

# MS4S21 - Big Data Engineering and Applications

Moizzah Asif, J418, moizzah.asif@southwales.ac.uk

University of South Wales



# Contents

<b>1 Apache Hadoop multi-node cluster on Ubuntu VMs</b>	<b>3</b>
1.1 VM creation . . . . .	3
1.2 Ubuntu 20.04 installation . . . . .	5
1.3 Setting up Ubuntu for Hadoop-3.2.2 hdfs cluster . . . . .	7
1.4 Java for Hadoop-3.2.2 . . . . .	10
1.5 Download and configure hadoop . . . . .	11
1.5.1 Cloning VM . . . . .	13
1.5.2 Virtual Box network configuration . . . . .	14
1.5.3 Revisiting system files . . . . .	19
1.5.4 Establish Password less SSH access between hadoop nodes . . . . .	21
1.5.5 Configuring the cluster . . . . .	24
1.5.6 Worker/s node only configuration . . . . .	28
1.6 Starting the hadoop cluster . . . . .	29
<b>2 Mapreduce on Hadoop Cluster</b>	<b>32</b>
2.1 Configuration on hadoop cluster . . . . .	32
2.1.1 Worker/data nodes configuration . . . . .	32
2.1.2 Master/name node configuration . . . . .	32
2.2 Executing a demo mapreduce job on the cluster . . . . .	33
2.3 A word count mapreduce task with python . . . . .	36
<b>3 Twitter App creation for the module</b>	<b>38</b>

# Apache Hadoop multi-node cluster on Ubuntu VMs

This chapter will walk you through step-wise general guidelines of creating a Apache hadoop cluster 3.2.1 at present using Ubuntu latest stable release via VMs.

## 1.1 VM creation

Please note these steps can be followed to create any OS's VM in general, however the Linux Ubuntu 19.10 is used as example here.

1. Open virtual box and click on the new icon.
2. You are about to create a Ubuntu VM, set the values of each field as shown in Fig 1.1, and then press continue. Please note that chose the machine folder based on where you want to save it on your computer. If you are using university computer, then please choose your network drive.

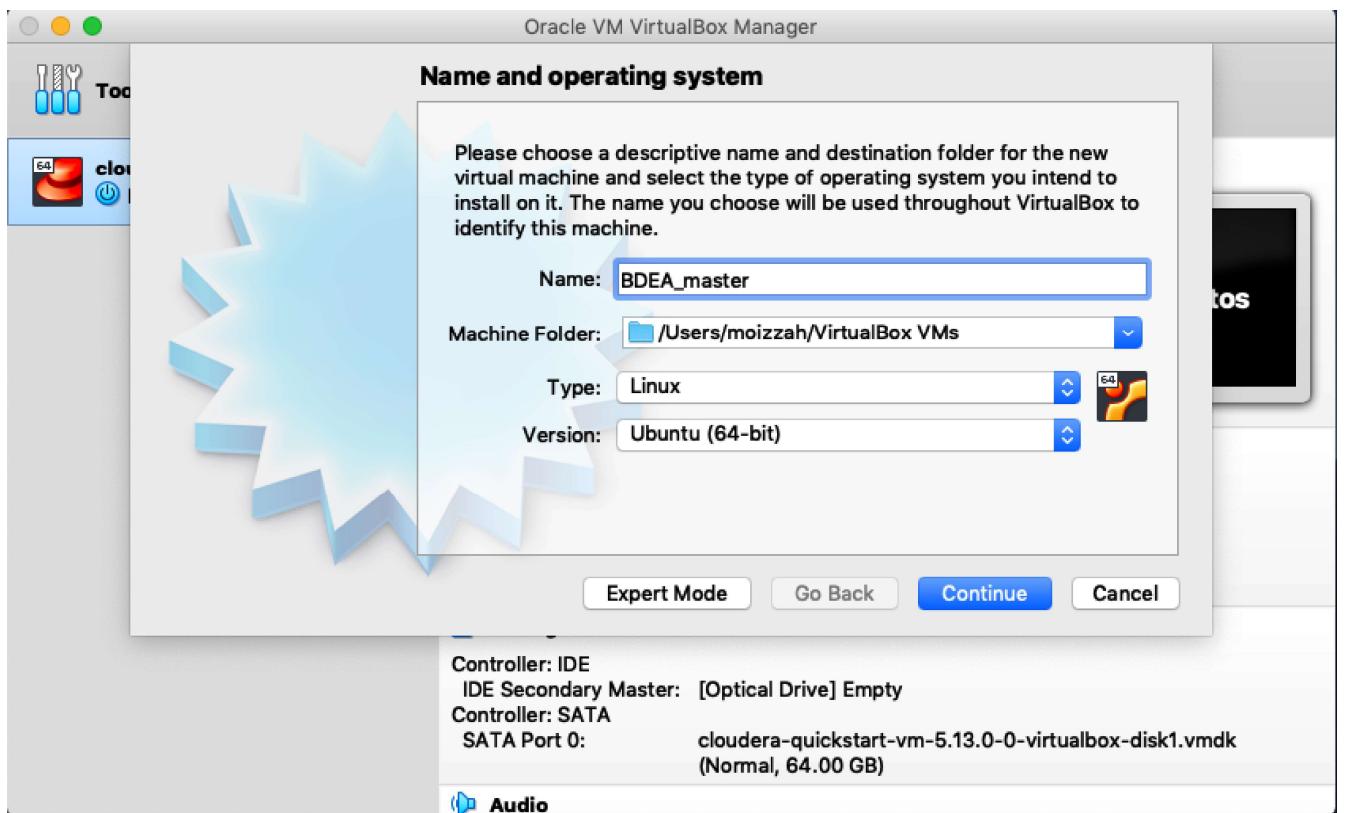


Figure 1.1: Ubuntu VM creation on virtual box - 1

3. Set RAM/memory to 12 GB/ 12288 MBs, and press continue.
4. Select the radio button which enables you to create a virtual hard disk, and then press create.
5. It will be followed by a pop up window where you can specify the hard disk file type as shown in Fig 1.2, please select VHD (virtual Hard Disk) to stay consistent with lectures, and then press continue.

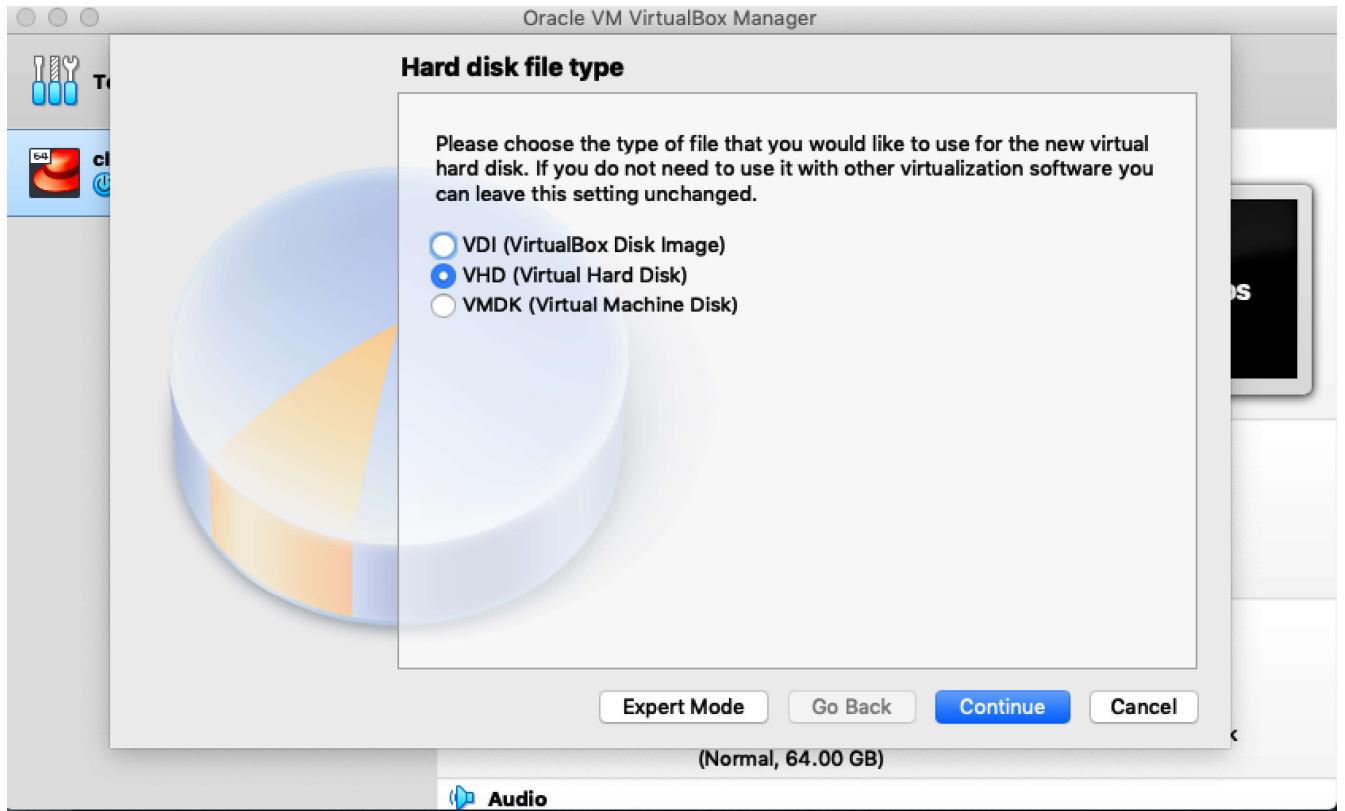


Figure 1.2: Ubuntu VM creation on virtual box - Hard disk file type

6. Select Fixed size radio button for physical storage on hard disk, press continue and then either keep or change the default value of 10 GB. This decision should be made based on your local machine's available hard drive storage, as well as the number of VMs you will have to run at once. In this case consider at least two VMs at a time to simulate the cluster. Press continue and wait for the VM to be created. Please note that you have just configured the physical and memory storage requirement until now. This is similar to bringing home a computer which doesn't have any operating system installed on it.
7. You should be able to see the VM on Virtual Box left pane now. As shown in Fig 1.3 the name of this empty VM should be what you named it in the first step of creation.

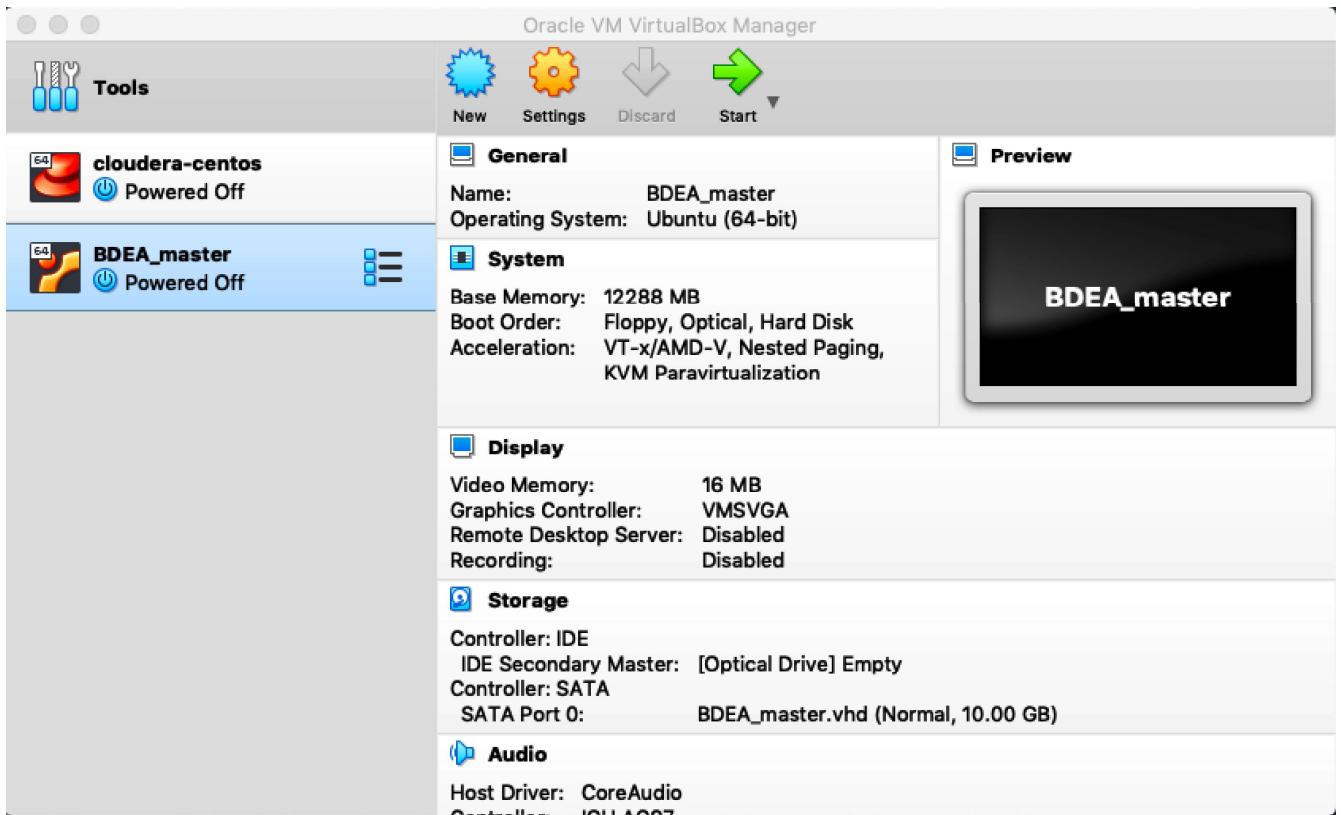


Figure 1.3: Ubuntu VM creation on virtual box - Empty VM

## 1.2 Ubuntu 20.04 installation

You will have to use the .iso image provided in BB Week 1 folder or the one you have downloaded on your computer or saved on your uni network drive.

1. Start the VM that you have just created in the previous section; click on the folder icon with a green arrow; Click on the add icon in the new pop up window; select the .iso image from the location where you have saved/downloaded it; The file should appear on the pop up windows as shown in Fig 1.4; choose this file and process with installation in the next window that pops up on your screen. The essential options in the process are listed below, please make sure that you have selected them.
  - (a) install ubuntu
  - (b) English UK
  - (c) normal installation
  - (d) erase disk an install ubuntu(it's empty already)

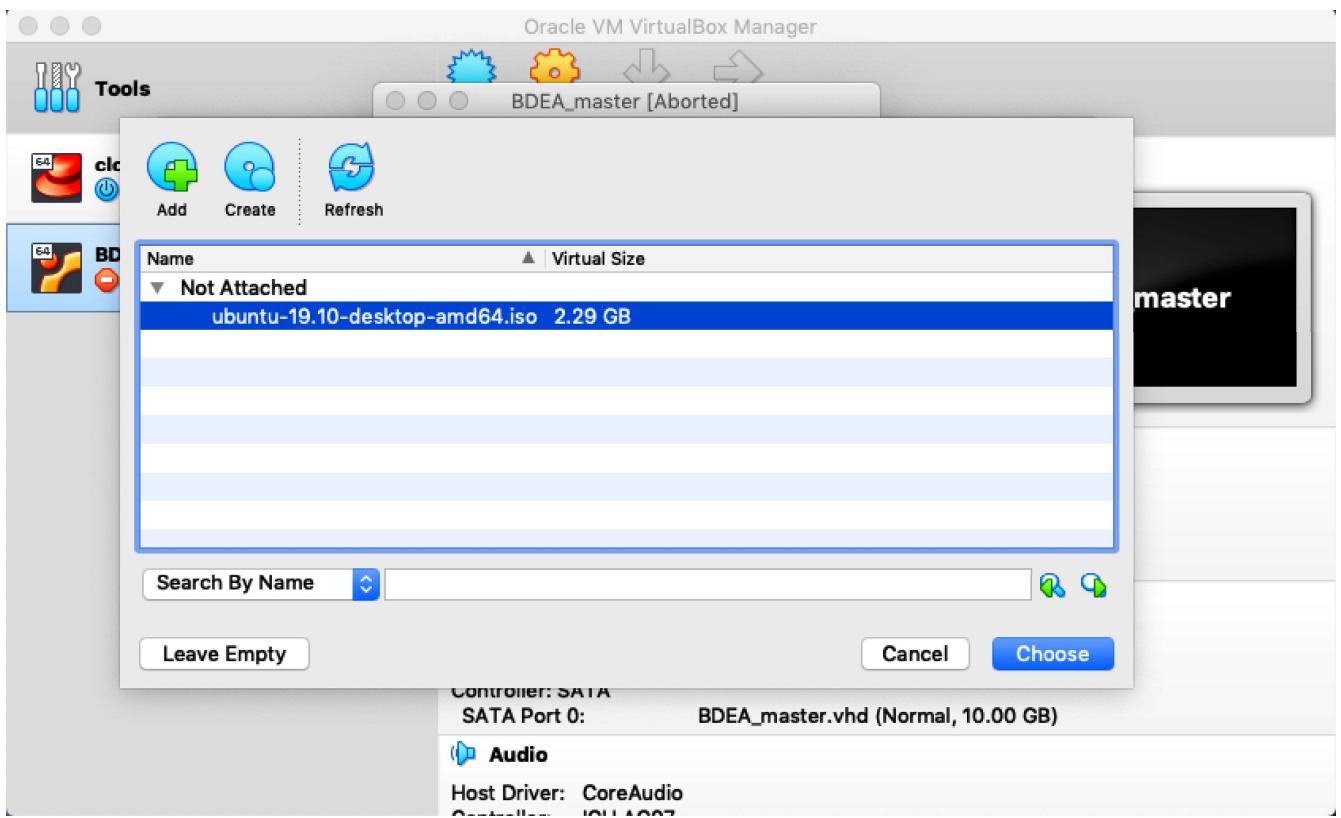


Figure 1.4: Ubuntu VM creation on virtual box - adding iso image 1

2. The screen display shown in Fig 1.5 helps you create a user profile on ubuntu. Please name the user profile intuitively. A user profile BDEA (Big Data Engineering and Applications) will be created for this tutorial's purpose.

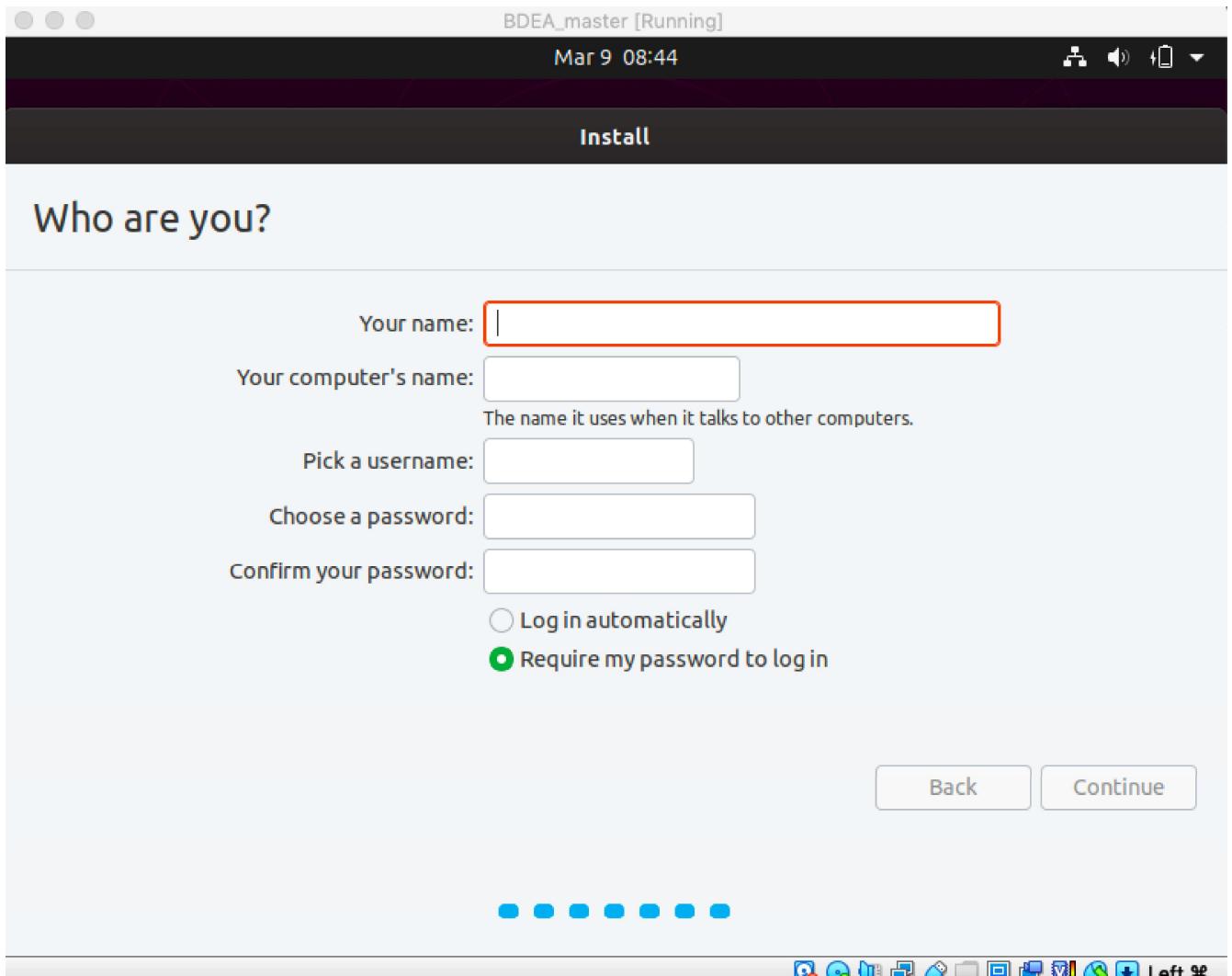


Figure 1.5: Ubuntu user creation

### 1.3 Setting up Ubuntu for Hadoop-3.2.2 hdfs cluster

Once you arrive at your Ubuntu's desktop, open Terminal so that you may set up the machine for hadoop-3.2.2 installation. Follow the guidelines listed below after opening terminal.

*For more information on linux terminal please use the forum in blackboard module.*

1. upgrade and update apt with sudo privileges (APT - advanced packaging tool) - `sudo apt update/upgrade`
2. Install openssh server - `sudo apt install openssh-server`; verify the status now: `sudo systemctl status ssh`. You should be able to get an output similar to Fig 1.6
3. check hostname - `sudo hostname`; rename to hadoop-master - `sudo hostname hadoop-master`; verify successful renaming as shown in Fig 1.6.

```

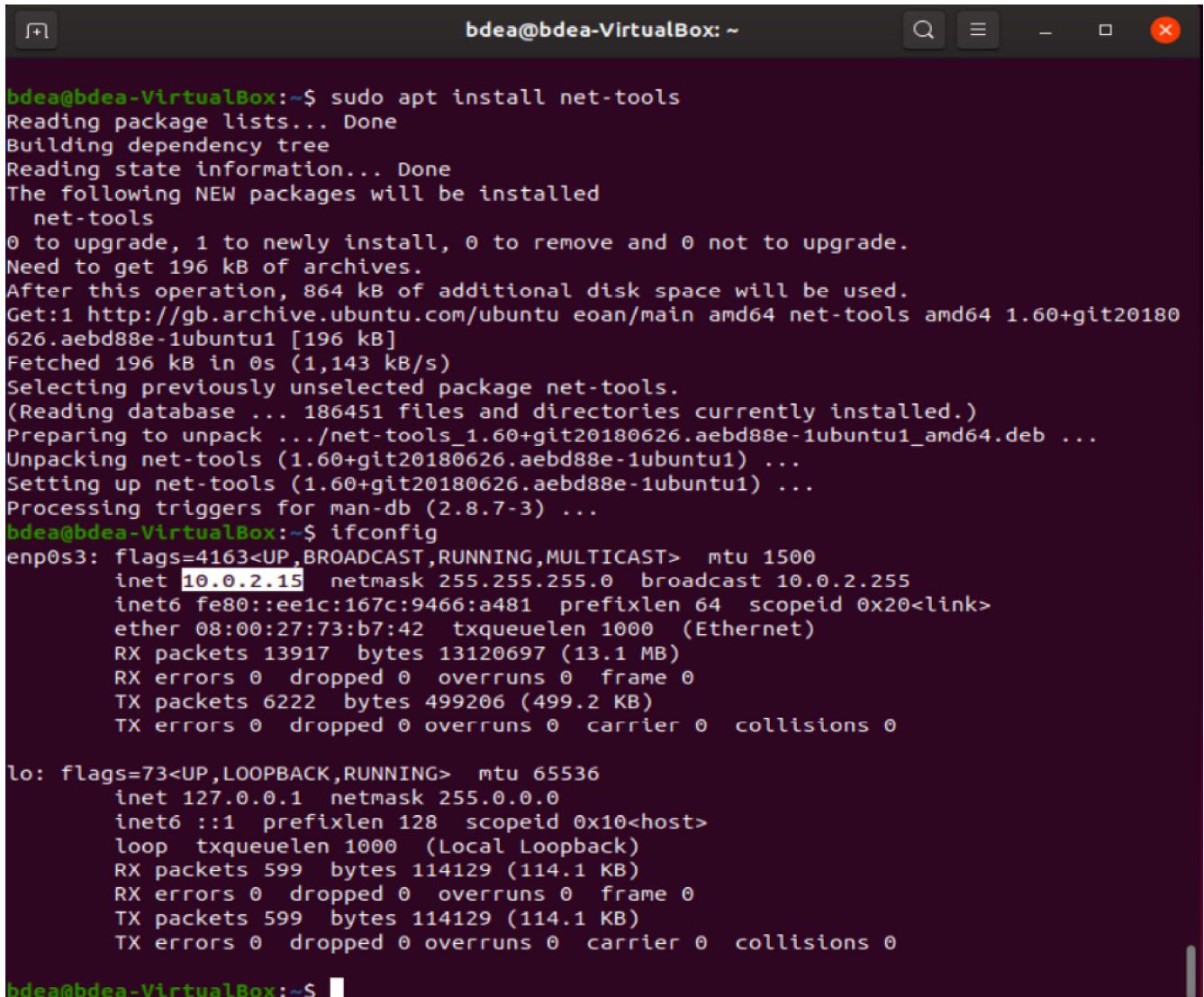
bdea@bdea-VirtualBox:~$ sudo systemctl status ssh
● ssh.service - OpenBSD Secure Shell server
  Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: enabled)
  Active: active (running) since Mon 2020-03-09 09:05:48 GMT; 1min 21s ago
    Docs: man:sshd(8)
          man:sshd_config(5)
 Main PID: 27955 (sshd)
   Tasks: 1 (limit: 4915)
  Memory: 1.1M
   CGroup: /system.slice/ssh.service
           └─27955 /usr/sbin/sshd -D

Mar 09 09:05:48 bdea-VirtualBox systemd[1]: Starting OpenBSD Secure Shell server...
Mar 09 09:05:48 bdea-VirtualBox sshd[27955]: Server listening on 0.0.0.0 port 22.
Mar 09 09:05:48 bdea-VirtualBox sshd[27955]: Server listening on :: port 22.
Mar 09 09:05:48 bdea-VirtualBox systemd[1]: Started OpenBSD Secure Shell server.
bdea@bdea-VirtualBox:~$ sudo hostname
bdea-VirtualBox
bdea@bdea-VirtualBox:~$ sudo hostname hadoop-master
bdea@bdea-VirtualBox:~$ sudo hostname
hadoop-master
bdea@bdea-VirtualBox:~$ █

```

Figure 1.6: openssh-server and hostname verification

4. install gedit text editor for smooth editing - *sudo apt install gedit*. If you are working on remote hosts, this editor may not be available and you may have to work with vim. (A healthy discussion on vim had been conducted during lectures. for more info, please create a thread in the forum, or use one if it is already there.)
5. update the hosts file by adding hadoop-master as a host.
  - (a) Install ifconfig to find the ip address of your machine via terminal - *sudo apt install net-tools*. Fig 1.7 shows the output and script of installing net-tools and listing the internet configurations of your machine.
  - (b) Find the ip address of your vm - *ifconfig*. Fig 1.7 shows the ip address highlighted manually.



```

bdea@bdea-VirtualBox:~$ sudo apt install net-tools
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed
  net-tools
0 to upgrade, 1 to newly install, 0 to remove and 0 not to upgrade.
Need to get 196 kB of archives.
After this operation, 864 kB of additional disk space will be used.
Get:1 http://gb.archive.ubuntu.com/ubuntu eoan/main amd64 net-tools amd64 1.60+git20180626.aebd88e-1ubuntu1 [196 kB]
Fetched 196 kB in 0s (1,143 kB/s)
Selecting previously unselected package net-tools.
(Reading database ... 186451 files and directories currently installed.)
Preparing to unpack .../net-tools_1.60+git20180626.aebd88e-1ubuntu1_amd64.deb ...
Unpacking net-tools (1.60+git20180626.aebd88e-1ubuntu1) ...
Setting up net-tools (1.60+git20180626.aebd88e-1ubuntu1) ...
Processing triggers for man-db (2.8.7-3) ...
bdea@bdea-VirtualBox:~$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
      inet 10.0.2.15  netmask 255.255.255.0  broadcast 10.0.2.255
        inet6 fe80::ee1c:167c:9466:a481  prefixlen 64  scopeid 0x20<link>
          ether 08:00:27:73:b7:42  txqueuelen 1000  (Ethernet)
            RX packets 13917  bytes 13120697 (13.1 MB)
            RX errors 0  dropped 0  overruns 0  frame 0
            TX packets 6222  bytes 499206 (499.2 KB)
            TX errors 0  dropped 0  overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
      inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
          loop  txqueuelen 1000  (Local Loopback)
            RX packets 599  bytes 114129 (114.1 KB)
            RX errors 0  dropped 0  overruns 0  frame 0
            TX packets 599  bytes 114129 (114.1 KB)
            TX errors 0  dropped 0  overruns 0  carrier 0  collisions 0
bdea@bdea-VirtualBox:~$ 

```

Figure 1.7: net-tools installation and ifconfig output

- (c) open the hosts file for editing, to add hadoop-master as a host and map it to its ip address  
*- sudo gedit hosts.* The hosts file configuration shown in Fig 1.8 shows the correct format of adding a new hosts; in this example the new host is hadoop-master. Please note the highlighted line refers to a host which does not exist any more, please remove this line to avoid any carried forward errors. You may as well comment this line to keep as a reminder for future references.



```

Open  +  *hosts
Save  -  ×
127.0.0.1      localhost
127.0.1.1      bdea-VirtualBox
10.0.2.15      hadoop-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

```

Figure 1.8: interim configuration of hosts file

6. Download, install and configure the latest stable Hadoop-3.2.2 compatible version of Java. Please follow the instructions in next section.

## 1.4 Java for Hadoop-3.2.2

Hadoop-3.2.2 is not compatible with Java 11 for development purposes. A hdfs cluster could be successfully simulated with Java 11, however, some of the big data services on hadoop cluster can not work with Java 11, such as: mapreduce framework on hadoop.

Therefore, we will work with Java 8, as it the latest stable and compatible version of Java with hadoop-3.2.2. You can follow the next set of instructions for Java 11 as well.

1. Install Java 8 JDK - *sudo apt install openjdk-8-jdk*

The Java Development Kit (JDK) is a software development environment used for developing Java applications and applets. It includes the Java Runtime Environment (JRE), an interpreter/loader (Java), a compiler (javac), an archiver (jar), a documentation generator (Javadoc) and other tools needed in Java development

2. Configure bash file to set user variable JAVA\_HOME. Setting this variable will allow the OS to locate Java directory whenever Java is called by any process/application. Either enter the absolute to path to bash file (it is located in your home directory) or open it from the home directory as follows - *sudo gedit .bashrc*

The value of JAVA\_HOME should be set as - */usr/lib/jvm/java-8-openjdk-amd64/jre*. Please refer to Fig for exact syntax of variable initialisation in bash file.

```

.bashrc
/home/bdea
alias egrep='egrep --color=auto'
fi

# colored GCC warnings and errors
#export GCC_COLORS='error=01;31:warning=01;35:note=01;36:caret=01;32:locus=01:quote=01'

# some more ls aliases
alias ll='ls -alF'
alias la='ls -A'
alias l='ls -CF'

# Add an "alert" alias for long running commands.  Use like so:
# sleep 10; alert
alias alert='notify-send --urgency=low -i "$( [ $? = 0 ] && echo terminal || echo error)" "$
(history|tail -n1|sed -e '\''s/^\\s*[0-9]\\+\\s*/;s/[;&]\\s*alert$//'\'')"

# Alias definitions.
# You may want to put all your additions into a separate file like
# ~/.bash_aliases, instead of adding them here directly.
# See /usr/share/doc/bash-doc/examples in the bash-doc package.

if [ -f ~/.bash_aliases ]; then
  . ~/.bash_aliases
fi

# enable programmable completion features (you don't need to enable
# this, if it's already enabled in /etc/bash.bashrc and /etc/profile
# sources /etc/bash.bashrc).
if ! shopt -oq posix; then
  if [ -f /usr/share/bash-completion/bash_completion ]; then
    . /usr/share/bash-completion/bash_completion
  elif [ -f /etc/bash_completion ]; then
    . /etc/bash_completion
  fi
fi

export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64/jre

```

sh ▾ Tab Width: 8 ▾ Ln 1, Col 1 ▾ INS

Figure 1.9: Initialising JAVA\_HOME in bash file

## 1.5 Download and configure hadoop

Unlike Java, hadoop is not installed. Hadoop's directories are downloaded and configuration files are modified to ensure smooth execution and run of cluster.

1. Download hadoop-3.2.2 from apache hadoop website -  
`wgethttps://archive.apache.org/dist/hadoop/common/hadoop-3.2.2/hadoop-3.2.2.tar.gz`
2. unzip and delete the tar file was follows:
  - `tar -xvf hadoop-3.2.2.tar.gz`
  - `rm hadoop-3.2.2.tar.gz`
3. move hadoop to the usr local directory while renaming it to hadoop - `sudo mv hadoop-3.2.2 /usr/local/hadoop`
4. Add hadoop bin and sbin directories to the PATH variable using bash file. Please follow the syntax in Fig 1.10 to concatentae the paths to PATH variable.

```

*.bashrc
/home/bdea
Save
Open ▾
+ ×

alias grep='grep --color=auto'
alias fgrep='fgrep --color=auto'
alias egrep='egrep --color=auto'
fi

# colored GCC warnings and errors
#export
GCC_COLORS='error=01;31:warning=01;35:note=01;36:caret=01;32:locus=01:quote=01;33:help=01;36'

# some more ls aliases
alias ll='ls -alF'
alias la='ls -A'
alias l='ls -CF'

# Add an "alert" alias for long running commands.  Use like so:
#   sleep 10; alert
alias alert='notify-send --urgency=low -i "$( [ $? = 0 ] && echo terminal
|| echo error)" "$(history|tail -n1|sed -e '\''s/^[\s*][0-9]+\+\s*//;s/[;&]
\s*alert$/'\''")"

# Alias definitions.
# You may want to put all your additions into a separate file like
# ~/.bash_aliases, instead of adding them here directly.
# See /usr/share/doc/bash-doc/examples in the bash-doc package.

if [ -f ~/.bash_aliases ]; then
    . ~/.bash_aliases
fi

# enable programmable completion features (you don't need to enable
# this, if it's already enabled in /etc/bash.bashrc and /etc/profile
# sources /etc/bash.bashrc).
if ! shopt -oq posix; then
    if [ -f /usr/share/bash-completion/bash_completion ]; then
        . /usr/share/bash-completion/bash_completion
    elif [ -f /etc/bash_completion ]; then
        . /etc/bash_completion
    fi
fi

export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64/jre
export PATH=$PATH:/usr/local/hadoop/bin:/usr/local/hadoop/sbin
export CONF=/usr/local/hadoop/etc/hadoop

```

Figure 1.10: Bash file with Hadoop bin and sbin added to path

5. create another environment variable using the bash file, so that accessing the hadoop config directory becomes easier. Note the creation of CONF variable in Fig 1.10

Don't forget to source the bash file whenever you update it.

Before we start editing the hadoop configuration files, it's essential that we clone the master VM to create a worker VM. Cloning will help save time and effort at this stage.

### 1.5.1 Cloning VM

Please follow the steps below to clone the master VM.

1. Power off the VM that you would like to clone.
2. Select the VM and right click on it in the Virtualbox interface.
3. Select the clone option as shown in Fig 1.11

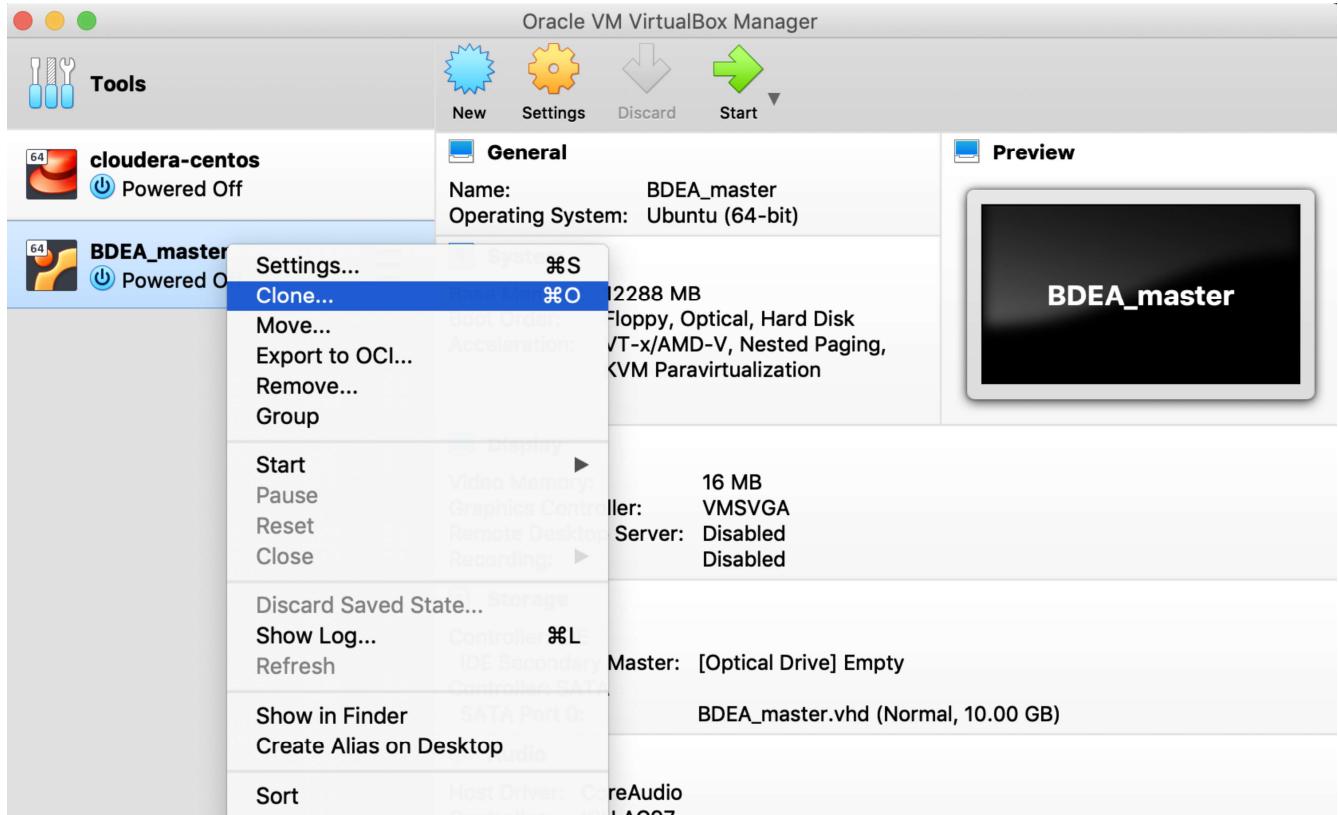


Figure 1.11: Cloning VM - 1

4. In the next pop up screen you have to very carefully choose MAC Address Policy; if you are not careful enough you may end up messing up with network adapters of both, the clone and master. Please select the option as shown in Fig 1.12; it enables you to generate new network adapters.

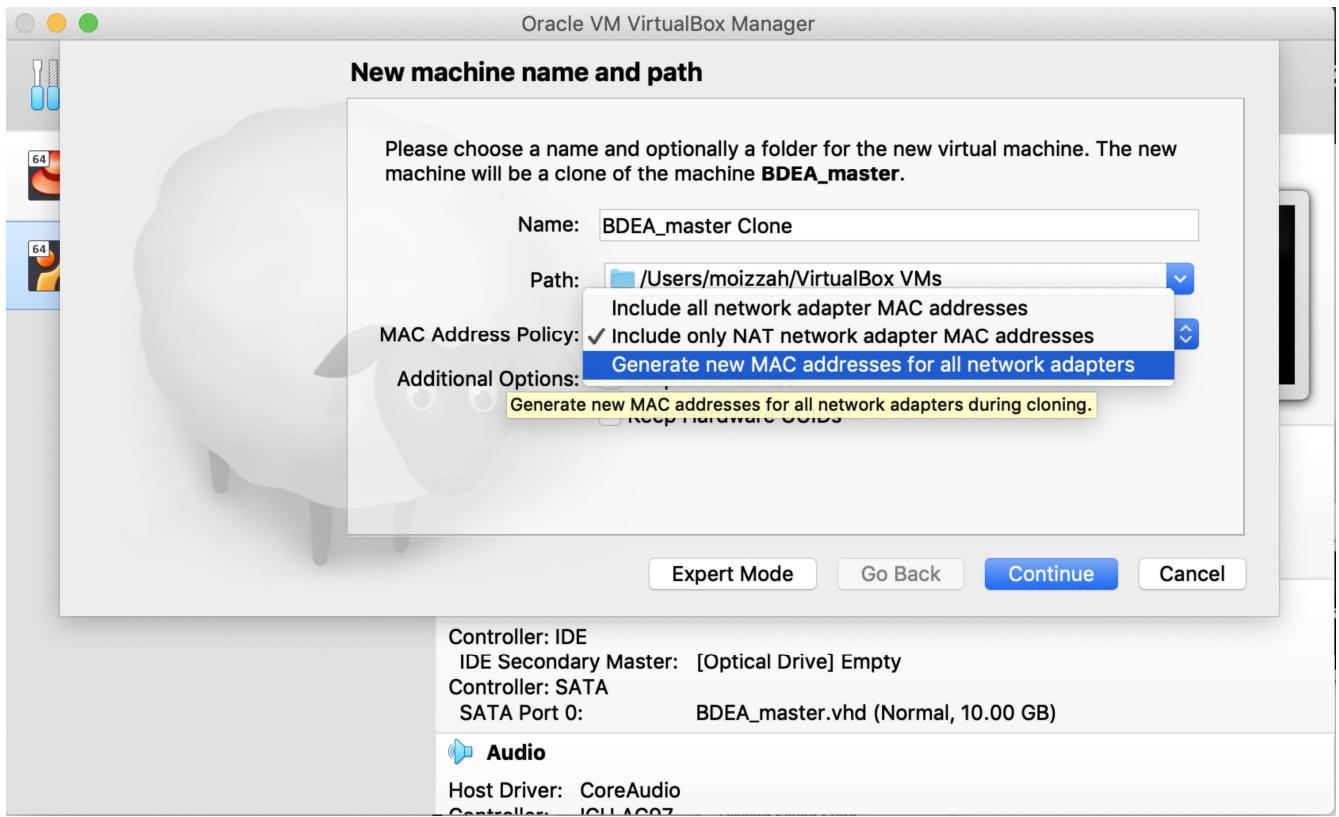


Figure 1.12: Cloning VM - Network adapter configuration

5. Create a full clone.
6. Once the cloning has successfully finished, rename the clone in a way that it indicates that this is going to be a worker node.

### 1.5.2 Virtual Box network configuration

As discussed in the lectures, there are 5 different network mode settings available on virtual box. Table 1.1 shows the list and connectivity features of each mode; you can further explore the characteristics of each networking mode following the hyperlinks in the table.

Mode	VM→Host	VM←Host	VM1↔VM2	VM←Net/LAN	VM→Net/LAN
Host-only	+	+	+	-	-
Internal	-	-	+	-	-
Bridged	+	+	+	+	+
NAT	+	Port forward	+	-	Port forward
NATservice	+	Port forward	+	-	Port forward

Table 1.1: Virtual box network modes

The two suitable networking modes for communication between the hadoop nodes as well as localhost are *host-only* and *bridged* networking modes. The following subsections provide instruction on setting up these two networking modes.

#### Host-only Adapter

Here are a few selected lines from virtualbox user manual on host-only network mode.

*“When host-only networking is used, Oracle VM VirtualBox creates a new software interface on the host which then appears next to your existing network interfaces. In other words, whereas with bridged networking an existing physical interface is used to attach virtual machines to, with host-only networking a new loopback interface is created on the host. Host-only networking is particularly useful for preconfigured virtual appliances, where multiple virtual machines are shipped together and designed to cooperate. For example, one virtual machine may contain a web server and a second one a database, and since they are intended to talk to each other, the appliance can instruct Oracle VM VirtualBox to set up a host-only network for the two. A second, bridged, network would then connect the web server to the outside world to serve data to, but the outside world cannot connect to the database.”*

Before a VM can be configured to work with host-only networking mode, creating a host only interface is essential. Please take a moment here to remind yourself that you did not have to create any interfaces, instead the networking mode was simply switched to host-only network mode in the lectures. The virtual box version compatible with the windows OS installed on university lab computers automatically generates this interfaces and reduces the number of configuration steps you may have to take otherwise.

This is quite possible that the windows-OS/macOS version on your system may not support such automatic interface creation. Therefore this section covers instruction for configuring host-only adapter on macOS platomr assuming that automatic interface creation for host-only network mode is disabled, and one has to manually create it. The section also re-demonstrates switching to host-only adapter network mode on windows OS as demonstrated in lectures. The purpose of this replication is to highlight the differences between setting host-only adapter network on virtual box for a VM when an adapter is and isn't preconfigured.

## Windows

Switching to host-only adapter network mode with a pre-configured adapter.

1. Click on the VM, the network settings need to be changed for.
2. Select network.
3. Change the selection for Attached to by selecting Host-only adapter from the drop down menu.
4. Select one the pre-configured network adapter from Name's drop down menu (You may have a choice of selecting from more than one adapters. If selecting one of them doesn't generate separate ip addresses, then select another adapter until you get unique ip addresses for of each of your VM).
5. Select Advanced.
6. Ensure that promiscous mode is set to allow all.
7. The pop up menu and configuration window should look similar to the one shown in Fig 1.13.
8. Click ok, and repeat the exercise from bullet point 1 to bullet point 7 on the rest of the VMs.

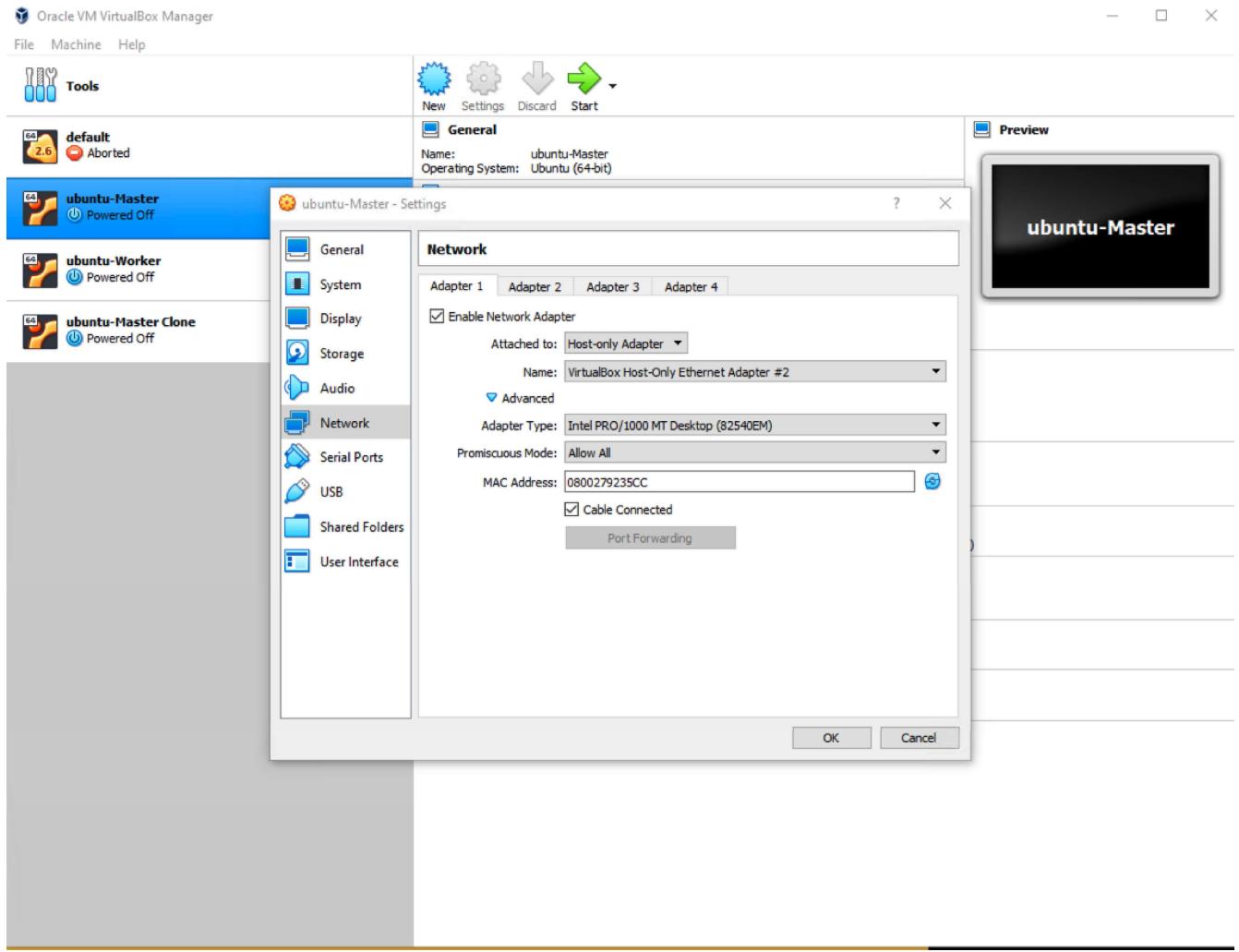


Figure 1.13: Switching to host-only adapter network mode on virtual box windows release

## MacOS

Switching to host-only adapter network mode when adapter isn't preconfigured.

1. Open the menu named virtual box for virtual box.
2. Open Host network manager.
3. Click on create.
4. A new adapter named vboxnet should be created now; make sure that DHCP server is enabled. Your screen should look similar to the one shown in Fig 1.14
5. Apply the settings and close the host network manager window.

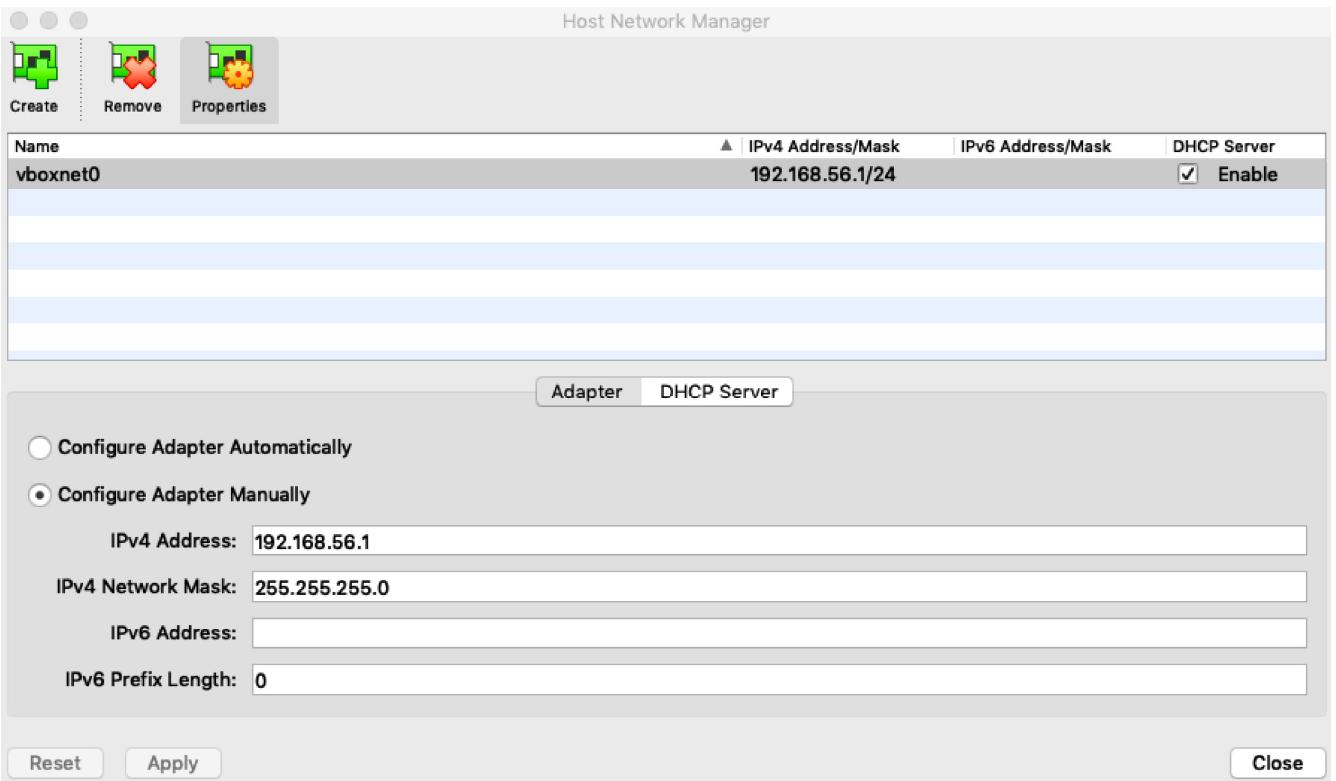


Figure 1.14: Adapter creation for host-only adapter network mode on virtual box macOS release

6. Follow the steps listed below on your master and worker VMs respectively, to switch to host-only network mode:
  - (a) click on the VM, select settings;
  - (b) select network mode;
  - (c) change the Attached to: selection from the drop down menu to Host-only Adapter;
  - (d) click on advanced;
  - (e) change promiscous mode to Allow all by using the drop down menu;
  - (f) the pop window should look similar to the one shown in Fig ;
  - (g) click ok, and repeat the same exercise from bullet point 6a to 6f on the other VM/s.

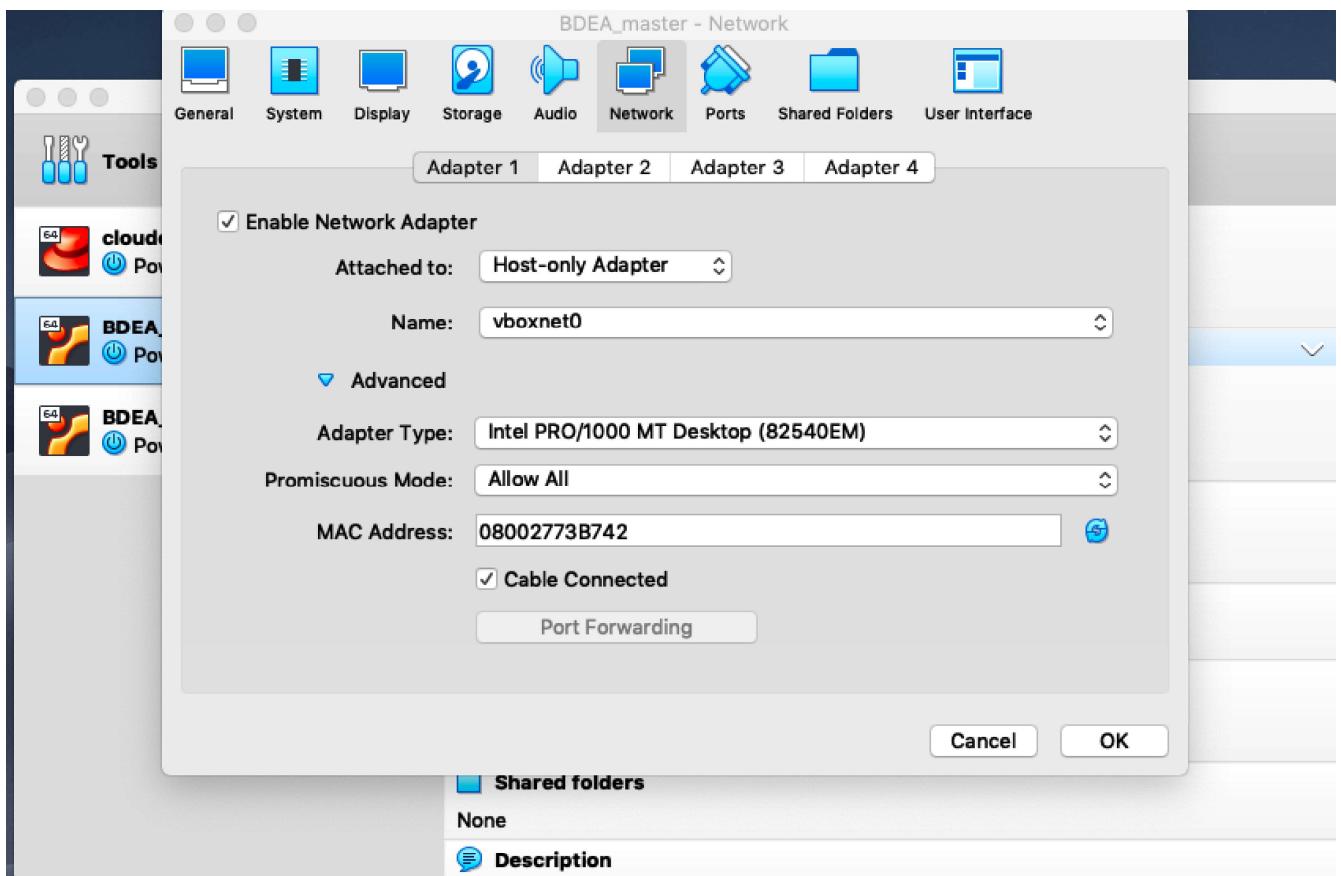


Figure 1.15: Selecting host-only adapter network mode for VMs on virtual box macOS release

## Bridged Adapter

Below are a few lines selected from virtualbox user manual on bridged adapter network mode.

*With bridged networking, Oracle VM VirtualBox uses a device driver on your host system that filters data from your physical network adapter. This driver is therefore called a net filter driver. This enables Oracle VM VirtualBox to intercept data from the physical network and inject data into it, effectively creating a new network interface in software. When a guest is using such a new software interface, it looks to the host system as though the guest were physically connected to the interface using a network cable. The host can send data to the guest through that interface and receive data from it. This means that you can set up routing or bridging between the guest and the rest of your network.*

To setup this network mode on a windows host machine, may need admin rights to configure certain settings. Therefore, we resorted to using the host-only adapter network mode during lectures. The only drawback to this mode is that the VM can not be connected to internet via the adapter the network setting is configured on. Whereas bridged adapter mode enables all possible internet and intra-machine communications.

The virtualbox release for MacOS has been used to demonstrate setting this network mode. The experience is usually smooth on MacOS for most of the network mode virtualbox has to offer as compared to windows operating system.

1. Click on the VM you want to change the network mode for.
2. Click on settings.

3. Click on Network tab.
4. Change the Attached to selection to Bridged Adapter mode from drop down menu.
5. Select either ethernet, wireless adapter, usb-c or anyother port from where internet access is possible on your machine from the drop down menu of Name.
6. Click on the Advanced option.
7. Ensure that promiscous mode is set to allow all.
8. The pop network configuration window should look similar to the one shown in Fig 1.16; click on ok to save the settings and exit.

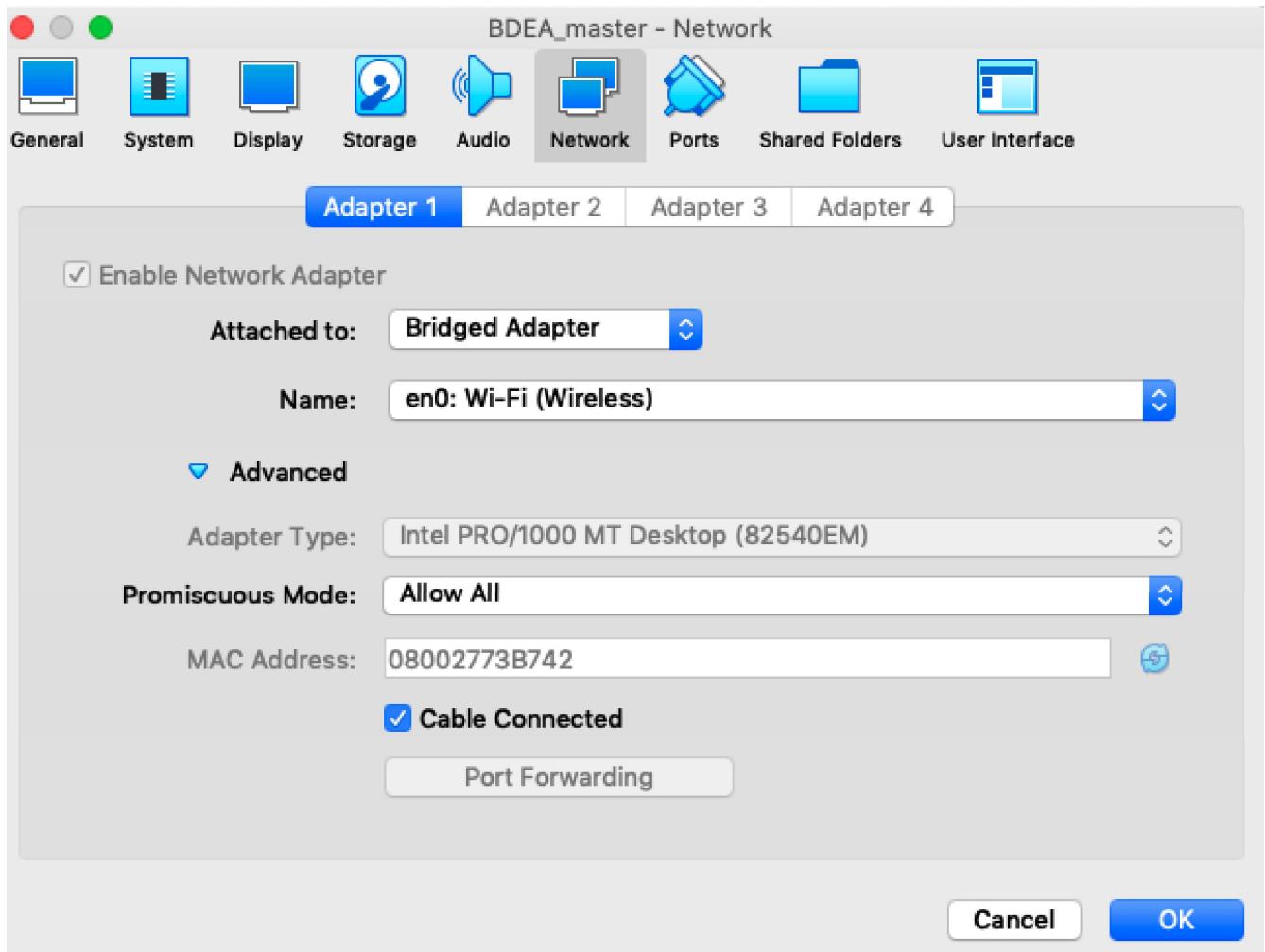


Figure 1.16: Selecting Bridged adapter network mode for VMs on virtual box macOS release

### 1.5.3 Revisiting system files

Since you have another node now and have changed network mode, the /etc/hosts file need to be altered accordingly on both name and data nodes respectively.

1. Copy the new ip address assigned to each VM being used as a node in hadoop cluster. Use *ifconfig* command on the terminals as shown in Fig 1.17

```
bdea@BDEA-master:~$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.0.33 netmask 255.255.255.0 broadcast 192.168.0.255
        inet6 fe80::ee1c:167c:9466:a481 prefixlen 64 scopeid 0x20<link>
            ether 08:00:27:73:b7:42 txqueuelen 1000 (Ethernet)
            RX packets 91213 bytes 134785094 (134.7 MB)
            RX errors 0 dropped 2 overruns 0 frame 0
            TX packets 48743 bytes 3528421 (3.5 MB)
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
        loop txqueuelen 1000 (Local Loopback)
        RX packets 832 bytes 87959 (87.9 KB)
        RX errors 0 dropped 0 overruns 0 frame 0
        TX packets 832 bytes 87959 (87.9 KB)
        TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

bdea@BDEA-master:~$
```

Figure 1.17: Selecting Bridged adapter network mode for VMs on virtual box macOS release

2. Update the /etc/hosts file of each VM by adding ip address and hostname mapping of all nodes - use *sudo gedit /etc/hosts* to configure the hosts files in all nodes. The hosts files on all Hadoop nodes should look similar to the one shown in Fig 1.18

```

*hosts
/etc

127.0.0.1      localhost
#127.0.1.1    bdea-VirtualBox

192.168.0.33   BDEA-master
192.168.0.42   BDEA-worker

# 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

```

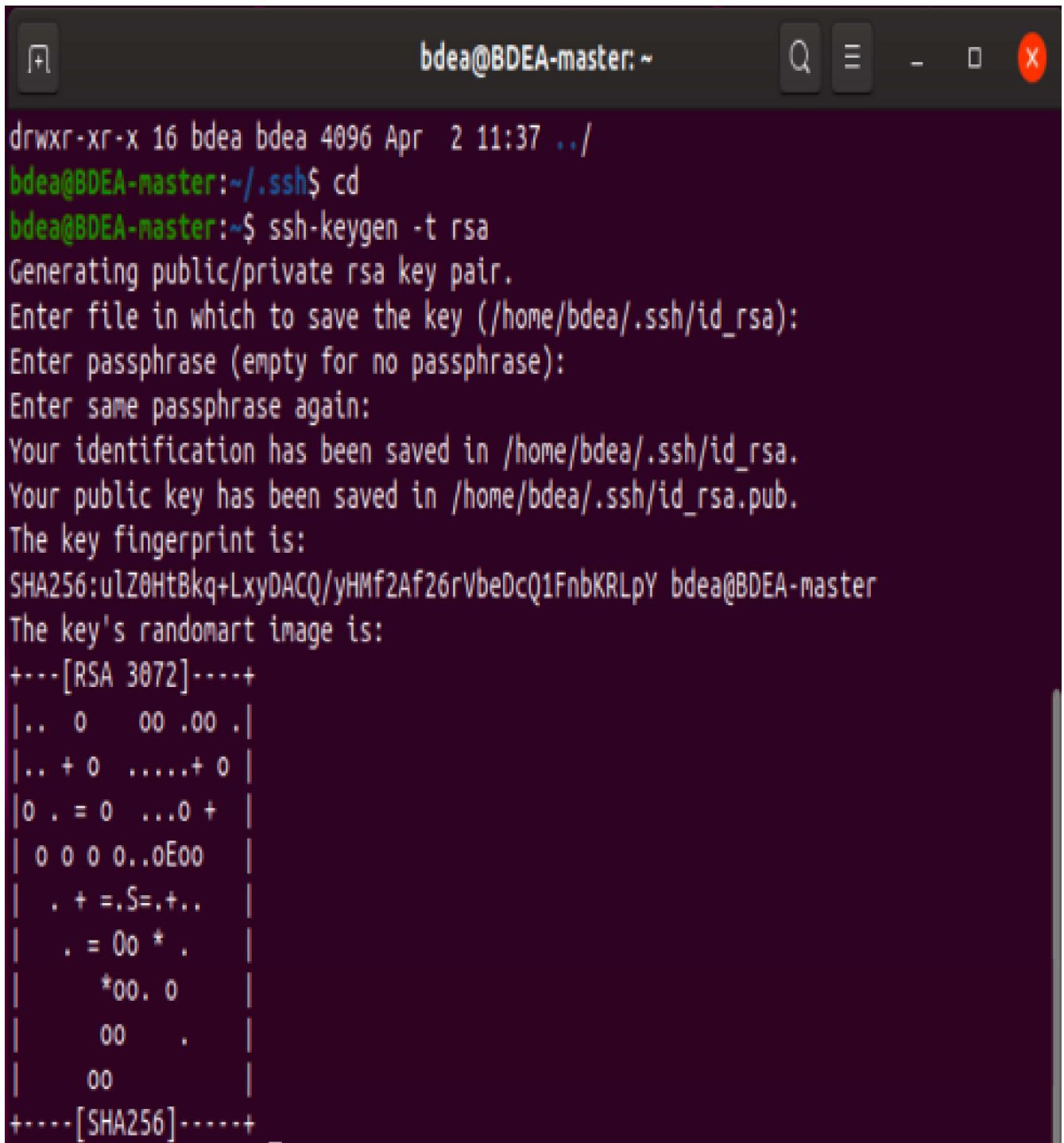
Figure 1.18: final version of /etc/hosts file for Hadoop cluster nodes

#### 1.5.4 Establish Password less SSH access between hadoop nodes

We have discussed about SSH protocol in lectures; you may as well refresh your knowledge and find answers to some questions from this [web-link](#). Similarly, we have already had a discussion around the need for establishing password less SSH access between hadoop nodes in the lectures.

Follow the steps below on name node to establish the passwordless ssh connection between all hadoop nodes. Please note that these steps can be replicated in any other context where either SSH key generation, password less access or both are required.

1. Generate a public and private SSH key pair: `ssh-keygen -t rsa`.
2. When asked for file name, press enter to apply default.
3. When asked for passphrase, press enter to use none.
4. A successful key generation terminal output should look similar to the one shown in Fig 1.19.



The screenshot shows a terminal window with the following session:

```
bdea@BDEA-master: ~
drwxr-xr-x 16 bdea bdea 4096 Apr  2 11:37 ../
bdea@BDEA-master:~/ssh$ cd
bdea@BDEA-master:~/ssh$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/bdea/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/bdea/.ssh/id_rsa.
Your public key has been saved in /home/bdea/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:ulZ0HtBkq+LxyDACQ/yHMf2Af26rVbeDcQ1FnKRLpY bdea@BDEA-master
The key's randomart image is:
+---[RSA 3072]----+
|.. o  oo .oo .|
|.. + o ....+ o |
|o . = o ...o + |
| 0 0 0 0..oEoo |
| . + =.S=,+.. |
| . = 00 * , |
| *00. o |
| 00 . |
| 00 |
+---[SHA256]----+
```

Figure 1.19: SSH key generation

5. Now copy master node's public key to the authorised users on all hadoop worker nodes. Use the following syntax to copy to all worker/data nodes respectively: `ssh-copy-id <username>@<worker-hostname>`.
6. When asked to continue connection, type yes and then press enter to continue.
7. Enter the password when prompted to enter the user's password on the worker node to which you are copying the keys.
8. The terminal output for a successful `ssh-copy-id` to a worker node should look similar to the one shown in Fig 1.20.

```
bdea@BDEA-master:~$ ssh-copy-id bdea@BDEA-worker
The authenticity of host 'bdea-worker (192.168.0.42)' can't be established.
ECDSA key fingerprint is SHA256:Ejd+Ci2dcnLDWTF+R3kFM+QshBYZTFnh68G/R0h6kfg.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter
out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are promp
ted now it is to install the new keys
bdea@bdea-worker's password:
Permission denied, please try again.
bdea@bdea-worker's password:
Permission denied, please try again.
bdea@bdea-worker's password:

Number of key(s) added: 1

Now try logging into the machine, with: "ssh 'bdea@BDEA-worker'"
and check to make sure that only the key(s) you wanted were added.
```

Figure 1.20: copying SSH key pairs to remote servers

9. you can ensure the successful completion of this process by logging into one of the worker nodes using SSH. The terminal syntax should be similar to the following: `ssh <username>@<worker-hostname>`.
10. After establishing the connection, you can disconnect and get back to the host's terminal by pressing 'control' and 'd' keys together. An example of logging in via SSH and logging out with this key combination is shown in Fig 1.21. Please note the change of username and host name in green colour on the terminal before and after ssh connection is established and disconnected.

```

bdea@BDEA-master:~$ ssh bdea@BDEA-worker
Welcome to Ubuntu 19.10 (GNU/Linux 5.3.0-45-generic x86_64)

 * Documentation: https://help.ubuntu.com
 * Management: https://landscape.canonical.com
 * Support: https://ubuntu.com/advantage

56 updates can be installed immediately.
1 of these updates is a security update.
To see these additional updates run: apt list --upgradable

bdea@BDEA-worker:~$ logout
Connection to bdea-worker closed.
bdea@BDEA-master:~$ 

```

Figure 1.21: Establishing and disconnecting SSH connection to remote servers

### 1.5.5 Configuring the cluster

#### Configuring hadoop files on all hadoop nodes

Quoting Apache Hadoop's documentation:

“adoop’s Java configuration is driven by two types of important configuration files:

1. Read-only default configuration -

- core-default.xml,
- hdfs-default.xml,
- yarn-default.xml, and
- mapred-default.xml.

2. Site-specific configuration -

- etc/hadoop/core-site.xml,
- etc/hadoop/hdfs-site.xml,
- etc/hadoop/yarn-site.xml, and
- etc/hadoop/mapred-site.xml.

Additionally, you can control the Hadoop scripts found in the bin/ directory of the distribution, by setting site-specific values via the etc/hadoop/hadoop-env.sh and etc/hadoop/yarn-env.sh.

To configure the Hadoop cluster you will need to configure the environment in which the Hadoop daemons execute as well as the configuration parameters for the Hadoop daemons.

HDFS daemons are NameNode, SecondaryNameNode, and DataNode. YARN daemons are ResourceManager, NodeManager, and WebAppProxy. If MapReduce is to be used, then the MapReduce Job History Server will also be running. For large installations, these are generally running on separate hosts.”

Site specific configuration may vary for some file in worker and master nodes. So, the recommended approach is to configure the files with common configurations for master and worker, first in the master node and then copy them over to the worker node.

### **Hadoop files on master/name and worker/data nodes**

Common configuration of site specific configuration files of a hadoop cluster will be discussed in this section. You may have noticed that all the site specific configuration files are located in hadoop’s etc/hadoop folder. So, based on our lab demonstrations the exact path to these files is /usr/local/hadoop/etc/hadoop/.

If you scroll back up to Fig 1.10 you will notice that we have already created a path variable to this folder, called CONF. CONF will help you to navigate and access these files with ease and quickly.

We will be editing haddop-env.sh, core-site.xml, hdfs-site.xml and workers file in this section.

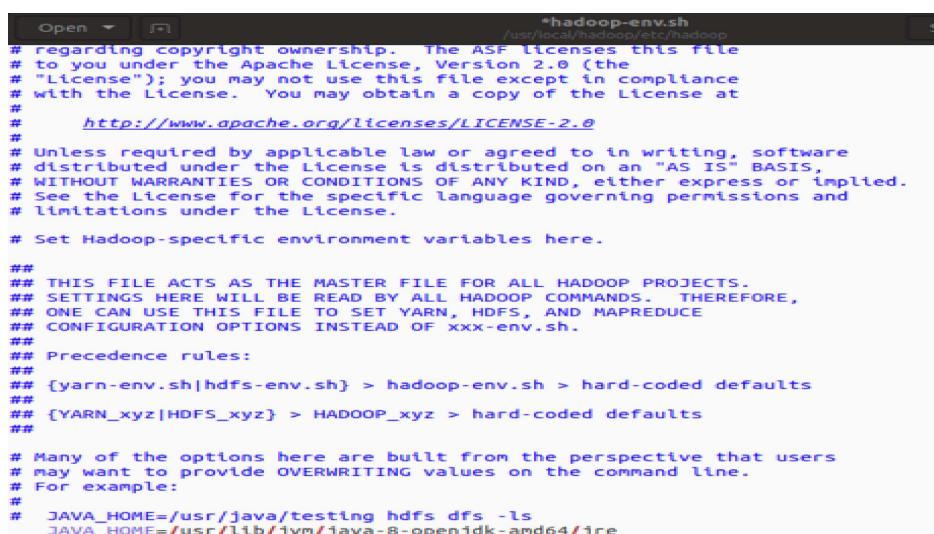
The following syntax maybe used to edit each of the following files:

```
gedit $CONF/<filename>
```

#### **1. hadoop-env.sh**

The variables set in this file are used by hadoop daemon when the hadoop start-up script is called through terminal. The start-up script is called to launch the hadoop cluster.

Define the variable JAVA\_HOME, use the same path as defined for this variable in bashfile. The final version of the file should look similar to the one shown in Fig 1.22.



```
*hadoop-env.sh
#!/bin/bash -x
# regarding copyright ownership. The ASF licenses this file
# to you under the Apache License, Version 2.0 (the
# "License"); you may not use this file except in compliance
# with the License. You may obtain a copy of the License at
#
#     http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.

# Set Hadoop-specific environment variables here.

## ## THIS FILE ACTS AS THE MASTER FILE FOR ALL HADOOP PROJECTS.
## SETTINGS HERE WILL BE READ BY ALL HADOOP COMMANDS. THEREFORE,
## ONE CAN USE THIS FILE TO SET YARN, HDFS, AND MAPREDUCE
## CONFIGURATION OPTIONS INSTEAD OF xxx-env.sh.

## Precedence rules:
## {yarn-env.sh|hdfs-env.sh} > hadoop-env.sh > hard-coded defaults
## {YARN_xyz|HDFS_xyz} > HADOOP_xyz > hard-coded defaults
## 

# Many of the options here are built from the perspective that users
# may want to provide OVERWRITING values on the command line.
# For example:
#
# JAVA_HOME=/usr/java/testing hdfs dfs -ls
# JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64/jre
```

Figure 1.22: final version of hadoop-env.sh file

## 2. core-site.xml

This file informs hadoop daemons about the location of namenode of the cluster. It also contains configuration settings for hadoop core such as I/O settings that are common to HDFS and mapreduce.

The read-only version of this file is called core-default.xml. This file entails all the properties that exist in hadoop core. As this is a read-only file, so any changes that one may need to make to any of these properties should be made in core-site.xml.

You can find the list of all properties and their default parameters of core-default.xml in [hadoop 3.2.2's documentation](#).

We will modify the property called fs.defaultFS, and set the hostname of your hadoop cluster's namenode's hostname with port 9000. The final version of core-site.xml should look similar to the one shown in Fig 1.23. The uri provided as the value of the property is used to determine the host, port, etc. for a filesystem.



The screenshot shows a code editor window titled "core-site.xml" with the path "/usr/local/hadoop/etc/hadoop". The file content is an XML configuration for HDFS. It includes a license notice for the Apache License, Version 2.0, followed by a section for site-specific overrides. A single property is defined under the <configuration> tag:

```
<?xml version="1.0" encoding="UTF-8"?>
<xm...>
<!--
Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License. See accompanying LICENSE file.
-->

<!-- Put site-specific property overrides in this file. -->

<configuration>
    <property>
        <name>fs.defaultFS</name>
        <value>hdfs://BDEA-master:9000</value>
    </property>
</configuration>
```

Figure 1.23: final version of core-site.xml file

## 3. hdfs-site.xml

This file should contain configurations for the following hadoop daemons: namenode, secondary namenode and datanodes. This file can also be used to change the default data block replication behaviour. If you do not specify any property which does not require modification, the default values for that property would be used from hdfs-default.xml file.

You can find the list of all properties and their default parameters of hdfs-default.xml here in [hadoop 3.2.2's documentation](#).

Add the following properties and values pairs to the file: namenode dfs directory and its path; datanode directory and its path; and number of replication blocks. The final version of this file should look similar to the one shown in Fig 1.24.

```

*dfs-site.xml
/usr/local/hadoop/etc/hadoop

http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License. See accompanying LICENSE file.
-->

<!-- Put site-specific property overrides in this file. -->

<configuration>
    <property>
        <name>dfs.namenode.name.dir</name>
        <value>/usr/local/hadoop/data/nameNode</value>
    </property>
    <property>
        <name>dfs.datanode.name.dir</name>
        <value>/usr/local/hadoop/data/dataNode</value>
    </property>
    <property>
        <name>dfs.replication</name>
        <value>2</value>
    </property>
</configuration>

```

Figure 1.24: final version of core-site.xml file

#### 4. workers

Add the hostnames of all worker/data nodes of hadoop cluster in this file. Ensure that the hosts file in etc folder of the system has all the required hostnames mapped to their ip addresses. The final version of workers file should look similar to the one shown in Fig 1.25



Figure 1.25: final version of workers file

#### Copying hadoop configuration files to worker nodes :

Now that all the files which should have common configurations on name and data nodes, have been configured on name node; it is time to copy these files to all the worker/data nodes in hadoop cluster.

You will need to use the *SCP* command to move these files. As we have already established password less SSH connection between all hadoop nodes, the following syntax can be used to copy the files: *scp <absolute path to the file/s> <username>@<worker-hostname>:<path to directory where files are to be copied>* . An example is shown in Fig 1.26. The asterisk in the figure denotes ‘all files’.

```
bdea@BDEA-master:~$ scp $CONF/* bdea@BDEA-worker:/usr/local/hadoop/etc/hadoop
capacity-scheduler.xml          100%  8260     2.8MB/s  00:00
configuration.xsl               100% 1335     829.2KB/s 00:00
container-executor.cfg          100% 1940      1.4MB/s 00:00
core-site.xml                   100%  861     721.1KB/s 00:00
hadoop-env.cmd                 100% 3999      3.0MB/s 00:00
hadoop-env.sh                  100%   16KB    5.1MB/s 00:00
hadoop-metrics2.properties     100% 3321      2.4MB/s 00:00
hadoop-policy.xml              100%  11KB    7.8MB/s 00:00
hadoop-user-functions.sh.example 100% 3414      2.6MB/s 00:00
hdfs-site.xml                  100% 1071     833.8KB/s 00:00
httpfs-env.sh                  100% 1484      1.3MB/s 00:00
httpfs-log4j.properties        100% 1657      1.6MB/s 00:00
httpfs-signature.secret        100%   21      19.6KB/s 00:00
httpfs-site.xml                100%  620     413.8KB/s 00:00
kms-acls.xml                   100% 3518      2.6MB/s 00:00
kms-env.sh                     100% 1351      1.1MB/s 00:00
kms-log4j.properties           100% 1860      1.7MB/s 00:00
kms-site.xml                   100%  682     681.8KB/s 00:00
log4j.properties               100%   13KB    10.1MB/s 00:00
mapred-env.cmd                 100%  951     841.2KB/s 00:00
mapred-env.sh                  100% 1764      1.5MB/s 00:00
mapred-queues.xml.template    100% 4113      3.0MB/s 00:00
mapred-site.xml                100%   758     756.9KB/s 00:00
/usr/local/hadoop/etc/hadoop/shellprofile.d: not a regular file
ssl-client.xml.example         100% 2316      2.0MB/s 00:00
ssl-server.xml.example         100% 2697      2.3MB/s 00:00
user_ec_policies.xml.template 100% 2642      2.7MB/s 00:00
workers                        100%   12      12.6KB/s 00:00
yarn-env.cmd                   100% 2250      2.3MB/s 00:00
yarn-env.sh                     100% 6056      5.4MB/s 00:00
yarnservice-log4j.properties   100% 2591      2.5MB/s 00:00
yarn-site.xml                  100%   690     773.7KB/s 00:00
```

Figure 1.26: Copying files from localhost to remote servers via SCP

### 1.5.6 Worker/s node only configuration

Now that the configured hadoop files with common configurations are setup on all hadoop nodes, files which are supposed to be configured on worker nodes only need to be setup. Configure these files on a worker node and then copy to the any of the rest of the worker/data nodes in the cluster.

We are still editing the files in hadoop configuration directory, so use the same syntax as in previous section to navigate and edit the files.

#### **Yarn-site.xml**

This file contains configuration information that overrides the default values for YARN parameters. The default values for core configuration properties are stored in the Default YARN Parameters file. We already

know from our lecture discussions that YARN stands for yet another resource negotiator and negotiates resources between the hadoop nodes. Additionally we will be executing MapReduce scripts using yarn daemons in order to make use of it's resource negotiations capabilities as required during the map reduce execution.

Add the property which tells the worker nodes the location of the yarn resource manager. For the sake of this tutorial it will be same as the master node. the final version of yarn-site.xml on worker nodes should look similar to the one shown in Fig 1.27.



```
Open  +  yarn-site.xml  
/usr/local/hadoop/etc/hadoop  Save  
  
<?xml version="1.0"?>  
<!--  
 Licensed under the Apache License, Version 2.0 (the "License");  
 you may not use this file except in compliance with the License.  
 You may obtain a copy of the License at  
  
 http://www.apache.org/licenses/LICENSE-2.0  
  
 Unless required by applicable law or agreed to in writing, software  
 distributed under the License is distributed on an "AS IS" BASIS,  
 WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.  
 See the License for the specific language governing permissions and  
 limitations under the License. See accompanying LICENSE file.  
-->  
<configuration>  
  <property>  
    <name>yarn.resourcemanager.hostname</name>  
    <value>BDEA-master</value>  
  </property>  
</configuration>
```

Figure 1.27: Final version of yarn-site.xml on worker/data node/s

## 1.6 Starting the hadoop cluster

Before we may proceed to start the cluster by using any hadoop shell files; name/master node needs to be formatted.

It needs to be ensured that changes made to bash file previously are in effect. If you are unsure, you can always make bashfile the source in the terminal as follows: `source .bashrc`

In order to format the master VM as namenode of the hadoop cluster; enter the following in the master VM's terminal: `hadoop namenode -format`. The beginning and end of your terminal output after executing the aforementioned command should look similar to those shown in Fig 1.28(a) and 1.28(b).

Good job, you have just simulated a bigdata cluster on a local machine using VMs. Now it would be useful

```

bdea@BDEA-master:~ Connection to bdea-worker closed.
bdea@BDEA-master:~$ hadoop namenode -format
WARNING: Use of this script to execute namenode is deprecated.
WARNING: Attempting to execute replacement "hdfs namenode" instead.

WARNING: /usr/local/hadoop/logs does not exist. Creating.
2020-04-06 11:34:39,302 INFO namenode.NameNode: STARTUP_MSG:
/*****
STARTUP_MSG: Starting NameNode
STARTUP_MSG:   host = BDEA-master/192.168.56.103
STARTUP_MSG:   args = [-format]
STARTUP_MSG:   version = 3.2.1
STARTUP_MSG:   classpath = /usr/local/hadoop/etc/hadoop:/usr/local/hadoop/share/
hadoop/common/lib/commons-lang3-3.7.jar:/usr/local/hadoop/share/hadoop/common/li

```

(a) output at the beginning

```

bdea@BDEA-master:~ Cache
2020-04-06 11:34:40,647 INFO util.GSet: VM type      = 64-bit
2020-04-06 11:34:40,647 INFO util.GSet: 0.029999999329447746% max memory 2.8 GB
= 889.7 KB
2020-04-06 11:34:40,647 INFO util.GSet: capacity      = 2^17 = 131072 entries
2020-04-06 11:34:40,686 INFO namenode.FSImage: Allocated new BlockPoolId: BP-120
0949481-192.168.56.103-1586169280673
2020-04-06 11:34:40,718 INFO common.Storage: Storage directory /usr/local/hadoop
/data/nameNode has been successfully formatted.
2020-04-06 11:34:40,779 INFO namenode.FSImageFormatProtobuf: Saving image file /
usr/local/hadoop/data/nameNode/current/fsimage.ckpt_00000000000000000000 using no
compression
2020-04-06 11:34:40,903 INFO namenode.FSImageFormatProtobuf: Image file /usr/loc
al/hadoop/data/nameNode/current/fsimage.ckpt_00000000000000000000 of size 399 byt
es saved in 0 seconds .
2020-04-06 11:34:40,919 INFO namenode.NNStorageRetentionManager: Going to retain
1 images with txid >= 0
2020-04-06 11:34:40,958 INFO namenode.FSImage: FSImageSaver clean checkpoint: tx
id=0 when meet shutdown.
2020-04-06 11:34:40,958 INFO namenode.NameNode: SHUTDOWN_MSG:
/*****
SHUTDOWN_MSG: Shutting down NameNode at BDEA-master/192.168.56.103
*****
```

(b) output at the end

Figure 1.28: Terminal output after master VM is formatted as hadoop namenode

if you think on the following points:

1. how you can replicate this experiment to create an actual big data cluster with 1 tb of RAM?
2. how many PCs you may require?

3. are there any steps involved for which you may need to find alternate in actual scenario?
4. if you had admin rights on J448 machines; could you create the suggested cluster using actual computers? If not, then what needs to be changed in the cluster specifications?