Module 2: Kubernetes Architecture

Demo Document 1

Installation Guide

Note: For this installation we recommend a fresh ubuntu image since Kubernetes can take up a lot of resources.

Following are the preferable VM settings:

Master:

- 2GB RAM
- 2 Cores of CPU

Slave Node:

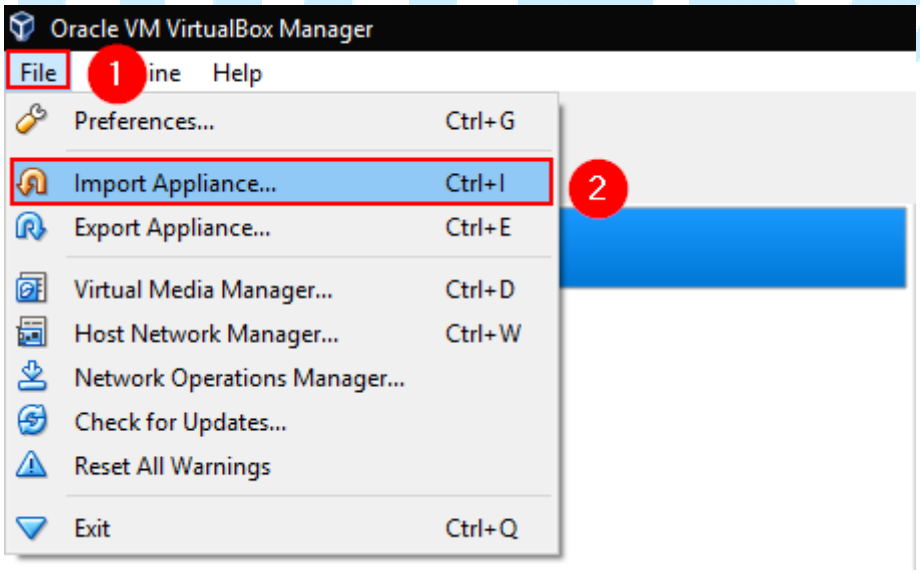
- 1 GB RAM
- 1 Core of CPU

Importing the Clean Ubuntu VM

Note: You will find the download link to this VM in the Pre-Installed VMs Guide

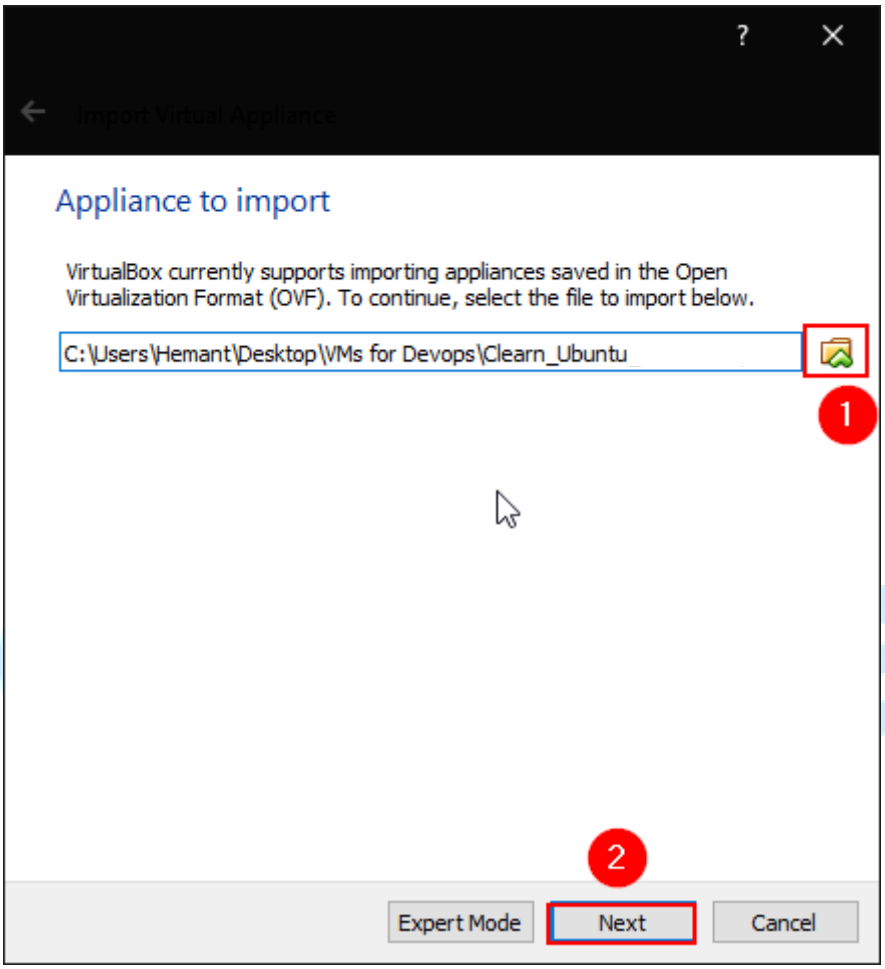
Step 1:

1. Open your Virtual Box Manager, and click on File.
2. Click on Import Appliance.

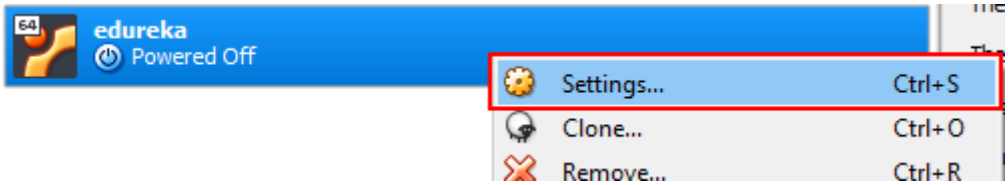


Step 2:

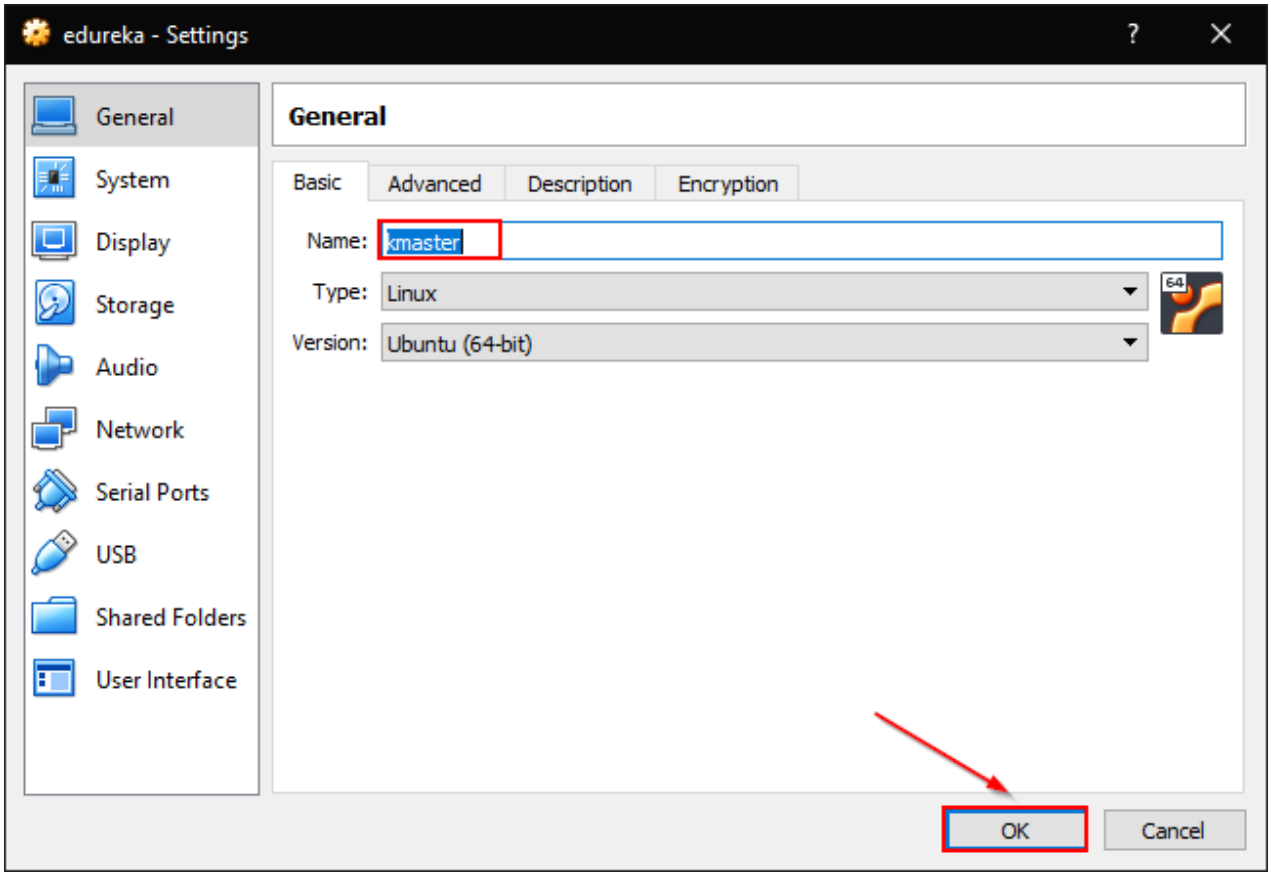
1. Click on Browse, and navigate to the place where you have downloaded the VM and select it.
2. Click on Next, and on the next page leave everything at default and Click on Import.



Step 3: Right Click on your VM, and click on Settings.



Step 4: Edit the name as kmaster, and click on OK



Step 5: Repeat the same steps to get a Slave Node, and name it as “knode”.

Steps for Master and Slave VMs

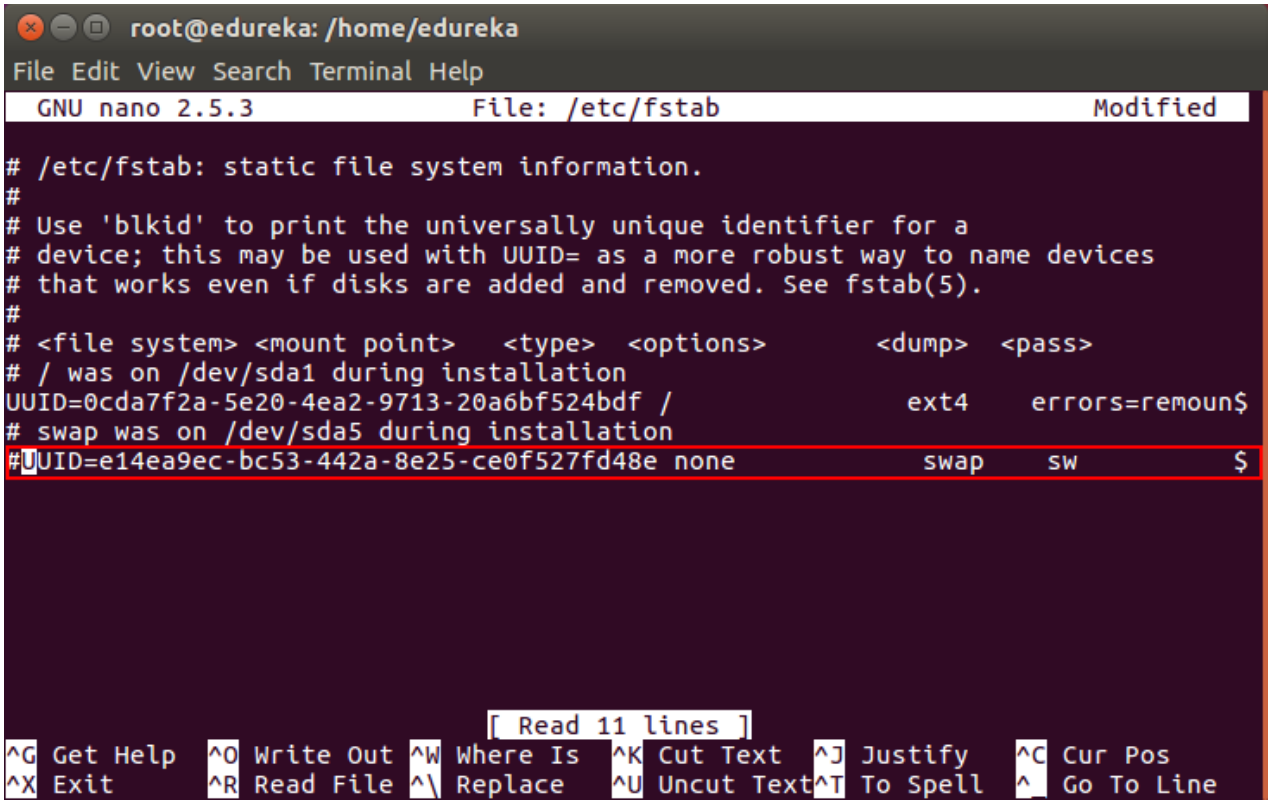
Note: These steps are common to both kmaster and knode VMs

Step 1:

1. Run the following commands:

```
sudo su
apt-get update
swapoff -a
nano /etc/fstab
```

2. Add a "#" in front of the last line, to comment it.



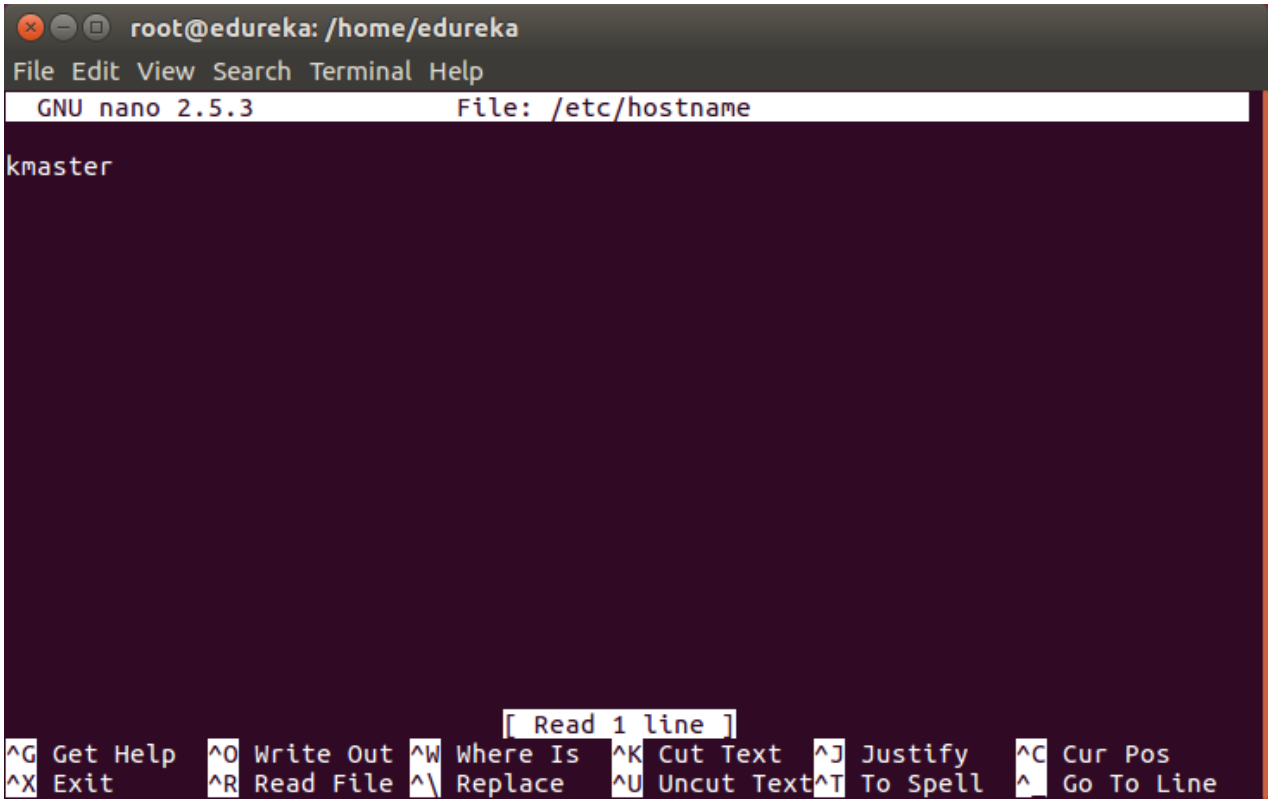
3. Press Ctrl+X, then press Y, and then press Enter to Save.

Step 2:

1. Run the following command:

```
nano /etc/hostname
```

2. Edit the name to “kmaster” for kmaster VM, and “knode” for knode VM.



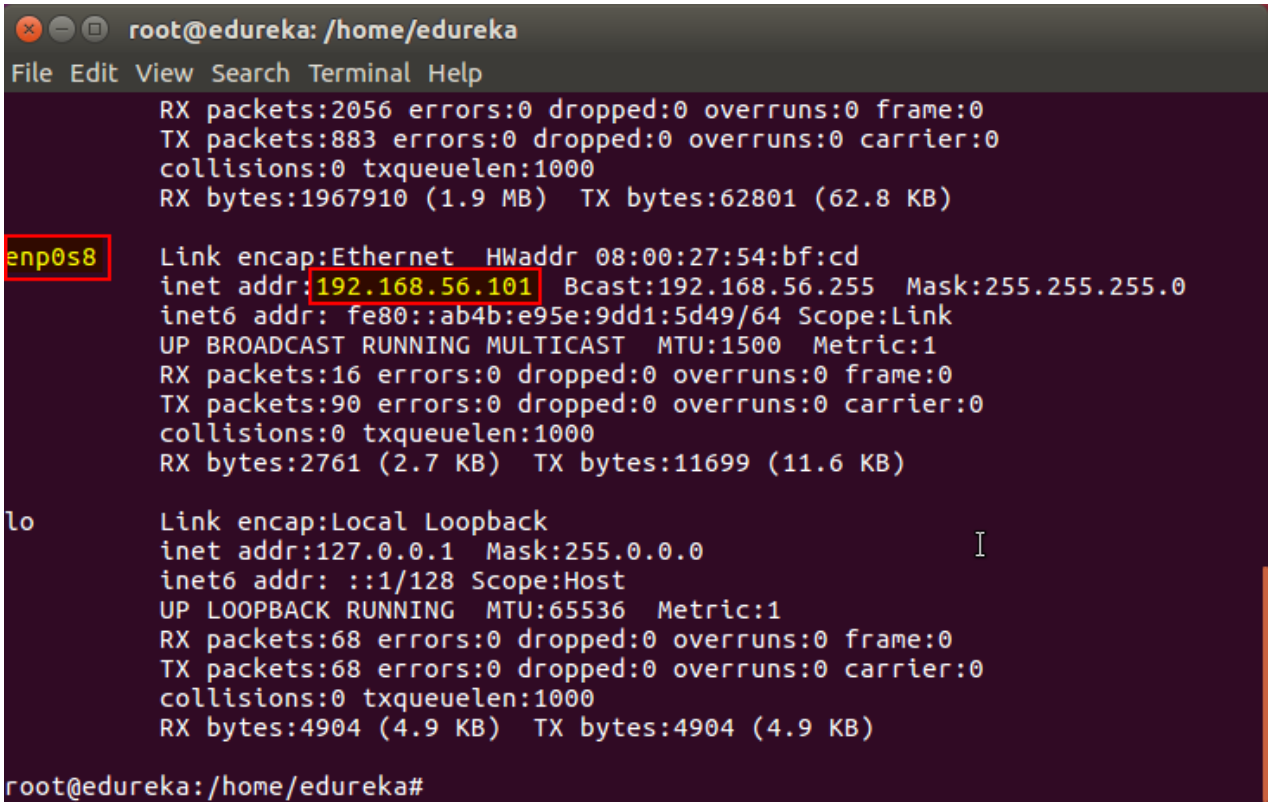
3. Press Ctrl+X, then press Y, and then press Enter to Save.

Step 3:

1. Run the following command:

```
ifconfig
```

2. Make a note of the IP address from the output of the above command. The IP address which has to be copied should be under “enp0s8”, as shown in the screenshot below.



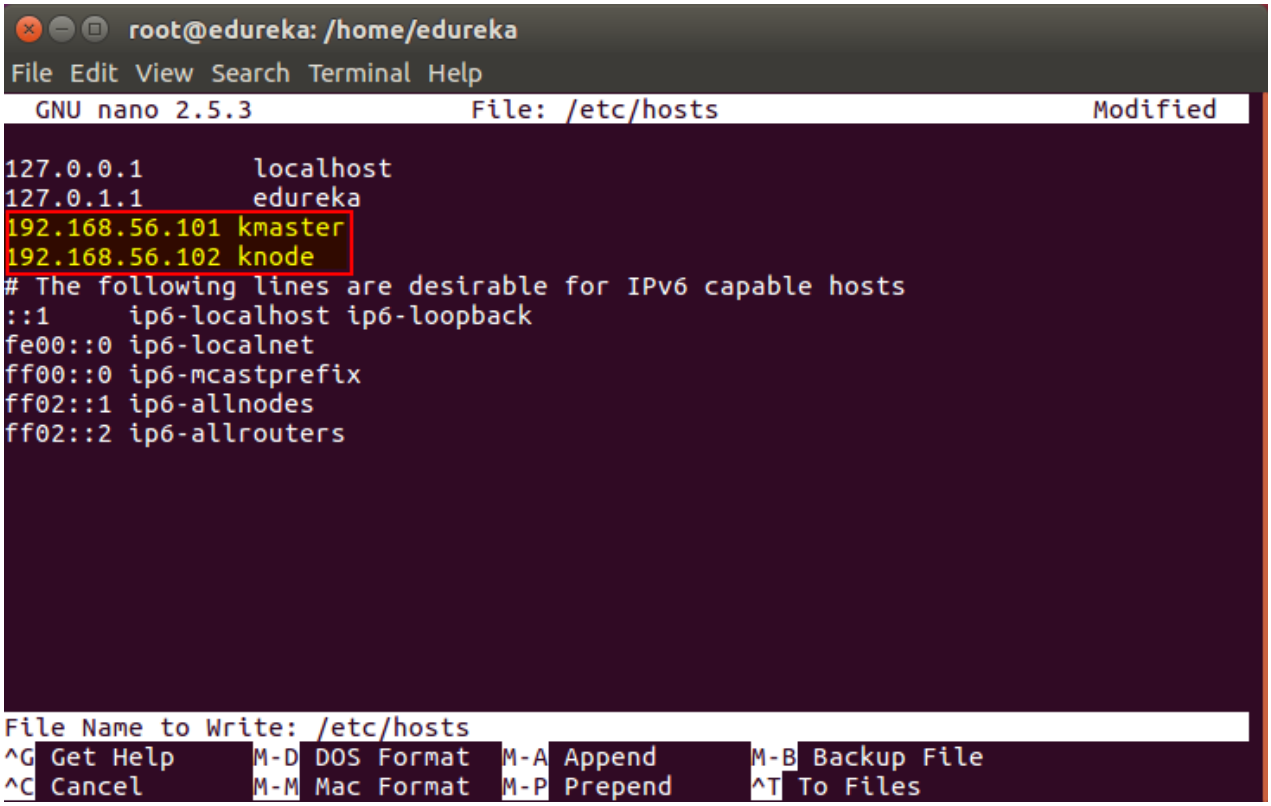
3. Make a note of the IP address of both the VMs using the above commands.

Step 4:

1. Run the following command:

```
nano /etc/hosts
```

2. Enter the IP address of the kmaster VM and the knode VM both in this file. (This has to be done in both the VMs). In front of the IP address of master write, “kmaster”. Similarly, in front of the Node IP address write “knode”.



3. Press Ctrl+X, then press Y, and then press Enter to Save.

Step 5: Next, we will make the IP addresses used above, static for the VMs.

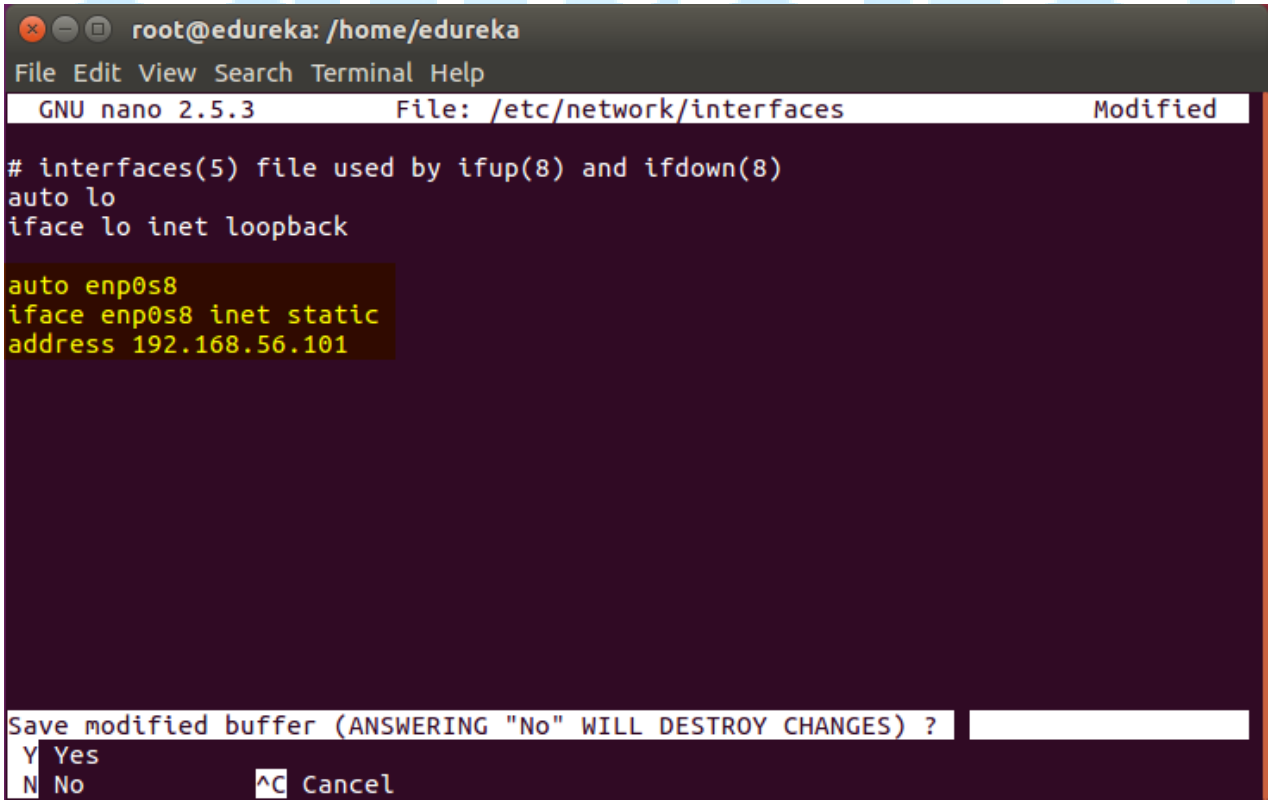
1. Run the following command:

```
nano /etc/network/interfaces
```

2. Enter the following in this document:

```
auto enp0s8
iface enp0s8 inet static
address <IP-Address-Of-VM>
```

3. It will look like the following:



4. Press Ctrl+X, then press Y, and then press Enter to Save.
5. After this, restart your machine.

Step 6: Now we will install openssh-server. Run the following command:

```
sudo apt-get install openssh-server
```

Step 7: Next, we will install Docker. Run the following commands:

```
sudo su
apt-get update
apt-get install -y docker.io
```

Step 8: Next, we will install kubeadm, kubelet and kubectl. Run the following commands:

```
apt-get update && apt-get install -y apt-transport-
https curl
```

```
curl -s
https://packages.cloud.google.com/apt/doc/apt-
key.gpg | apt-key add -
```

```
cat <<EOF >/etc/apt/sources.list.d/kubernetes.list
deb http://apt.kubernetes.io/ kubernetes-xenial main
EOF
```

```
apt-get update
apt-get install -y kubelet kubeadm kubectl
```

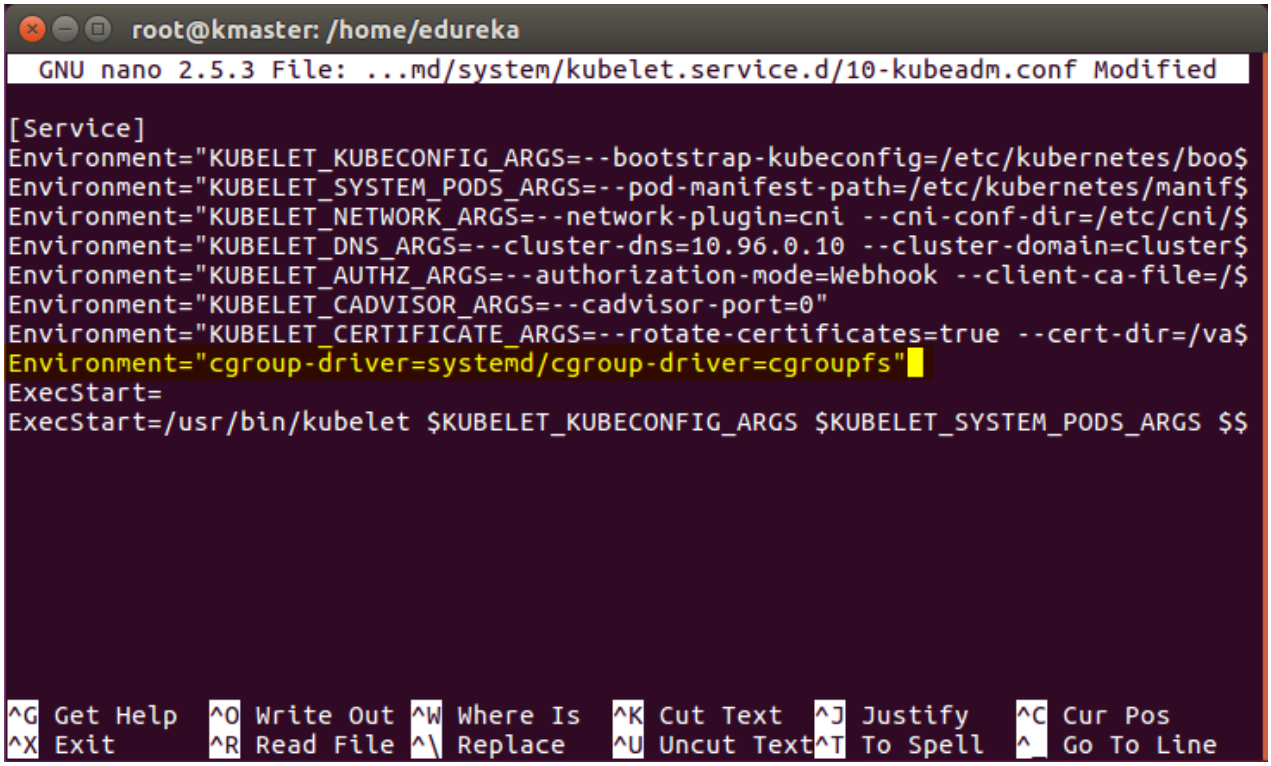
Step 9:

1. Next, we will change the configuration file of Kubernetes. Run the following command:

```
nano /etc/systemd/system/kubelet.service.d/10-kubeadm.conf
```

2. This will open a text editor, enter the following line after the last "Environment" Variable.

```
Environment="cgroup-driver=systemd/cgroup-driver=cgroupfs"
```



3. Press Ctrl+X, then press Y, and then press Enter to Save.

You have successfully installed Kubernetes on both the machines now!

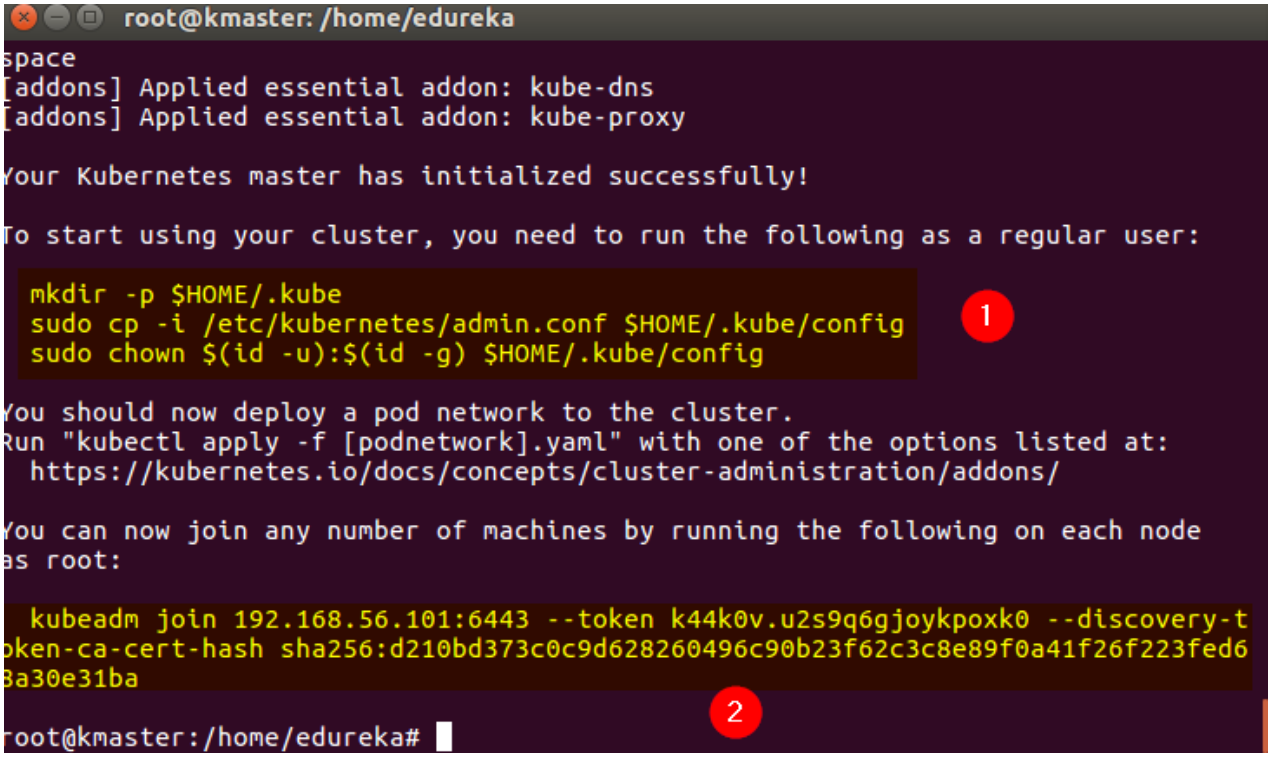
Steps for only Master VM

Note: These steps will only be executed on the master node (kmaster VM).

Step 1: We will now Initialize our Master VM. For that execute the following command:

```
kubeadm init --apiserver-advertise-address=<ip-address-of-kmaster-vm> --pod-network-cidr=192.168.0.0/16
```

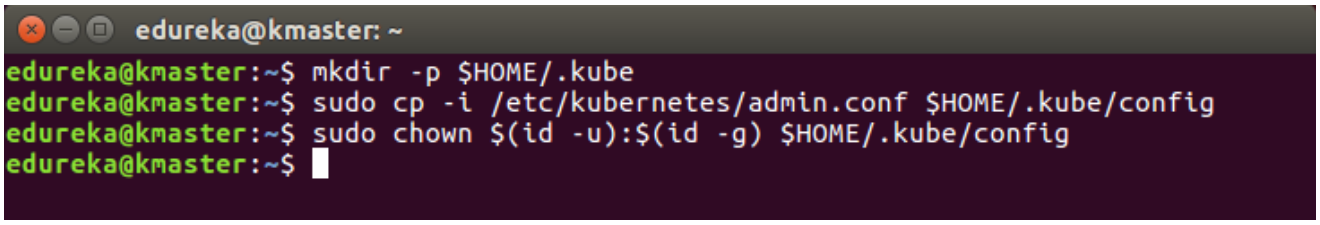
1. You will get the below output. The commands marked as (1), execute them as a non-root user. This will enable you to use kubectl from the CLI
2. The command marked as (2) should also be saved for future. This will be used to join nodes to your cluster.



Step 2:

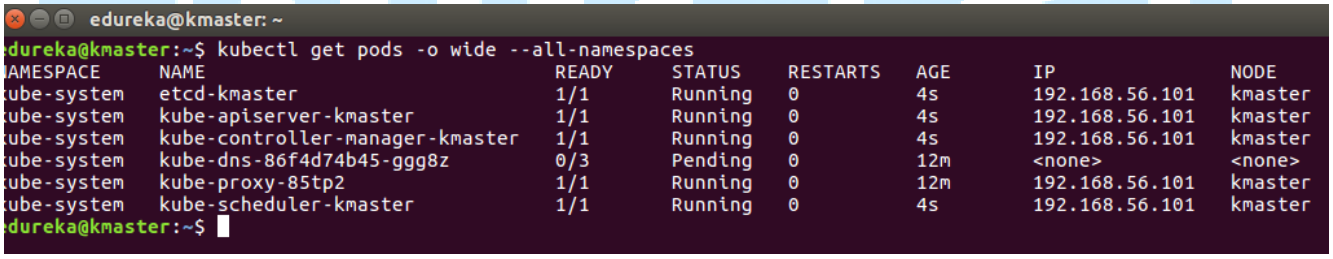
1. Like mentioned before, run the commands from the above output as a non-root user.

```
mkdir -p $HOME/.kube

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

sudo chown $(id -u):$(id -g) $HOME/.kube/config
```



2. To verify, if kubectl is working or not, run the following command:

```
kubectl get pods -o wide --all-namespaces
```

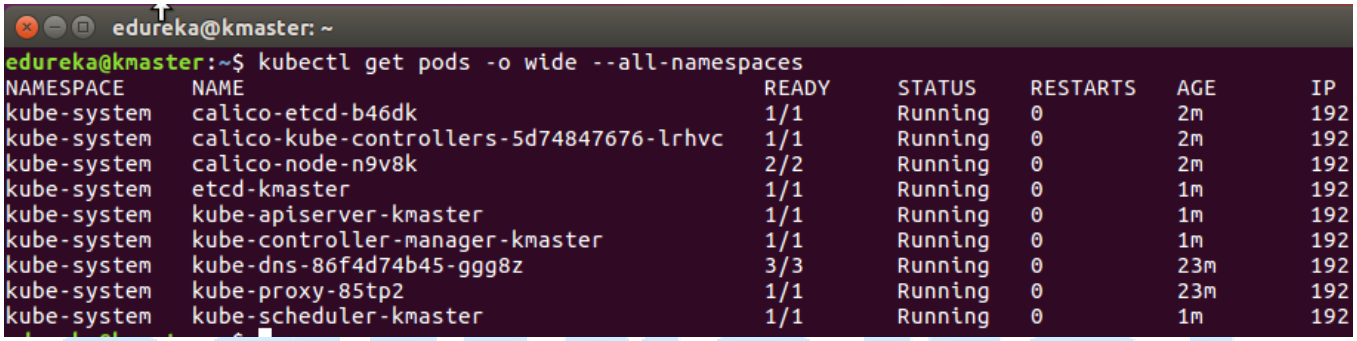


Step 3:

1. You will notice from the previous command, all the pods are running except one, kube-dns. For resolving this we will install a pod network. To install the pod network, run the following command:

```
kubectl apply -f
https://docs.projectcalico.org/v3.0/getting-
started/kubernetes/installation/hosted/kubeadm/1.7/calico.
yaml
```

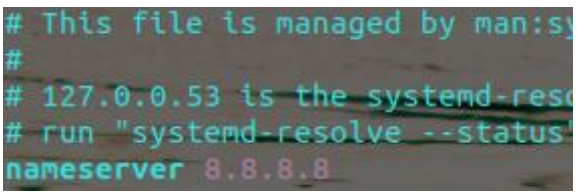
2. After some time, you will notice that all pods shift to the running state.



Troubleshooting:

1) If the internet stops working after installing the pod network or you get an image pull back error in pods, edit the **resolv.conf** file inside the **/etc** directory and change the nameserver to **8.8.8.8**

Syntax: sudo vi /etc/reslov.conf



The internet should work fine now.

2) In case calico doesn't start, use flannel instead as your pod network. To do that you'll need to restart the server with the flannel pod-network-cidr

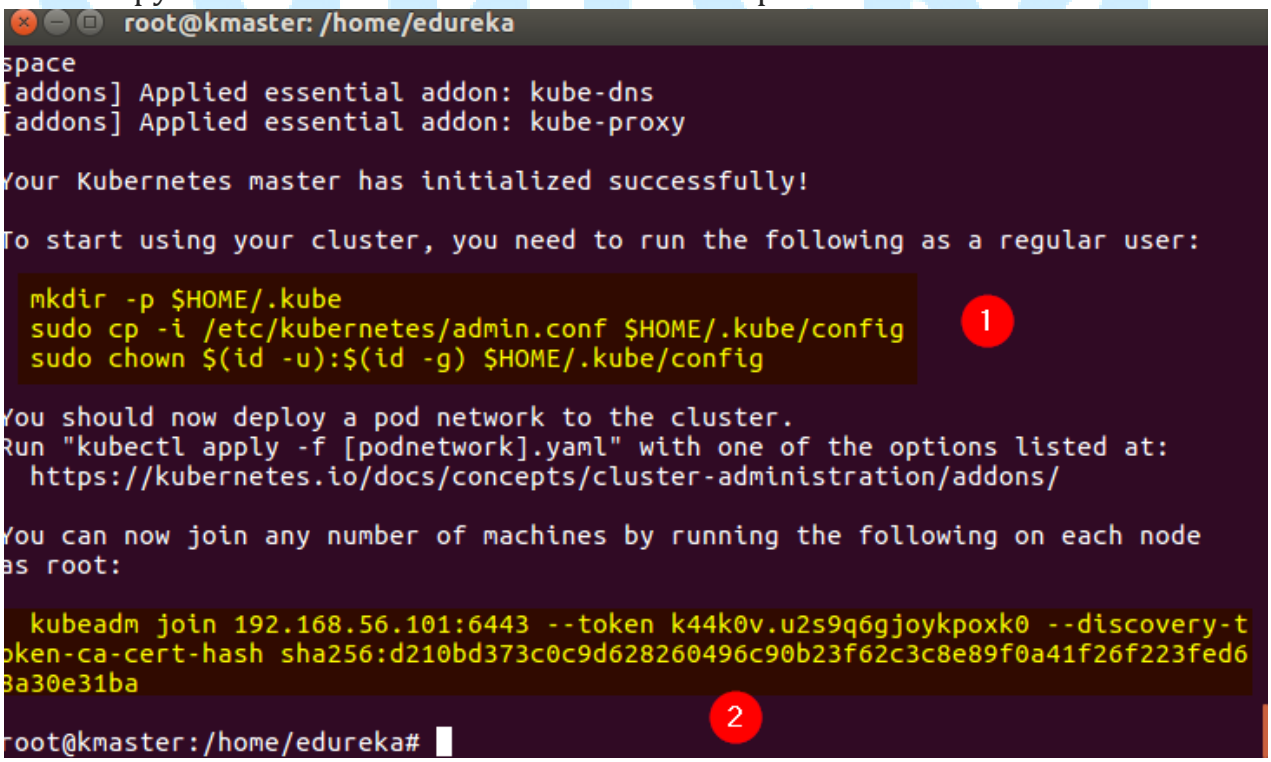
```
sudo kubeadm reset

sudo kubeadm init –apiserver-advertise-address=<kmaster-IP>
--pod-network-cidr=10.244.0.0/16
```

And use the following command to start the flannel pod network

```
kubectl apply -f
https://raw.githubusercontent.com/coreos/flannel/62e44c867a2
846fefb68bd5f178daf4da3095ccb/Documentation/kube-flannel.yml
```

Steps for only Node VM

Step 1: It is time to join your node to the cluster! This is probably the only step that you will be doing on the node, after installing kubernetes on it. Run the join command that you saved, when you ran kubeadm init command on the master.
Note: Copy and run the 2nd Command from this Step in Node VM.



```
edureka@knode:~$ sudo kubeadm join 192.168.56.101:6443 --token n6qrh0.opyhe2c655
ay3j04 --discovery-token-ca-cert-hash sha256:84dd965586c1b2d82b345706382ec43bc62
aa8e460b54dfc02b367f85f218b84
[sudo] password for edureka:
[preflight] Running pre-flight checks.
        [WARNING Service-Docker]: docker service is not enabled, please run 'sys
temctl enable docker.service'
        [WARNING FileExisting-crictl]: crictl not found in system path
Suggestion: go get github.com/kubernetes-incubator/cri-tools/cmd/crictl
[discovery] Trying to connect to API Server "192.168.56.101:6443"
[discovery] Created cluster-info discovery client, requesting info from "https:/
/192.168.56.101:6443"
[discovery] Requesting info from "https://192.168.56.101:6443" again to validate
 TLS against the pinned public key
[discovery] Cluster info signature and contents are valid and TLS certificate va
lidates against pinned roots, will use API Server "192.168.56.101:6443"
[discovery] Successfully established connection with API Server "192.168.56.101:
6443"

This node has joined the cluster:
```

Your Kubernetes Cluster is ready! :-)