

Kubernetes Cluster on Ubuntu VMs

Installation Guide

edureka!

edureka!

© Brain4ce Education Solutions Pvt. Ltd.

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:

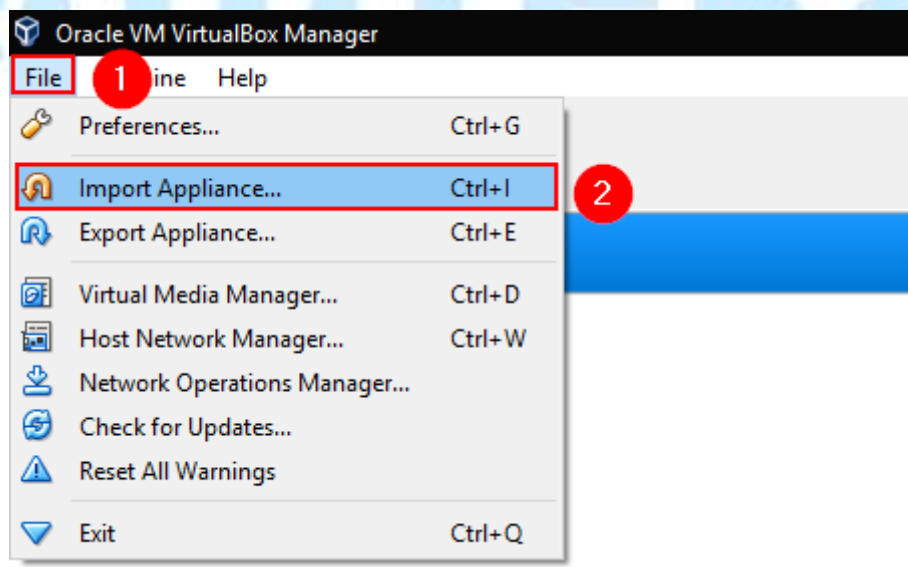
- 1 GB RAM
- 1 Core of CPU

Importing the Clean Ubuntu VM

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

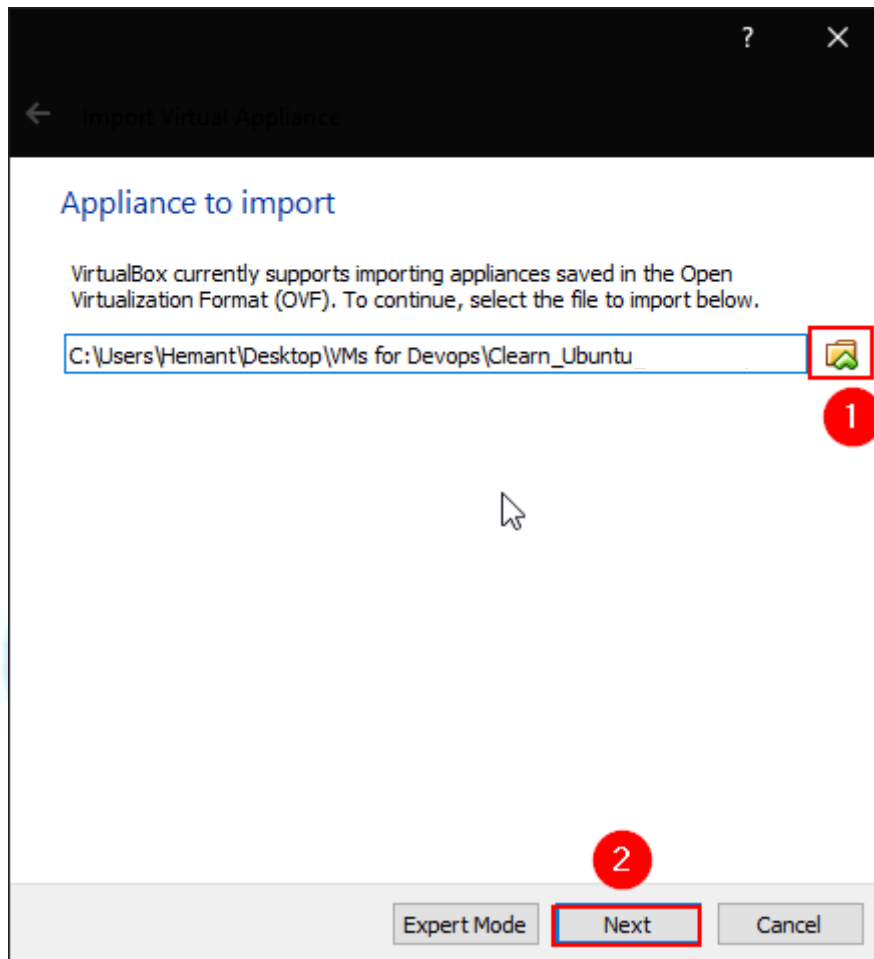
Step 1:

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



Step 2:

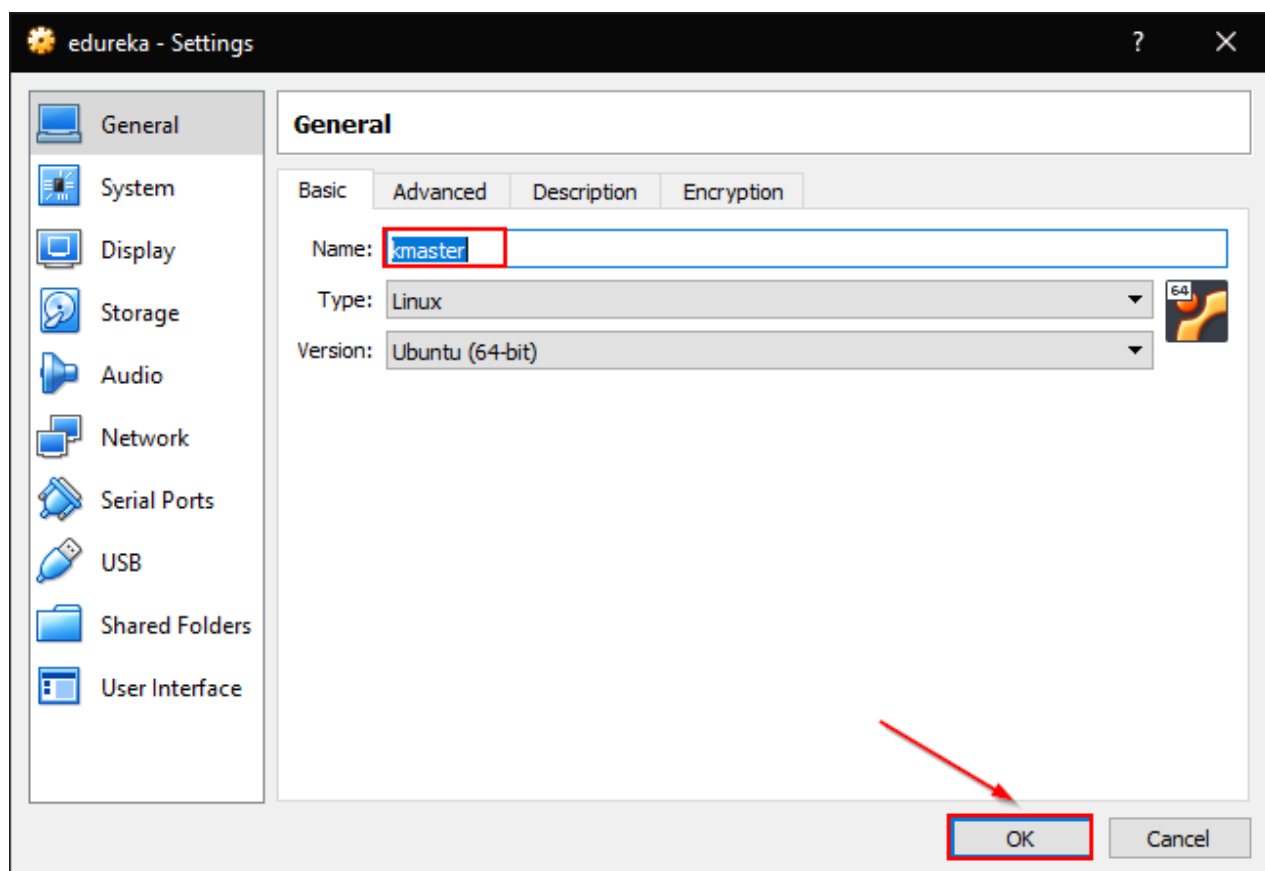
1. Click on Browse, and navigate to the place where you have downloaded the VM and select it.
2. Click on Next, and on the next page leave everything at default and Click on Import.



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

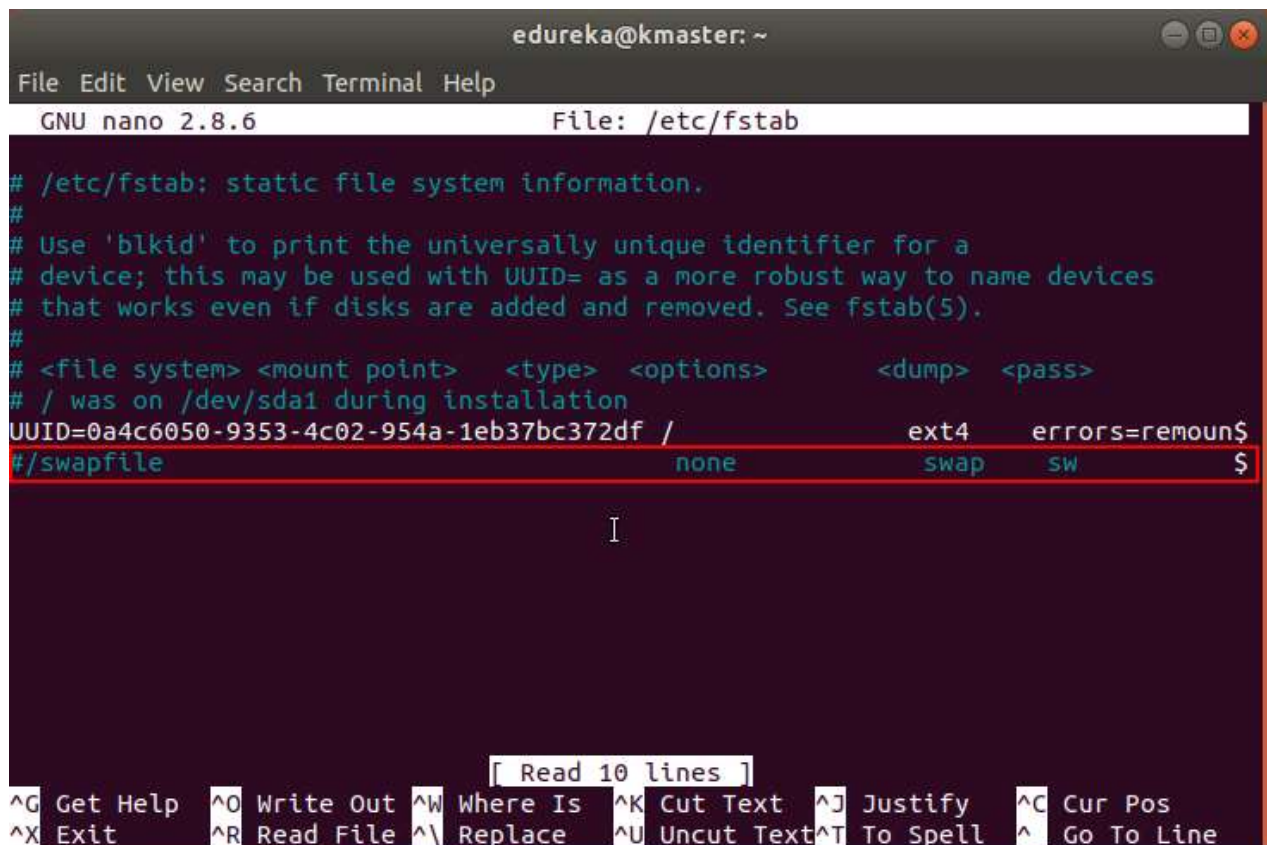
Note: These steps are common to both kmaster and knode VMs

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.



```
edureka@kmaster: ~
File Edit View Search Terminal Help
GNU nano 2.8.6 File: /etc/fstab
# /etc/fstab: static file system information.
#
# Use 'blkid' to print the universally unique identifier for a
# device; this may be used with UUID= as a more robust way to name devices
# that works even if disks are added and removed. See fstab(5).
#
# <file system> <mount point> <type> <options> <dump> <pass>
# / was on /dev/sda1 during installation
UUID=0a4c6050-9353-4c02-954a-1eb37bc372df / ext4 errors=remoun$
#/swapfile none swap sw $
I
[ Read 10 lines ]
^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos
^X Exit ^R Read File ^\ Replace ^U Uncut Text ^T To Spell ^_ Go To Line
```

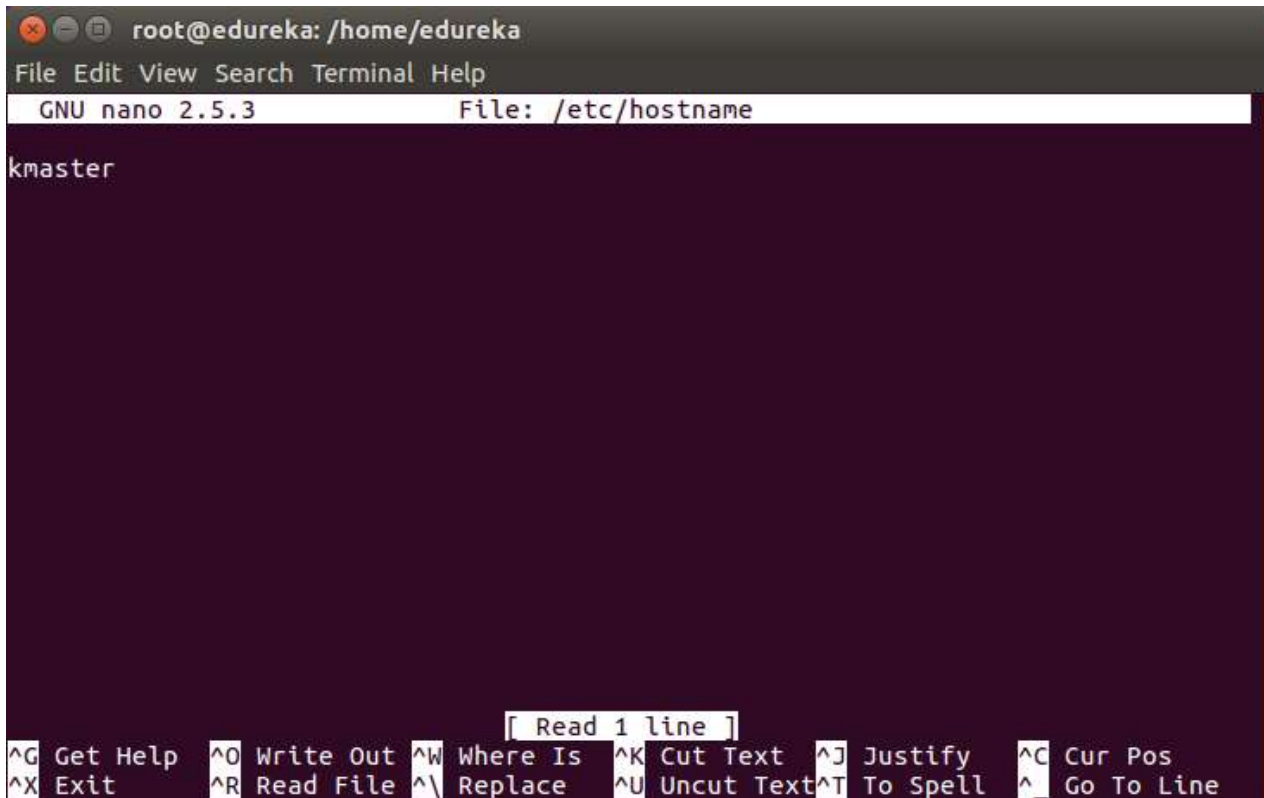
3. Press Ctrl+X, then press Y, and then press Enter to Save.

Step 2:

1. Run the following command:

```
nano /etc/hostname
```

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



```
root@edureka: /home/edureka
File Edit View Search Terminal Help
GNU nano 2.5.3 File: /etc/hostname
kmaster

[ Read 1 line ]
^G Get Help  ^O Write Out ^W Where Is  ^K Cut Text  ^J Justify   ^C Cur Pos
^X Exit      ^R Read File ^\ Replace   ^U Uncut Text ^T To Spell  ^_ Go To Line
```

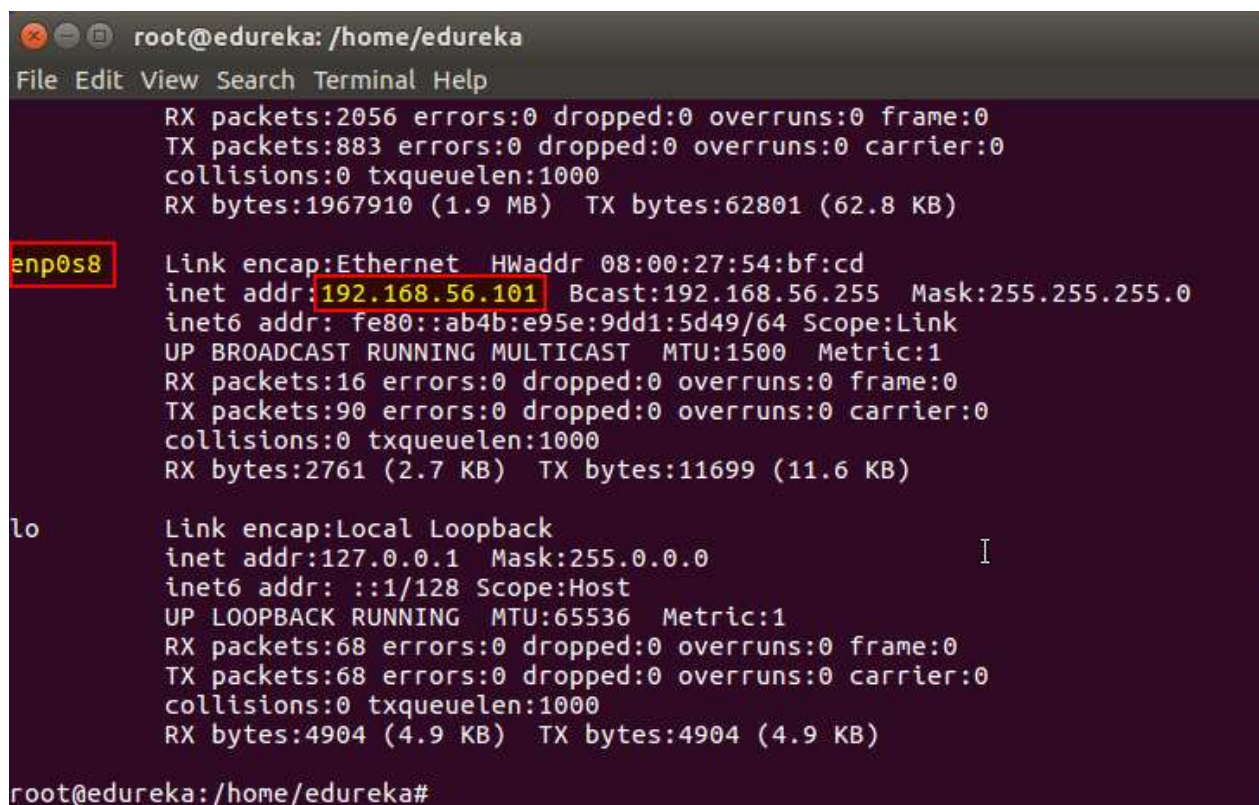
3. Press Ctrl+X, then press Y, and then press Enter to Save.

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.



```
root@edureka: /home/edureka
File Edit View Search Terminal Help
RX packets:2056 errors:0 dropped:0 overruns:0 frame:0
TX packets:883 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:1967910 (1.9 MB) TX bytes:62801 (62.8 KB)

enp0s8  Link encap:Ethernet HWaddr 08:00:27:54:bf:cd
        inet addr:192.168.56.101 Bcast:192.168.56.255 Mask:255.255.255.0
        inet6 addr: fe80::ab4b:e95e:9dd1:5d49/64 Scope:Link
        UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
        RX packets:16 errors:0 dropped:0 overruns:0 frame:0
        TX packets:90 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1000
        RX bytes:2761 (2.7 KB) TX bytes:11699 (11.6 KB)

lo      Link encap:Local Loopback
        inet addr:127.0.0.1 Mask:255.0.0.0
        inet6 addr: ::1/128 Scope:Host
        UP LOOPBACK RUNNING MTU:65536 Metric:1
        RX packets:68 errors:0 dropped:0 overruns:0 frame:0
        TX packets:68 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1000
        RX bytes:4904 (4.9 KB) TX bytes:4904 (4.9 KB)

root@edureka: /home/edureka#
```

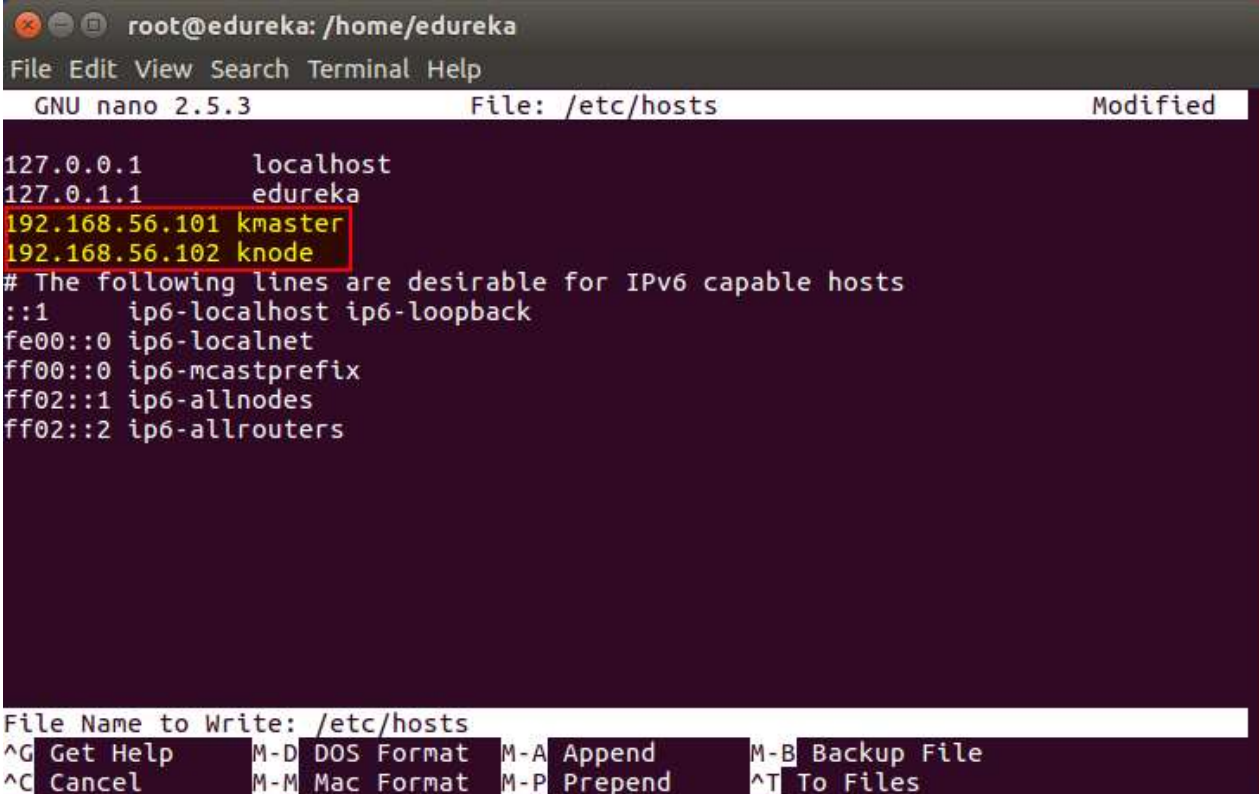
3. Make a note of the IP address of both the VMs using the above commands.

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".



```
root@edureka: /home/edureka
File Edit View Search Terminal Help
GNU nano 2.5.3 File: /etc/hosts Modified
127.0.0.1 localhost
127.0.1.1 edureka
192.168.56.101 kmaster
192.168.56.102 knode
# The following lines are desirable for IPv6 capable hosts
::1 ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
File Name to Write: /etc/hosts
^G Get Help M-D DOS Format M-A Append M-B Backup File
^C Cancel M-M Mac Format M-P Prepend ^T To Files
```

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.

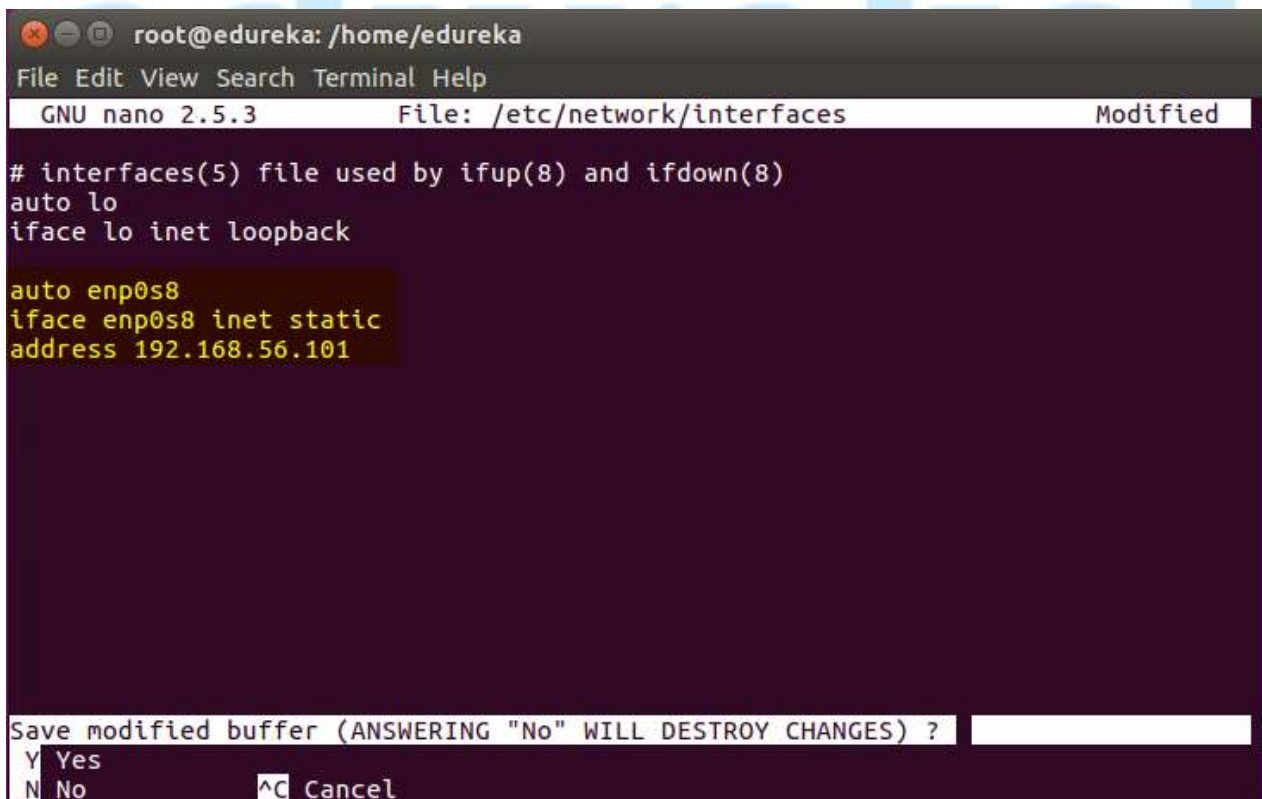
1. Run the following command:

```
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:



```
root@edureka: /home/edureka  
File Edit View Search Terminal Help  
GNU nano 2.5.3 File: /etc/network/interfaces Modified  
# interfaces(5) file used by ifup(8) and ifdown(8)  
auto lo  
iface lo inet loopback  
  
auto enp0s8  
iface enp0s8 inet static  
address 192.168.56.101  
  
Save modified buffer (ANSWERING "No" WILL DESTROY CHANGES) ?  
Y Yes  
N No ^C Cancel
```

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 kubect. 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
```

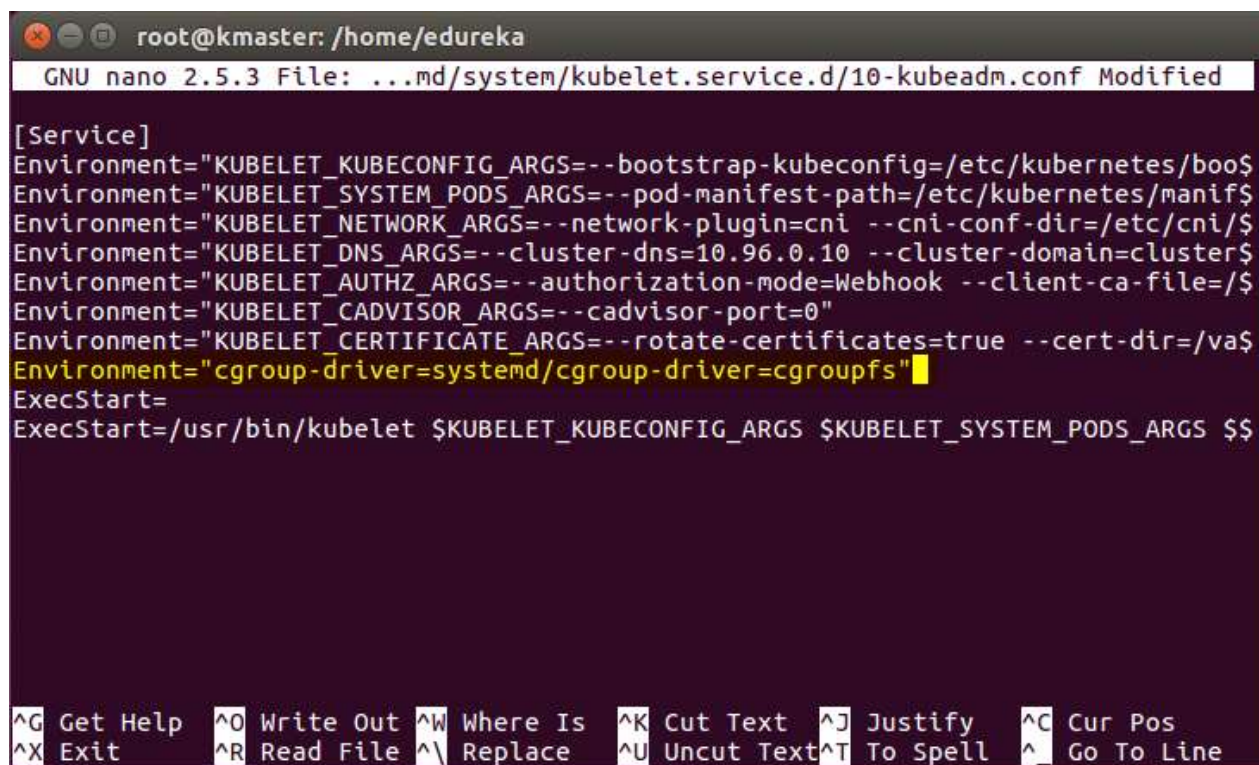
Step 9:

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"
```



```
root@kmaster: /home/edureka
GNU nano 2.5.3 File: ...md/system/kubelet.service.d/10-kubeadm.conf Modified

[Service]
Environment="KUBELET_KUBECONFIG_ARGS=--bootstrap-kubeconfig=/etc/kubernetes/boo$
Environment="KUBELET_SYSTEM_PODS_ARGS=--pod-manifest-path=/etc/kubernetes/manif$
Environment="KUBELET_NETWORK_ARGS=--network-plugin=cni --cni-conf-dir=/etc/cni/$
Environment="KUBELET_DNS_ARGS=--cluster-dns=10.96.0.10 --cluster-domain=cluster$
Environment="KUBELET_AUTHZ_ARGS=--authorization-mode=Webhook --client-ca-file=$
Environment="KUBELET_CADVISOR_ARGS=--cadvisor-port=0"
Environment="KUBELET_CERTIFICATE_ARGS=--rotate-certificates=true --cert-dir=/va$
Environment="cgroup-driver=systemd/cgroup-driver=cgroupfs"
ExecStart=
ExecStart=/usr/bin/kubelet $KUBELET_KUBECONFIG_ARGS $KUBELET_SYSTEM_PODS_ARGS $$

^G Get Help  ^O Write Out  ^W Where Is  ^K Cut Text  ^J Justify   ^C Cur Pos
^X Exit      ^R Read File  ^\ Replace   ^U Uncut Text ^T To Spell  ^_ Go To Line
```

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

Step 10: Restart your VMs for the changes to take effect.

You have successfully installed Kubernetes on both the machines now!

edureka!

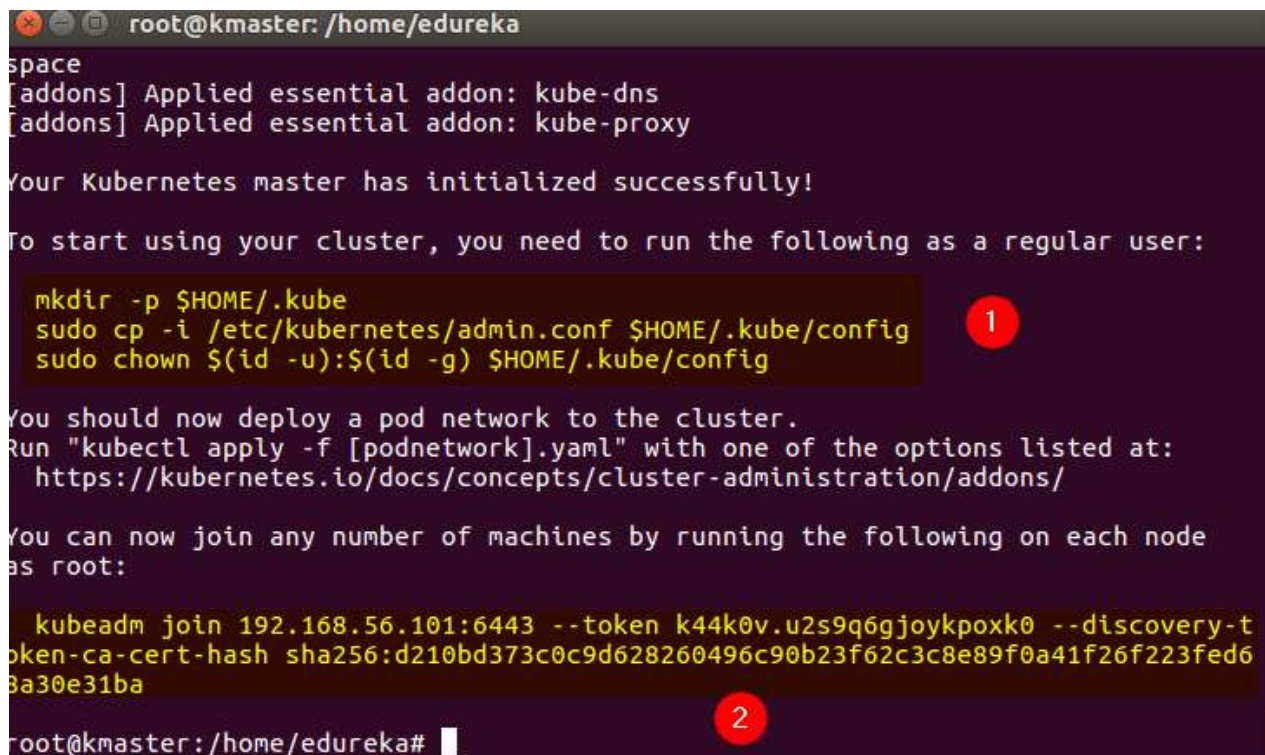
Steps for only Master VM

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.



```
root@kmaster: /home/edureka
space
[addons] Applied essential addon: kube-dns
[addons] Applied essential addon: kube-proxy

Your Kubernetes master has initialized successfully!

To start using your cluster, you need to run the following as a regular user:

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

You should now deploy a pod network to the cluster.
Run "kubectl apply -f [podnetwork].yaml" with one of the options listed at:
  https://kubernetes.io/docs/concepts/cluster-administration/addons/

You can now join any number of machines by running the following on each node
as root:

  kubeadm join 192.168.56.101:6443 --token k44k0v.u2s9q6gjoykpoxk0 --discovery-t
oken-ca-cert-hash sha256:d210bd373c0c9d628260496c90b23f62c3c8e89f0a41f26f223fed6
3a30e31ba
```

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
```

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

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

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

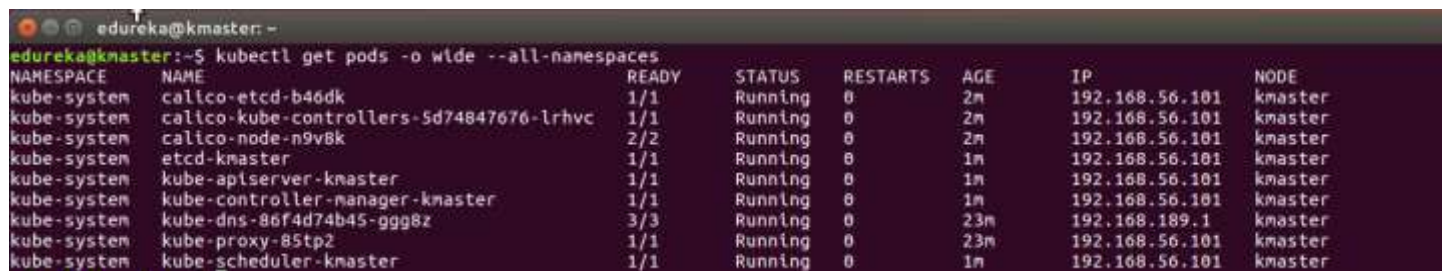
```
edureka@kmaster: ~  
edureka@kmaster:~$ kubectl get pods -o wide --all-namespaces  
NAMESPACE   NAME                                     READY   STATUS    RESTARTS   AGE   IP             NODE  
kube-system  etcd-kmaster                           1/1     Running   0           4s    192.168.56.101 kmaster  
kube-system  kube-apiserver-kmaster                 1/1     Running   0           4s    192.168.56.101 kmaster  
kube-system  kube-controller-manager-kmaster        1/1     Running   0           4s    192.168.56.101 kmaster  
kube-system  kube-dns-86f4d74b45-ggg8z             0/3     Pending   0          12m    <none>         <none>  
kube-system  kube-proxy-85tp2                      1/1     Running   0          12m    192.168.56.101 kmaster  
kube-system  kube-scheduler-kmaster                 1/1     Running   0           4s    192.168.56.101 kmaster  
edureka@kmaster:~$
```


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/v3.11/manifests/calico.yaml
```

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



```
edureka@kmaster:~$ kubectl get pods -o wide --all-namespaces
```

| NAMESPACE | NAME | READY | STATUS | RESTARTS | AGE | IP | NODE |
|-------------|--|-------|---------|----------|-----|----------------|---------|
| kube-system | calico-etcd-b46dk | 1/1 | Running | 0 | 2m | 192.168.56.101 | kmaster |
| kube-system | calico-kube-controllers-5d74847676-lrhvc | 1/1 | Running | 0 | 2m | 192.168.56.101 | kmaster |
| kube-system | calico-node-n9v8k | 2/2 | Running | 0 | 2m | 192.168.56.101 | kmaster |
| kube-system | etcd-kmaster | 1/1 | Running | 0 | 1m | 192.168.56.101 | kmaster |
| kube-system | kube-apiserver-kmaster | 1/1 | Running | 0 | 1m | 192.168.56.101 | kmaster |
| kube-system | kube-controller-manager-kmaster | 1/1 | Running | 0 | 1m | 192.168.56.101 | kmaster |
| kube-system | kube-dns-86f4d74b45-ggg8z | 3/3 | Running | 0 | 23m | 192.168.189.1 | kmaster |
| kube-system | kube-proxy-85tpz | 1/1 | Running | 0 | 23m | 192.168.56.101 | kmaster |
| kube-system | kube-scheduler-kmaster | 1/1 | Running | 0 | 1m | 192.168.56.101 | kmaster |

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/resolv.conf`

```
edureka@kmaster:/etc$ sudo vi resolv.conf
```

```
# This file is managed by man:systemd-resolved
#
# 127.0.0.53 is the systemd-resolved stub DNS
# run "systemd-resolve --status"
nameserver 8.8.8.8
```

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/2140ac876ef
134e0ed5af15c65e414cf26827915/Documentation/kube-flannel.yml
```

Step 4: Next, we will install the dashboard. To install the Dashboard, run the following command:

```
kubectl create -f
https://raw.githubusercontent.com/kubernetes/dashboard/mas
ter/src/ deploy/recommended/kubernetes-dashboard.yml
```

```
edureka@kmaster:~$ kubectl create -f https://raw.githubusercontent.com/kuberne
-dashboards.yml
secret "kubernetes-dashboard-certs" created
serviceaccount "kubernetes-dashboard" created
role.rbac.authorization.k8s.io "kubernetes-dashboard-minimal" created
rolebinding.rbac.authorization.k8s.io "kubernetes-dashboard-minimal" created
deployment.apps "kubernetes-dashboard" created
service "kubernetes-dashboard" created
```

Step 5: Your dashboard is now ready with it's the pod in the running state.

| | | | | |
|-------------|---------------------------------------|-----|---------|---|
| kube-system | etcd-kmaster | 1/1 | Running | 0 |
| kube-system | kube-apiserver-kmaster | 1/1 | Running | 0 |
| kube-system | kube-controller-manager-kmaster | 1/1 | Running | 0 |
| kube-system | kube-dns-86f4d74b45-ggg8z | 3/3 | Running | 0 |
| kube-system | kube-proxy-85tp2 | 1/1 | Running | 0 |
| kube-system | kube-scheduler-kmaster | 1/1 | Running | 0 |
| kube-system | kubernetes-dashboard-7d5dcdb6d9-bbmmr | 1/1 | Running | 0 |

Step 6:

1. By default dashboard will not be visible on the Master VM. Run the following command in the command line:

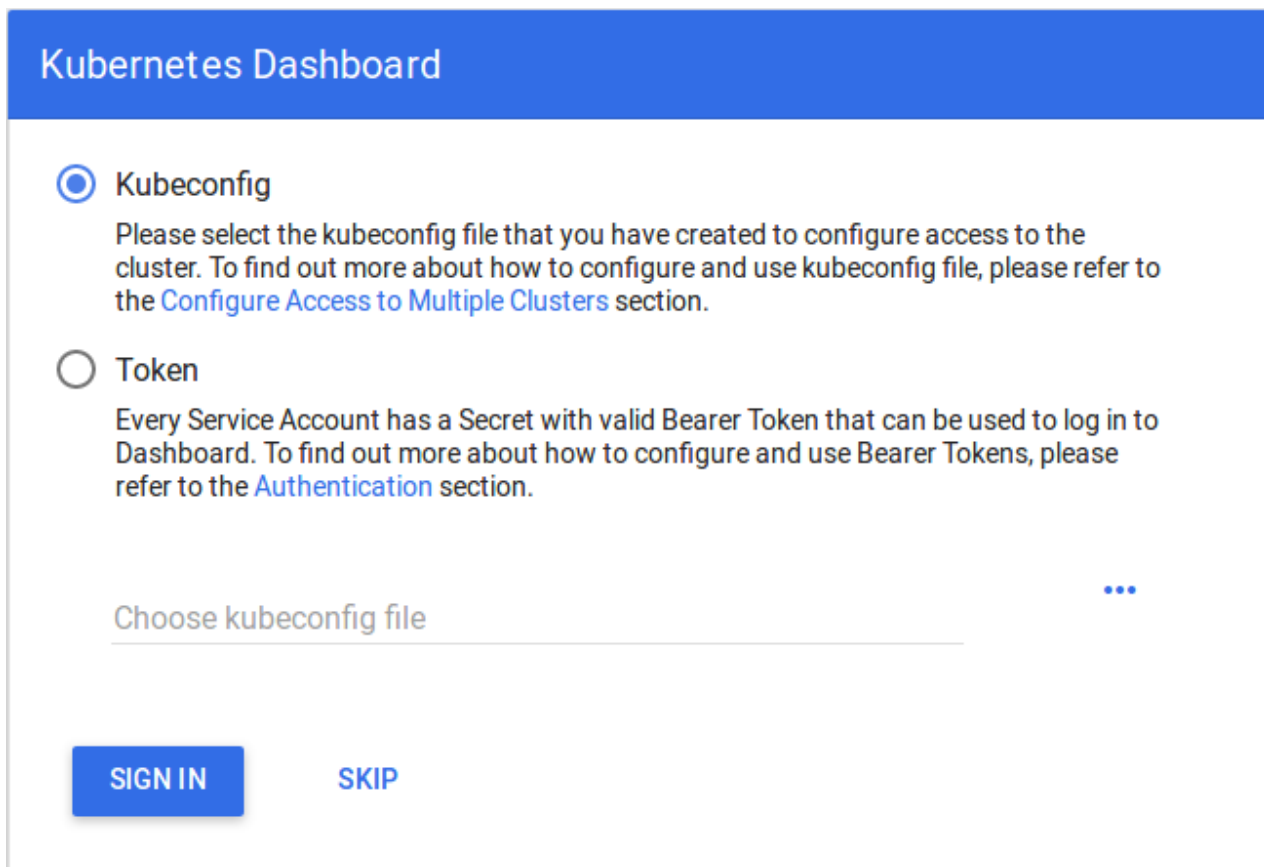
```
kubectl proxy
```

```
edureka@kmaster:~$ kubectl proxy
Starting to serve on 127.0.0.1:8001
```


2. To view the dashboard in the browser, navigate to the following address in the browser of your Master VM.

```
http://localhost:8001/api/v1/namespaces/kube-system/services/https:kubernetes-dashboard:/proxy/
```

3. You will be prompted with this page, to enter the credentials.

The image shows the Kubernetes Dashboard login interface. At the top is a blue header with the text 'Kubernetes Dashboard'. Below the header, there are two radio button options. The first option, 'Kubeconfig', is selected and includes a description: 'Please select the kubeconfig file that you have created to configure access to the cluster. To find out more about how to configure and use kubeconfig file, please refer to the [Configure Access to Multiple Clusters](#) section.' The second option, 'Token', is unselected and includes a description: 'Every Service Account has a Secret with valid Bearer Token that can be used to log in to Dashboard. To find out more about how to configure and use Bearer Tokens, please refer to the [Authentication](#) section.' Below these options is a text input field labeled 'Choose kubeconfig file' with a dropdown arrow icon to its right. At the bottom left is a blue 'SIGN IN' button, and at the bottom right is a 'SKIP' link.

Step 7: In this step, we will create the service account for the dashboard and get its credentials. Run the following commands:

Note: Run all these commands in a new terminal, or your kubectl proxy command will stop.

1. This command will create service account for dashboard in the default namespace.

```
kubectl create serviceaccount dashboard -n default
```

2. This command will add the cluster binding rules to your dashboard account

3. This command will give you the token required for your dashboard login.

```
edureka@kmaster:~$ kubectl get secret $(kubectl get serviceaccount dashboard -o jsonpath="{.secrets[0].name}") -o jsonpath="{.data.token}" | base64 --decode
eyJhbGciOiJIUzI1NiIsImtpZCI6IiJ9.eyJpc3MiOiJrdWJlcm5ldGVzL3NlcnZpY2VhY2NvdW50Iiwia3ViZXJuZXRlcy5pby9zZXJ2aWVudC9uYWw1lc3BhY2UiOiJkZWZhdWx0Iiwia3ViZXJuZXRlcy5pby9zZXJ2aWVudC9zZWNyZXQubmFtZSI6ImRhc2hib2FyZC10b2t1bi1iOTRocSIiImt1YmVybmV0ZXMuaW8vc2VydmljZWZjY291bnQvc2VydmljZS1hY2NvdW50Lm5hbWUiOiJkYXNoYm9hcmQ1LlCjrdWJlcm5ldGVzLmVlL3NlcnZpY2VhY2NvdW50L3NlcnZpY2UtYWNjb3VudC51awQioiJhYWY1YzL1mS01YWE0LExTgtGTgtOGY3YS0wODAwMjdmODlkZWQ1LlCjZldWIoOiJjeXN0ZW06c2VydmljZWZjY291bnQ6ZGVmYXVsdDpkYXNoYm9hcmQifQ.WKPKldojENDMJ4L74LhQNCIHQ2Gs2jUlo0vYdk4pkU4vN8IB54x7I9BqQYUiuJw_zEZqjnwYQdjndu2DAMtXwC_5uILO4SaTtL_bVARvb0OvCxxELaUyHqfppzEL8-EJNXGUiUqzvyYr8zkYRTAqTicbj3tXBLICrg5Ru-moN7IdPxXwaerWjdJWiH96h_VRmO5myiCoX_gTBHztWQ00sdgOWuFf2fTodCO-e516vxBzN0ThKdzGKBE2m7FenWxCCLTKzWuHUUK6yZuJq_vDpON1P7ARqQYnwXjh6eHzKgJ9b8rf41D6m6DmlS0vgd0SCPfwjkZ_ppv_tl-XVPdTQ
```

- Copy this token and paste it in Dashboard Login Page, by selecting token option.

Kubernetes Dashboard

☐ Kubeconfig
Please select the kubeconfig file that you have created to configure access to the cluster. To find out more about how to configure and use kubeconfig file, please refer to the [Configure Access to Multiple Clusters](#) section.


☒ Token
Every Service Account has a Secret with valid Bearer Token that can be used to log in to Dashboard. To find out more about how to configure and use Bearer Tokens, please refer to the [Authentication](#) section.

Enter token
.....

SIGN IN

SKIP

- You have successfully logged in your dashboard!

 **kubernetes**

Search

Overview

Cluster


- Namespaces
- Nodes
- Persistent Volumes
- Roles
- Storage Classes

Namespace

- default

Discovery and Load Balancing

Services

| Name | Labels | Cluster IP | Internal endpoints |
|---|--|------------|--|
|  kubernetes | component: apiserver provider: kubernetes | 10.96.0.1 | kubernetes:443 TCP kubernetes:0 TCP |

Config and Storage

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: Run this command with “sudo”.

```
sudo kubeadm join --apiserver-advertise-address=<ip-  
address-of-the master> --pod-network-cidr=192.168.0.0/16
```

```
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! :-)