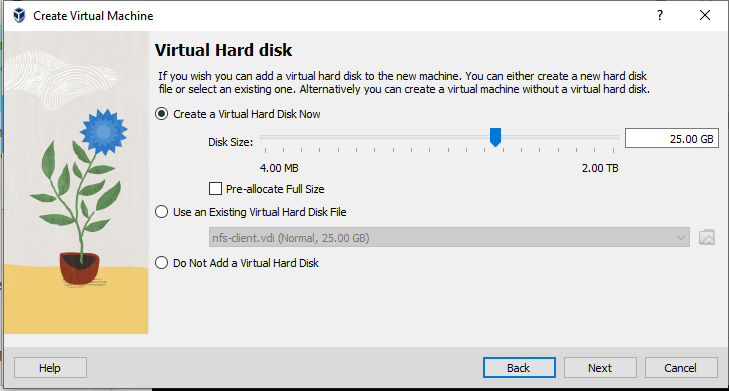
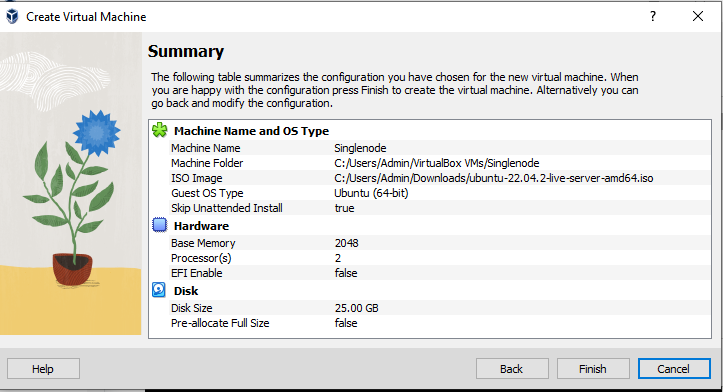
Kubernetes single node setup using kubeadm tool(CRI containerd)

Create virtual machine. Give the name and select iso image and enable skip unattended Installation option. Skip unattended – before start installation to do some particular changes we need to select this option. Click on next.

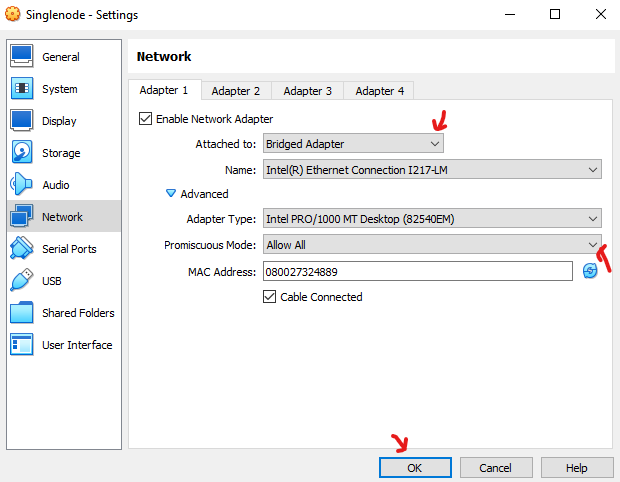








Go to setting – networking – select Bridge network – allow all (to connect with base machine and within the VM’s as well)



Do the remaining installation process as usual.

In the system If reboot is required we will see the file like reboot-required in this path /var/run/reboot-required means if the file is not their then reboot is not required.

<https://github.com/swapnachatla/kubernetes\_latest\_manifest/blob/main/Kubernetes/01-kubernetes-architecture-Installation/03-k8s-setup-kubeadm-containerd.md>

Before installing kubernetes we need to install, enable some modules and networking fileds.

Make the IP od VM must be static for that go iside of the particular path

# vi /etc/netplan/00-installer-config.yaml

network:

ethernets:

enp0s3:

dhcp4: false

addresses: ["192.168.1.62/24"]

routes:

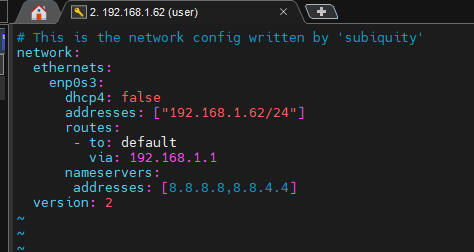
- to: default

via: 192.168.1.1

nameservers:

addresses: [8.8.8.8,8.8.4.4]

version: 2



Save & quit from that file (esc:wq!).

To apply the changes execute the below command.

# netplan apply



First we need to enable overlay and Bridge-network filters. For that we can use the below commands.

modprobe overlay

modprobe br\_netfilter

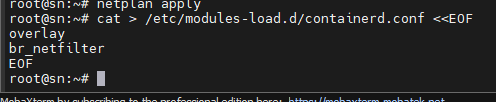
But the drawback of using these commands is after rebooting the system we need to execute the commands again. To enable them permanently we use the below commands.

cat > /etc/modules-load.d/containerd.conf <<EOF

overlay

br\_netfilter

EOF



Now setup the sysctl parameters.

# Setup required sysctl params, these persist across reboots.

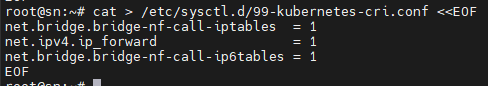
cat > /etc/sysctl.d/99-kubernetes-cri.conf <<EOF

net.bridge.bridge-nf-call-iptables = 1

net.ipv4.ip\_forward = 1

net.bridge.bridge-nf-call-ip6tables = 1

EOF

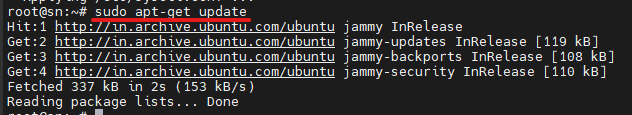


Now to continue without rebooting use the below command.

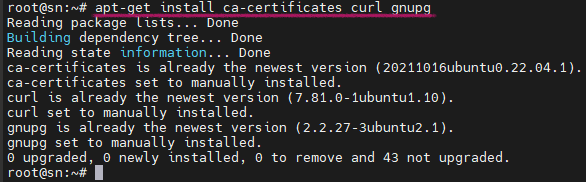
sysctl --system

Now install containerd package. For that we can use the official Docker install on Ubuntu document but don’t install docker just install containerd pkg only. <https://docs.docker.com/engine/install/ubuntu/>

# sudo apt-get update

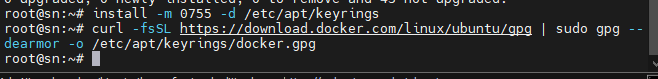


# apt-get install ca-certificates curl gnupg



# install -m 0755 -d /etc/apt/keyrings

# curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg



# chmod a+r /etc/apt/keyrings/docker.gpg

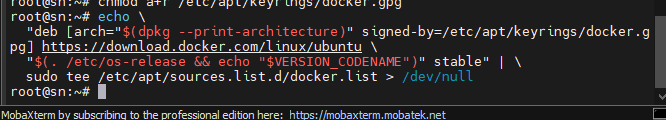


# echo \

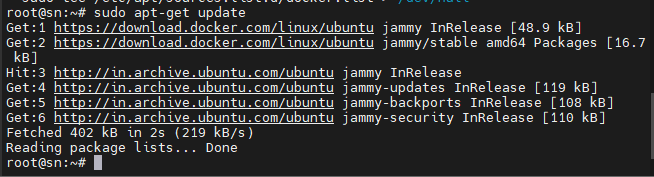
"deb [arch="$(dpkg --print-architecture)" signed-by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/ubuntu \

"$(. /etc/os-release && echo "$VERSION\_CODENAME")" stable" | \

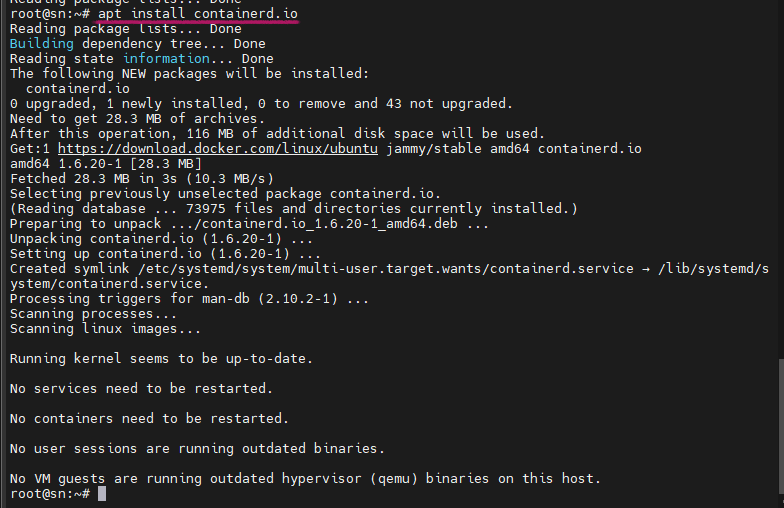
sudo tee /etc/apt/sources.list.d/docker.list > /dev/null



# sudo apt-get update



# apt install containerd.io



After installing containerd.io package we need to configure containerd.

# mkdir -p /etc/containerd

# containerd config default > /etc/containerd/config.toml



Now we need to change systemCgroup false to true.

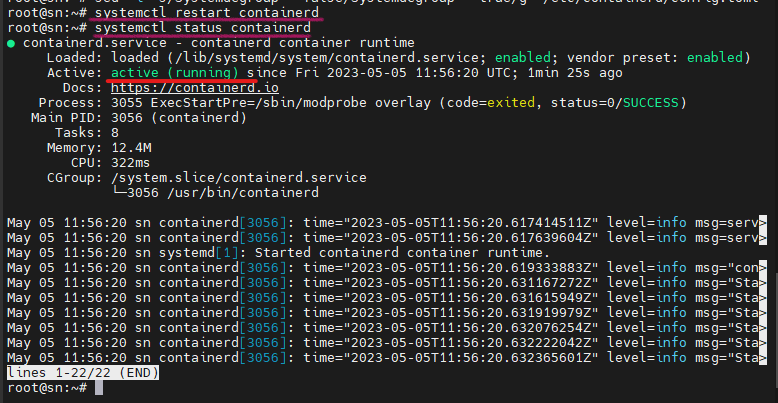
sed -i 's/SystemdCgroup = false/SystemdCgroup = true/g' /etc/containerd/config.toml



# Restart containerd

systemctl restart containerd

systemctl status containerd



To interact with containerd we use docker cli commands before. But here, we are not using Docker. So, now to interact with containerd we are using crictl CLI commands it is available by default.

# To execute crictl CLI commands, ensure we create a configuration file as mentioned below

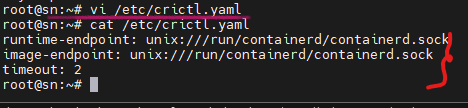
# vi /etc/crictl.yaml

runtime-endpoint: unix:///run/containerd/containerd.sock

image-endpoint: unix:///run/containerd/containerd.sock

timeout: 2

Crictl command will follow the yaml file to communicate with containerd.

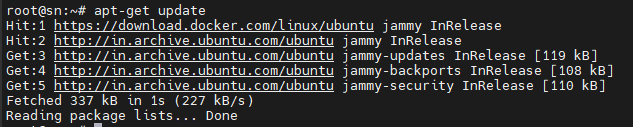


Install kubernetes package:

For install kubernetes follow the official document. In Browser just search like <Kubeadm install> then select the official document it’s URL looks like this

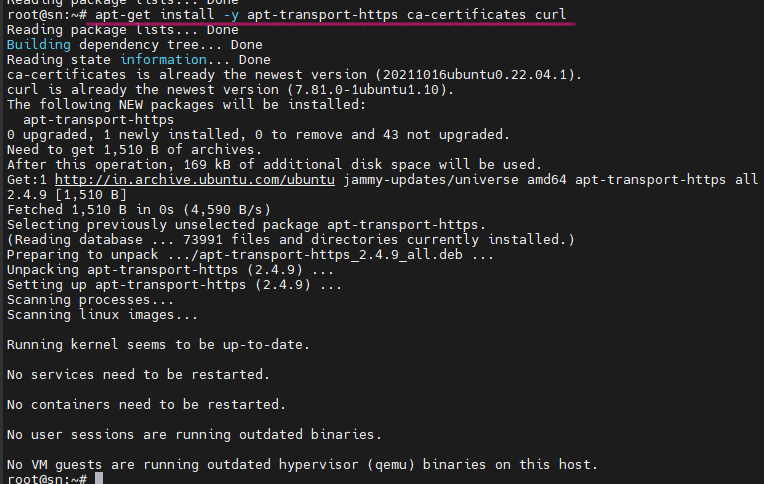
< https://kubernetes.io/docs/setup/production-environment/tools/kubeadm/install-kubeadm/>

# apt-get update



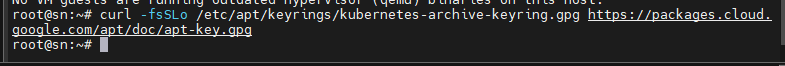
Now install the pre-required packages which are needed for kubernetes

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



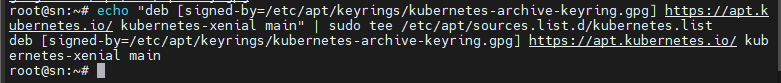
Download the Gpg key for kubernetes repository.

# curl -fsSLo /etc/apt/keyrings/kubernetes-archive-keyring.gpg <https://packages.cloud.google.com/apt/doc/apt-key.gpg>



Create a repository for kubernetes packages to be installed and download.

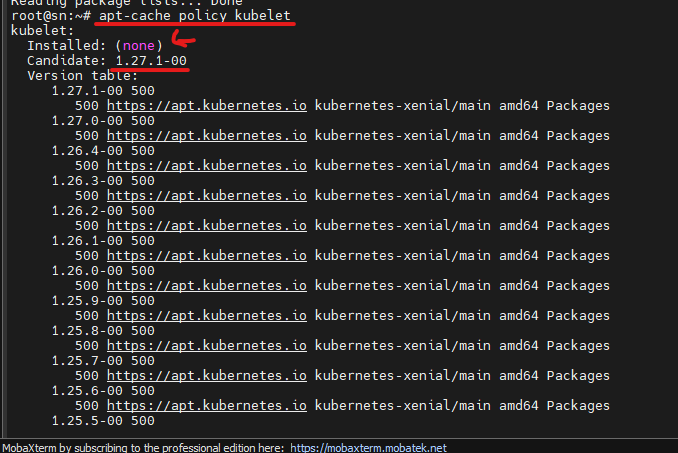
echo "deb [signed-by=/etc/apt/keyrings/kubernetes-archive-keyring.gpg] https://apt.kubernetes.io/ kubernetes-xenial main" | sudo tee /etc/apt/sources.list.d/kubernetes.list



# apt update

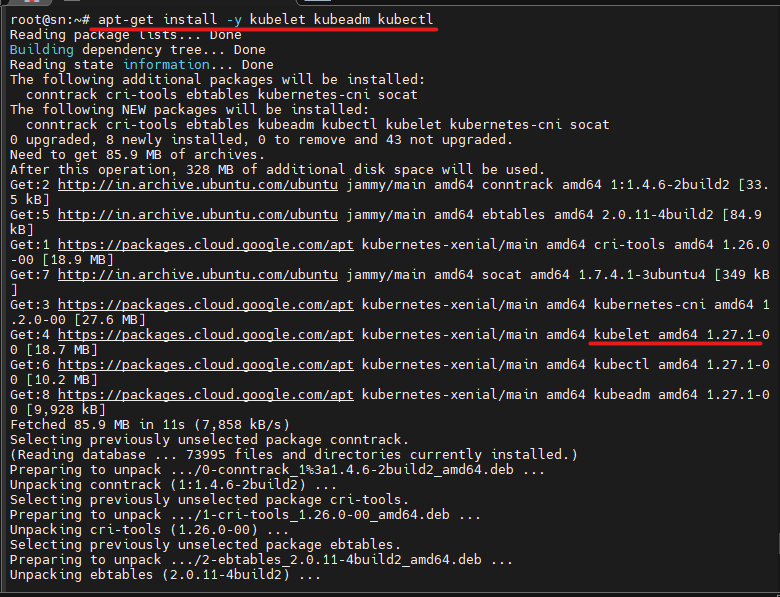
The below command will show the patch version. Candidate: showing the version, Installed: none means now package is not installed

# apt-cache policy kubelet



Now install the kubelet, kubeadm, kubectl packages. And the version of the package installed is 1.27

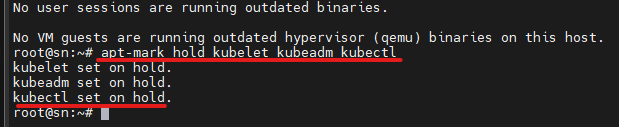
# apt-get install -y kubelet kubeadm kubectl



By using the below command the packages will not upgrade automatically.

# apt-mark hold kubelet kubeadm kubectl

See in the output it is showing that the packages set on hold.



Now run the command to create kubernetes cluster that particular task called as bootstrapping the control node.