How To Install Kubernetes Cluster On Ubuntu 16.04

Last updated on Nov 25,2020 *87.5K Views*

Hemant Sharma

edureka!
Find Out How Netflix Recommends You A Movie?
A Free AI-ML Workshop by Edureka
26th & 27th December
6 PM - 8 PM IST / 7:30 AM - 9:30 AM ET
REGISTER NOW
*Limited seats available

The first step towards ***Kubernetes Certification*** is installing Kubernetes. This blog is a step by step guide to install Kubernetes on top of Ubuntu VMs (Virtual Machines). Here, one VM will act as the master and the other VM will be the node. You can then replicate the same steps to deploy the Kubernetes cluster onto your prod.

Note*: For this installation, we recommend a fresh Ubuntu 16.04 image since Kubernetes can take up a lot of resources. If your* ***installation fails at any time****, then* ***execute all the steps mentioned*** *from the very beginning in a fresh VM, because debugging would take longer.*

To install Kubernetes, you have to diligently follow the 3 phases that come as part of the installation process:

1. Pre-requisites to install Kubernetes
2. Setting up Kubernetes environment
3. Installing Kubeadm, Kubelet, Kubectl
4. Starting the Kubernetes cluster from master
5. Getting the nodes to join the cluster

Pre-requisites To Install Kubernetes

Since we are dealing with VMs, we recommend the following settings for the VMs:-

Master:

- 2 GB RAM
- 2 Cores of CPU

Slave/ Node:

- 1 GB RAM
- 1 Core of CPU

By this point of time, I have assumed you have 2 plain Ubuntu VMs imported onto your Oracle Virtual Box. So, I'l just get along with the installation process.

Pre-Installation Steps On Both Master & Slave (To Install Kubernetes)

The following steps have to be executed on both the master and node machines. Let's call the the master as '*kmaster*' and node as '*knode*'.

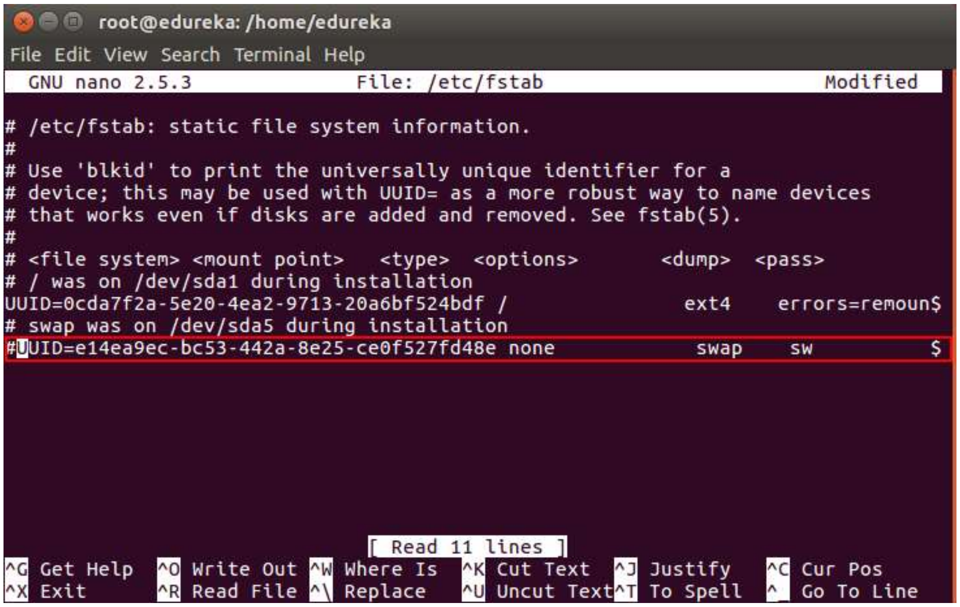
First, login as 'sudo' user because the following set of commands need to be executed with 'sudo' permissions. Then, update your 'apt-get' repository.

```
$ sudo su
# apt-get update
```

Note: After logging-in as 'sudo' user, note that your shell symbol will change to '#' from '$'.

Turn Off Swap Space

```
# swapoff -a
# nano /etc/fstab
```



root@edureka: /home/edureka
File Edit View Search Terminal Help

```
  GNU nano 2.5.3              File: /etc/fstab                        Modified

# /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=0cda7f2a-5e20-4ea2-9713-20a6bf524bdf /               ext4    errors=remoun$
# swap was on /dev/sda5 during installation
#UUID=e14ea9ec-bc53-442a-8e25-ce0f527fd48e none            swap    sw          $




                                [ Read 11 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
```

Then press ‘Ctrl+X’, then press ‘Y’ and then press ‘Enter’ to Save the file.

Update The Hostnames

To change the hostname of both machines, run the below command to open the file and subsequently rename the master machine to ‘kmaster’ and your node machine to ‘knode’.

```
# nano /etc/hostname
```

Run the following command on both machines to note the IP addresses of each.

```
# ifconfig
```

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.

Now go to the ‘hosts’ file on both the master and node and add an entry specifying their respective IP addresses along with their names ‘kmaster’ and ‘knode’. This is used for referencing them in the cluster. It should look like the below screenshot on both the machines.

```
# nano /etc/hosts
```

Then press ‘Ctrl+X’, then press ‘Y’ and then press ‘Enter’ to Save the file.

Setting Static IP Addresses

Next, we will make the IP addresses used above, static for the VMs. We can do that by modifying the network interfaces file. Run the following command to open the file:

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

Now enter the following lines in the file.

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

It will look something like the below screenshot.

Then press ‘Ctrl+X’, then press ‘Y’ and then press ‘Enter’ to Save the file.

After this, restart your machine(s).

Install OpenSSH-Server

Now we have to install openshh-server. Run the following command:

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

Install Docker

Now we have to install Docker because Docker images will be used for managing the containers in the cluster. Run the following commands:

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

Next we have to install these 3 essential components for setting up Kubernetes environment: kubeadm, kubectl, and kubelet.

Run the following commands before installing the Kubernetes environment.

```
# 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
```

Install kubeadm, Kubelet And Kubectl

Now its time to install the 3 essential components. ***Kubelet*** is the lowest level component in Kubernetes. It’s responsible for what’s running on an individual machine. ***Kuebadm*** is used for administrating the Kubernetes cluster. ***Kubectl*** is used for controlling the configurations on various nodes inside the cluster.

```
# apt-get install -y kubelet kubeadm kubectl
# nano /etc/systemd/system/kubelet.service.d/10-kubeadm.conf
```

This will open a text editor, enter the following line after the last “Environment Variable”:

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

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

Voila! You have successfully installed Kubernetes on both the machines now!

As of now, only the Kubernetes environment has been setup. But now, it is time to install Kubernetes completely, by moving onto the next 2 phases, where we will individually set the configurations in both machines.

Steps Only For Kubernetes Master VM (kmaster)

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

Step 1: We will now start our Kubernetes cluster from the master’s machine. Run 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: As 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
```

It should look like this:

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

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

Step 3: You will notice from the previous command, that all the pods are running except one: ‘kube-dns’. For resolving this we will install a pod network. To install the CALICO 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
```

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

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/master/src/deploy/recommended/kubernetes-
dashboard.yaml
```

It will look something like this:

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

Then you will get something like this:

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/

You will then be prompted with this page, to enter the credentials:

Step 7: In this step, we will create the service account for the dashboard and get it’s credentials.
Note: Run all these commands in a new terminal, or your kubectl proxy command will stop.

Run the following commands:

1. This command will create a 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

```
$ kubectl get secret $(kubectl get serviceaccount dashboard -o jsonpath="{.secrets[0].name}") -o jsonpath="
{.data.token}" | base64 --decode
```

You should get the token like this:

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

Become a Certified Professional→

5. You have successfully logged into your dashboard!

Steps For Only Kubernetes Node VM (knode)

It is time to get your node, to join 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
```

Bingo! Your Kubernetes Cluster is ready if you get something similar to the above screenshot.

So that brings an end to this blog on how to install kubernetes on Ubuntu 16.04. Do look out for other blogs in this series which will explain the various other aspects of Kubernetes.