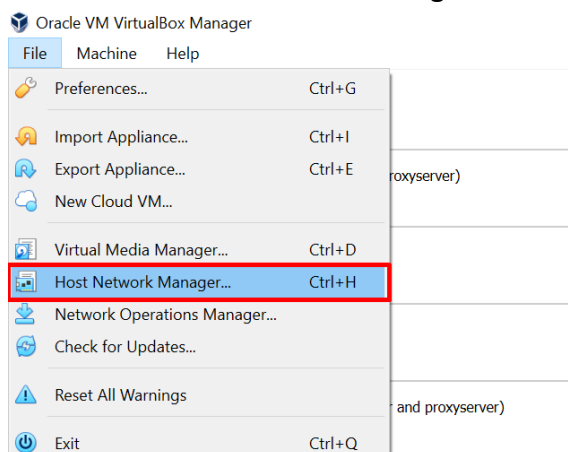


Swarm cluster Setup on Windows using VirtualBox

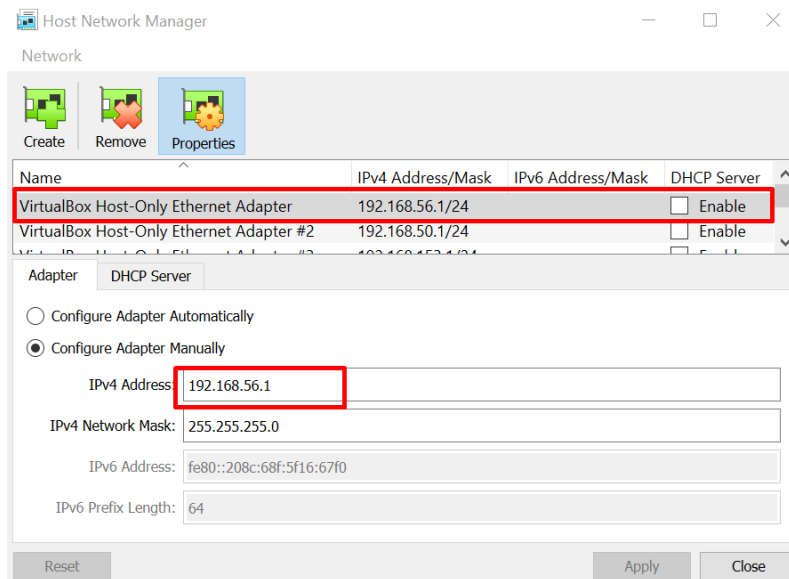
NOTE: This document is initial prepared for setting up Kubernetes cluster in windows using VirtualBox. So, the naming conventions used in this document will be slightly different. Replace **k8s** in the names of the VMs with **docker**. So **k8s-master** to be replaced with **docker-master** or **swarm-master** and so on...This is just to avoid confusion with the naming later on.

Prerequisites

- Hyper-V disabled on Windows Pro machines
- Admin rights
- VirtualBox installed
<https://www.virtualbox.org/wiki/Downloads>
- Host network adapter in VirtualBox
 - Go to file -> Host Network Manager

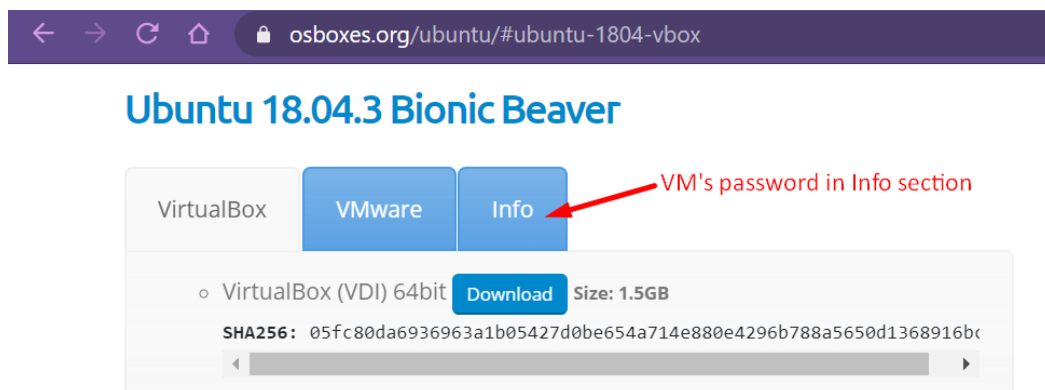


- By default, at least one host-only adapter should be available. If not create one using the create button
- **Turn off DHCP server** and note down gateway IPv4 address that is assigned to the adapter. We are going to use this subnet IPs to assign static IPs to VMs. In my case, the gateway IP is 192.168.56.1
- Click close once done



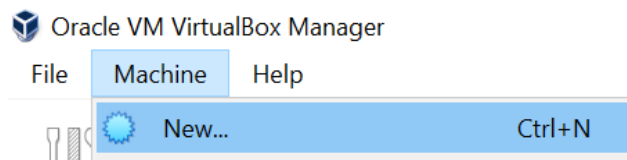
Preparing Master node VM

1. Download Ubuntu 18.04.3 Bionic Beaver(preferred) VirtualBox image
<https://www.osboxes.org/ubuntu/#ubuntu-1804-vbox>
 Password for this VM will be in the info section as shown below. Password is usually **osboxes.org**



2. Once downloaded, extract the image
3. Create a new VM in VirtualBox

Go to Machine -> New



Give a name to your VM and select options as in screenshot and click Next
 I have chosen name as k8s-master as this node will be acting as **Kubernetes master**


← Create Virtual Machine

Name and operating system

Please choose a descriptive name and destination folder for the new virtual machine and select the type of operating system you intend to install on it. The name you choose will be used throughout VirtualBox to identify this machine.

Name:

Machine Folder:

Type: 

Version:

Expert Mode

Next

Cancel

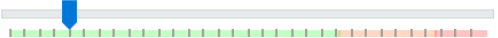
4. Select RAM size as 2048

← Create Virtual Machine

Memory size

Select the amount of memory (RAM) in megabytes to be allocated to the virtual machine.

The recommended memory size is **1024 MB**.

 2048 MB

4 MB 16384 MB

5. Choose the image that is downloaded and extracted from osboxes.org website and click create

← Create Virtual Machine

Hard disk

If you wish you can add a virtual hard disk to the new machine. You can either create a new hard disk file or select one from the list or from another location using the folder icon.

If you need a more complex storage set-up you can skip this step and make the changes to the machine settings once the machine is created.

The recommended size of the hard disk is **10.00 GB**.

☐ Do not add a virtual hard disk

☐ Create a virtual hard disk now

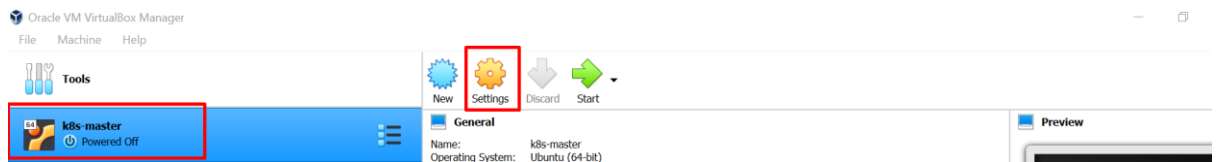
☒ Use an existing virtual hard disk file

Ubuntu 18.04.3 (64bit).vdi (Normal, 500.00 GB)

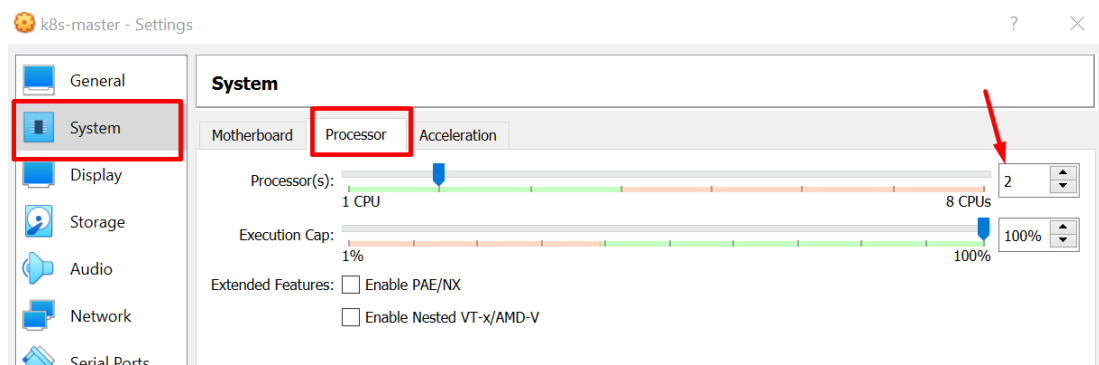
Create

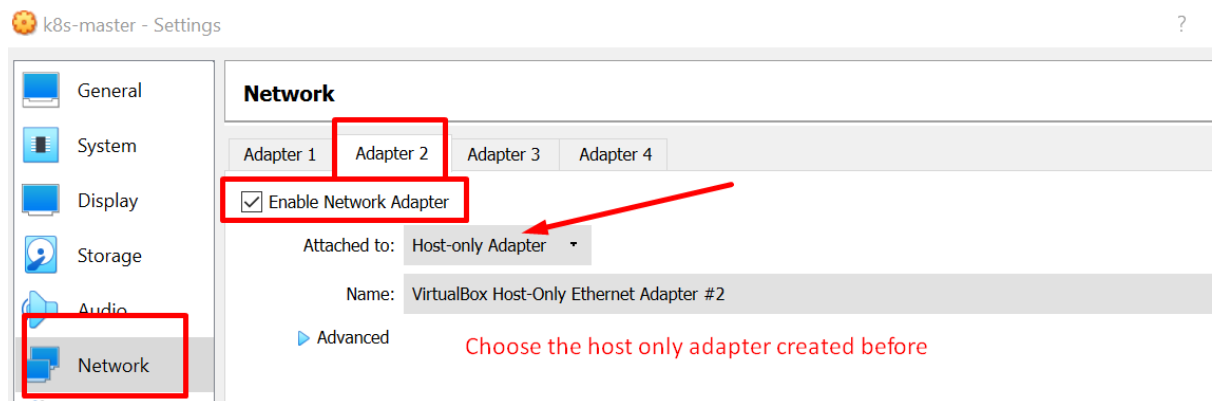
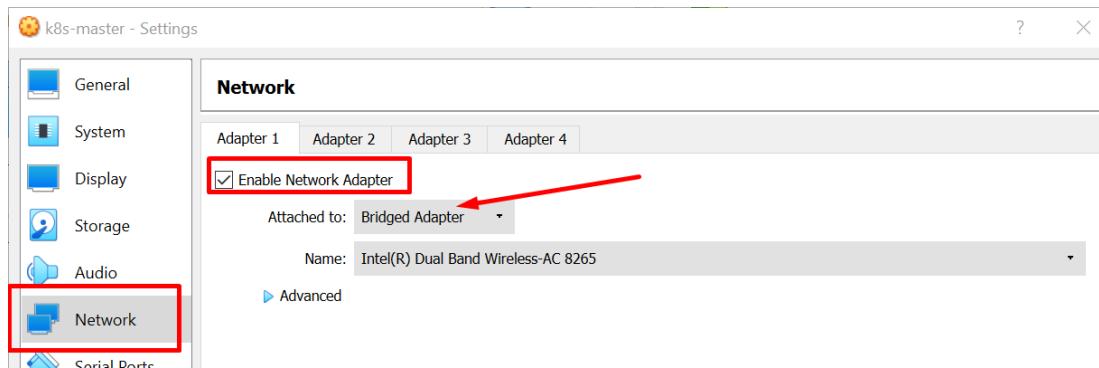
Cancel

- Once VM is created, we need to do some additional changes in the settings before running it. Go to settings after selecting the image

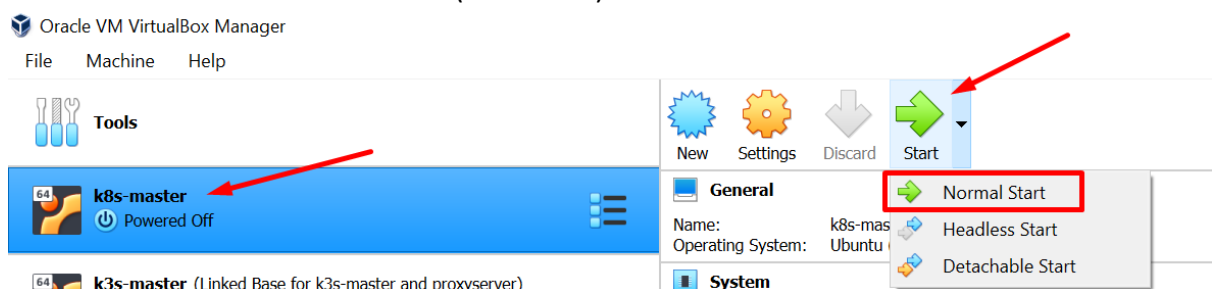


- Make changes as shown in the screenshots below and click close/ok.
(We will have 2 network adapters connected to the VM and hence the following changes)



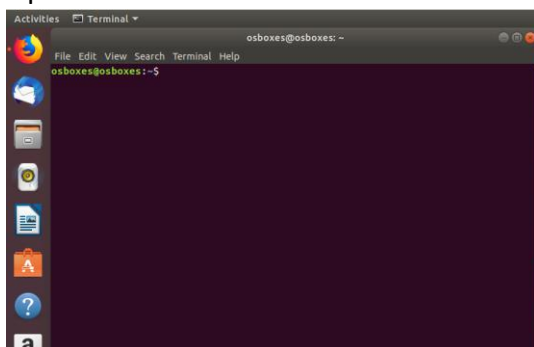


8. Now start the VM in normal mode(GUI mode)



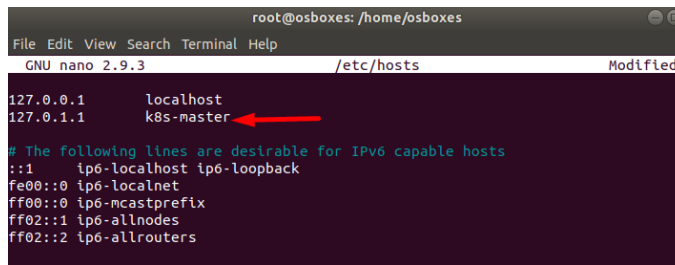
9. Once GUI is available, login using password(osboxes.org in most cases)

10. Open terminal in the Ubuntu VM and become root user



11. Change hostname and hosts file

- **nano /etc/hostname** -> change hostname to **k8s-master** (because this VM will be used as master, choose your own name of choice)
- Update hosts file with new hostname: **nano /etc/hosts**



```

root@osboxes: /home/osboxes
File Edit View Search Terminal Help
GNU nano 2.9.3 /etc/hosts Modified
127.0.0.1    localhost
127.0.1.1    k8s-master
# 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

```

12. Update and add related packages

- **apt update -y**
- **apt install net-tools**
- **apt install openssh-server -y**

13. Adding static IP to interface

This is required as VMs change their IP after every reboot(most of the time). We cannot use these dynamic IPs to start the cluster as worker node fails to find master every time master node changes its IP. In order to avoid starting the cluster every time with a new IP it's better to have a fixed IP set to an Interface and have the cluster running forever. In this case we will use **enp0s8** as the interface with fixed IP.

Also, we will choose static IPs from host-adapter subnet range created before.

- **ifconfig enp0s8 192.168.56.2**
- **nano /etc/network/interfaces** and append these lines


```

auto enp0s8
iface enp0s8 inet static
    address 192.168.56.2
    netmask 255.255.255.0
      
```

Note the IP 192.168.56.2 is used for master. Going forward we will use 192.168.56.3 for worker node 01(k8s-slave01) and 192.168.56.4 for worker node 02(k8s-slave02) and so on...

14. Installing Docker. Run these commands individually as root user

- **sudo apt update**
- **sudo apt install apt-transport-https ca-certificates curl software-properties-common**
- **curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -**
- **sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu bionic stable"**
- **sudo apt update**
- **apt-cache policy docker-ce**
- **sudo apt install docker-ce**
- **sudo systemctl status docker**

15. Check if docker is installed properly by running **docker version**

```

root@k8s-master:/home/osboxes# docker version
Client: Docker Engine - Community
Version: 19.03.11
API version: 1.40
Go version: go1.13.10
Git commit: 42e35e61f3
Built: Mon Jun 1 09:12:22 2020
OS/Arch: linux/amd64
Experimental: false

Server: Docker Engine - Community
Engine:
Version: 19.03.11
API version: 1.40 (minimum version 1.12)
Go version: go1.13.10
Git commit: 42e35e61f3
Built: Mon Jun 1 09:10:54 2020
OS/Arch: linux/amd64
Experimental: false
containerd:
Version: 1.2.13
GitCommit: 7ad184331fa3e55e52b890ea95e65ba581ae3429
runc:
Version: 1.0.0-rc10
GitCommit: dc9208a3303feef5b3839f4323d9beb36df0a9dd
docker-init:
Version: 0.18.0
GitCommit: fec3683
root@k8s-master:/home/osboxes#

```

16. Install docker-compose

- Docker-compose will not come preinstalled with docker engine on Linux distributions. We need to install manually.
- First check the latest version of docker compose at <https://github.com/docker/compose/releases>

1.26.0

docker-dsg-cibot released this 3 days ago · 31 commits to master since this release

Features

- Added `docker context` support.
- Added missing test dependency `set` to `setup.py`.
- Added `--attach-dependencies` to command up for attaching to dependencies.
- Allowed compatibility option with `COMPOSE_COMPATIBILITY` environment variable.
- Bumped `Pytest` to 5.3.4 and add refactor compatibility with new version.
- Bumped `OpenSSL` from 1.1.1f to 1.1.1g.
- Bumped `certifi` from 2019.11.28 to 2020.4.5.1.
- Bumped `docker-py` from 4.2.0 to 4.2.1.

At the time of writing this document it is **1.26.0**

- Run the following commands
`sudo curl -L https://github.com/docker/compose/releases/download/1.26.0/docker-compose-`uname -s`-`uname -m` -o /usr/bin/docker-compose`
 Note the docker-compose version in the URL
- `sudo chmod +x /usr/bin/docker-compose`

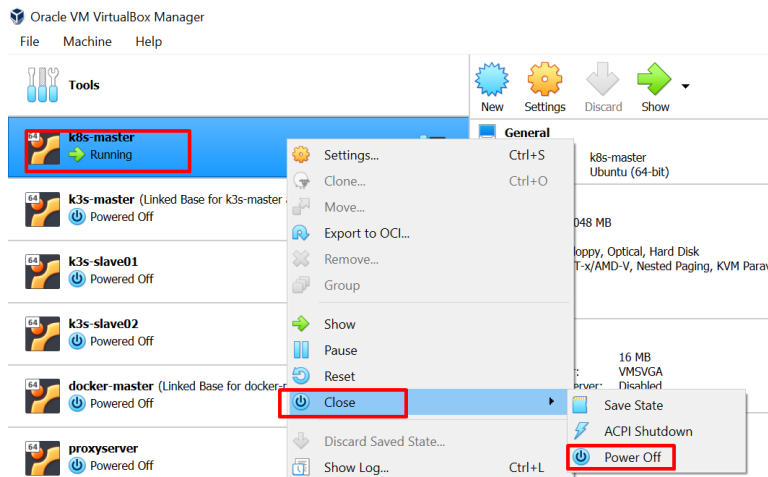
17. Check docker-compose installed version using **docker-compose version**

```

root@k8s-master:/home/osboxes# docker-compose version
docker-compose version 1.26.0, build d4451659
docker-py version: 4.2.1
CPython version: 3.7.7
OpenSSL version: OpenSSL 1.1.0l 10 Sep 2019
root@k8s-master:/home/osboxes#

```

18. Shut down the VM by doing **sudo init 0** in the terminal or from VirtualBox itself by

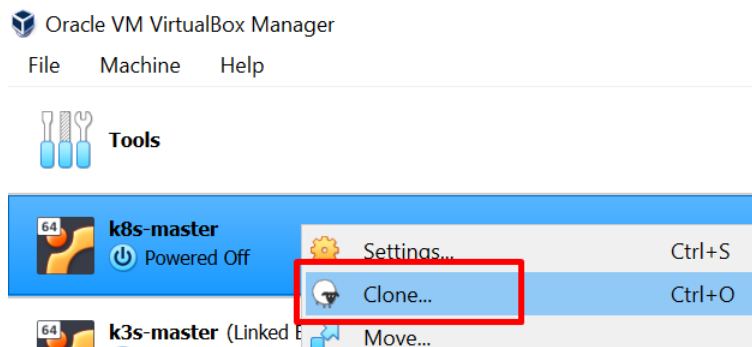


Preparing swarm node-01 VM

We will clone the master node for slave nodes as this will save significant amount of disk space on the host.

1. Cloning the master node

Make sure master node is powered off. Right click on master VM and select clone



2. Rename the VM to **k8s-slave01**, select the appropriate options as shown in screenshots and click Next

← Clone Virtual Machine

New machine name and path

Please choose a name and optionally a folder for the new virtual machine. The new machine will be a clone of the machine **k8s-master**.

Name:

Path:

MAC Address Policy:

Additional Options:

Generate

Expert Mode

Next

Cancel

← Clone Virtual Machine

New machine name and path

Please choose a name and optionally a folder for the new virtual machine. The new machine will be a clone of the machine **k8s-master**.

Name:

Path:

MAC Address Policy:

Additional Options: ☐ Keep Disk Names
☐ Keep Hardware UUIDs

Expert Mode

Next

Cancel

3. Select Linked clone and click clone

← Clone Virtual Machine

Clone type

Please choose the type of clone you wish to create.

If you choose **Full clone**, an exact copy (including all virtual hard disk files) of the original virtual machine will be created.

If you choose **Linked clone**, a new machine will be created, but the virtual hard disk files will be tied to the virtual hard disk files of original machine and you will not be able to move the new virtual machine to a different computer without moving the original as well.

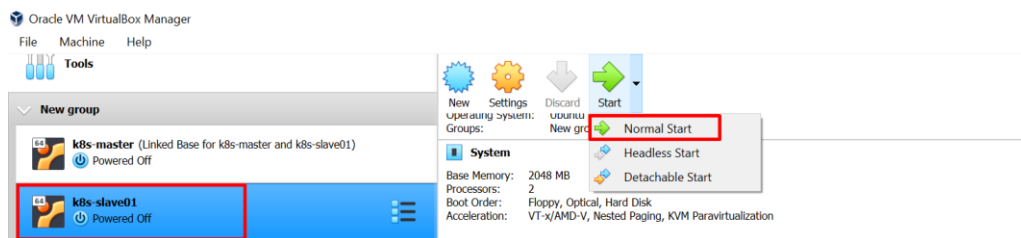
If you create a **Linked clone** then a new snapshot will be created in the original virtual machine as part of the cloning process.

- ☐ Full clone
☒ Linked clone

Clone

Cancel

4. Select the slave VM and start it in GUI mode



- Repeat Steps 11 & 13 used in setting up **master VM**. But this time use K8s-slave01 instead of k8s-master
 192.168.56.3 instead of 192.168.56.2
 Basically, we must change the hostname and hosts file with name of the slave chosen and giving static IP to VM which is different from static IP assigned to master
- Shutdown this VM

Preparing swarm node-02 VM

Same steps as in **Preparing Worker node 01 VM** but use hostname as **k8s-slave02** and static IP as **192.168.56.4**

What is done so far...

- We have configured 3 VMs with names k8s-master, k8s-slave01 and k8s-slave02 (It is expected that user has changed these names to something like docker-master, docker-slave01 & docker-slave02)
- We have given them static IPs 192.168.56.2, 192.168.56.3 and 192.168.56.4 (according to user's host adapter subnet)
- Since docker and docker-compose is installed on master and slaves are nothing but clones of master, these should be available in all VMs

Starting the swarm cluster

1. Start all 3 VMs and login as root in the terminal
2. **ON THE MASTER VM**, run

docker swarm init --advertise-addr 192.168.56.2

This will generate a token that should be used in other nodes to join them as swarm nodes in the cluster

```
root@k8s-master:/home/osboxes# docker swarm init --advertise-addr 192.168.50.2
Swarm initialized: current node (2la9pxp10lyuebf57uts1b059) is now a manager.

To add a worker to this swarm, run the following command:

docker swarm join --token SWMTKN-1-0rl0nhj7ge740ub28m24lbyh3d6ffswcwx63peeonchq0df-d8p7jwbu473omgu0syjbkyu3q 192.168.50.2:2377

To add a manager to this swarm, run 'docker swarm join-token manager' and follow the instructions.

root@k8s-master:/home/osboxes#
```

- If you need other nodes to be used as swarm slaves/workers, paste the token marked in **red** in all slave nodes inside the terminal. This will join them as workers in the cluster
- If you need other nodes to be used as master(as in multi master swarm cluster), use the command marked in the **green** to generate a separate token that should be used in other nodes to join them as master in the cluster
- Also ensure that you use only static IP of master that you have set and not the one used in this document.

Note:

Please note that I have used Static IP(192.168.56.2) of the master node that is assigned to enp0s8 interface in the initial setup. This is to ensure that master uses a fixed IP on the cluster and never depends on IP of enp0s3 which is dynamic. In this case, swarm slaves will always find the master in the cluster as it is running on static IP.

Use the static IP that you have assigned while setting up the master, 192.168.56.2 is in my case only.

If you are not sure of what Static IP you have set to master, simply run **ifconfig enp0s8**. In my case I have set the static IP of master as 192.168.50.2 as the host adapter subnet starts from 192.168.50.....

```
root@k8s-master:/home/osboxes# ifconfig enp0s8
enp0s8: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.50.2 netmask 255.255.255.0 broadcast 192.168.50.255
    inet6 fe80::a00:27ff:fe57:dad6 prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:57:da:d6 txqueuelen 1000 (Ethernet)
    RX packets 51 bytes 4993 (4.9 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 73 bytes 7202 (7.2 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

root@k8s-master:/home/osboxes#
```

3. Run **docker node ls** on **MASTER** to see the list of nodes in the cluster and their status
4. Check References section for more details

References

- https://knowledgepill.it/posts/docker_swarm_compendium/?fbclid=IwAR2JWNRHr_mfKlauScuEmcW1hwl7lxLJdf6_KUWN8GQiSqxc5pXmkw9RDEaM
- <https://medium.com/edureka/docker-swarm-cluster-of-docker-engines-for-high-availability-40d9662a8df1>

Credits

- Authors from references section