

MS4S21 - Big Data Engineering and Applications

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

University of South Wales



Contents

1 Apache Hadoop multi-node cluster on Ubuntu VMs	3
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
1.6 Starting the hadoop cluster	29
2 Mapreduce on Hadoop Cluster	32
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 locally with python	36
2.4 Map reduce word count task on cluster	38
2.4.1 executing the map reduce task on cluster	39
3 Signing up for AWS educate classrooms	41
4 NoSQL Databases: DynamoDB	42
4.1 Working from your local jupyter notebook in AWS DynamoDB	42
5 Twitter App creation for the module	45
5.1 Twitter developer account and app creation	45
6 Creating a simple hadoop cluster with AWS EMR	47
6.1 Creating the cluster with GUI	47
6.1.1 Overview of the main approach	47
6.1.2 Deep dive into each section of the create cluster webpage	49

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.

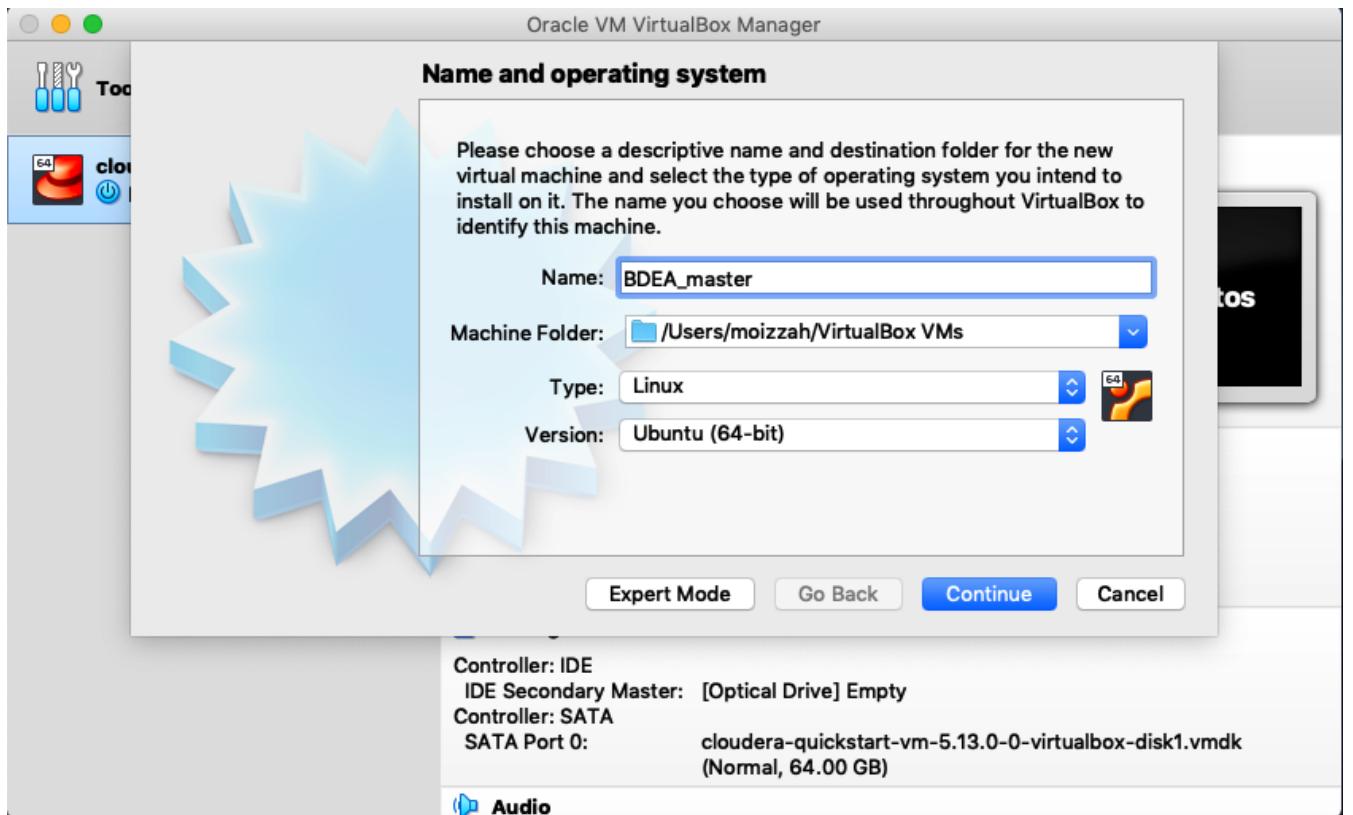


Figure 1.1: Ubuntu VM creation on virtual box - 1

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.

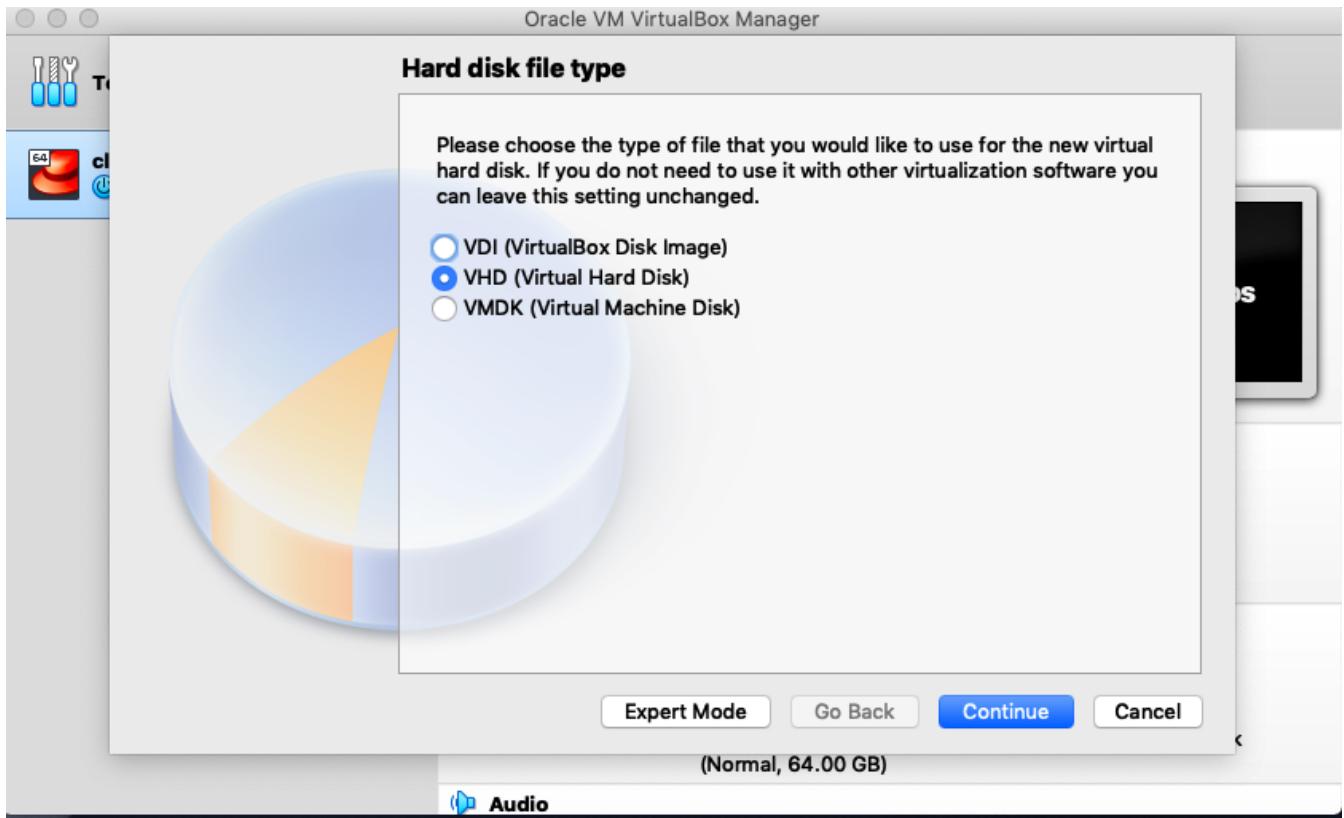


Figure 1.2: Ubuntu VM creation on virtual box - Hard disk file type

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.

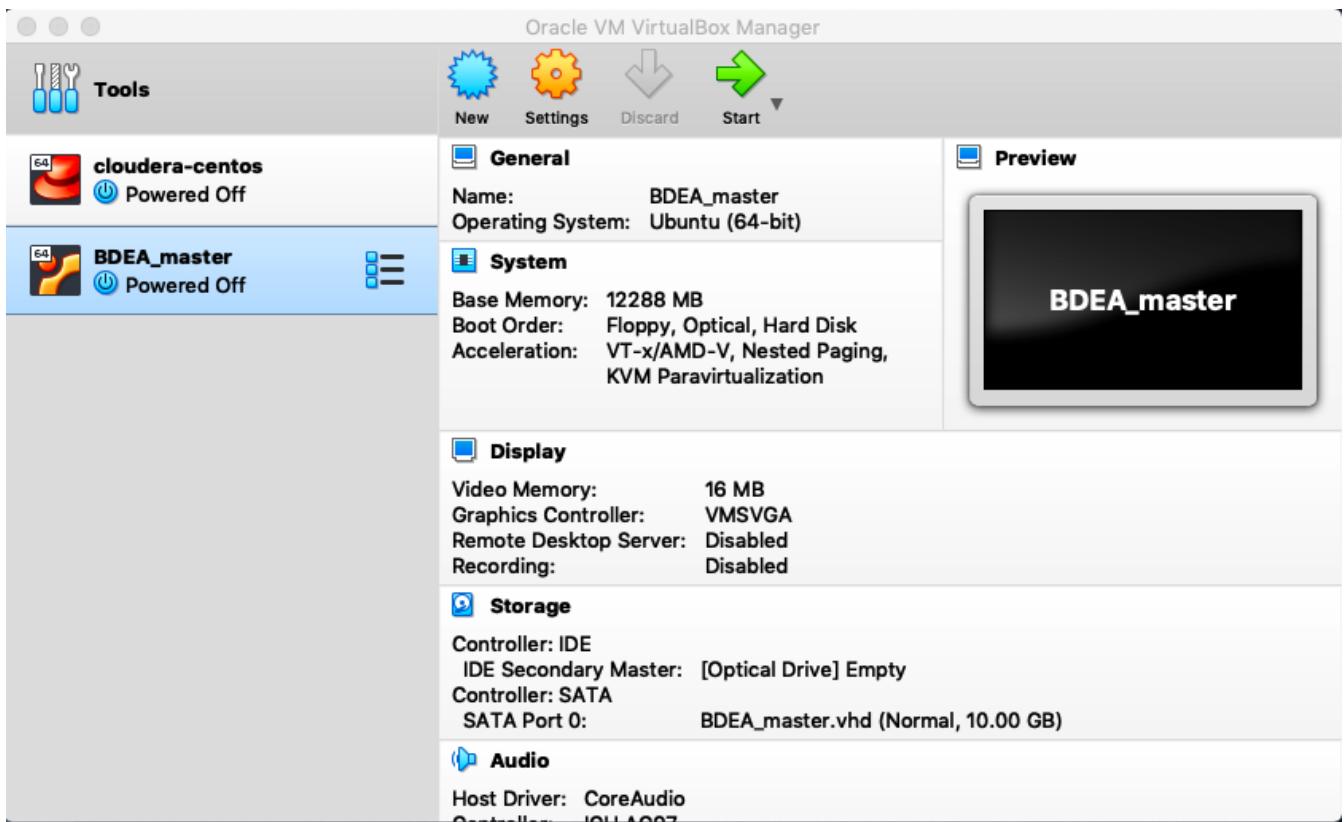


Figure 1.3: Ubuntu VM creation on virtual box - Empty VM

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 and install ubuntu(it's empty already)

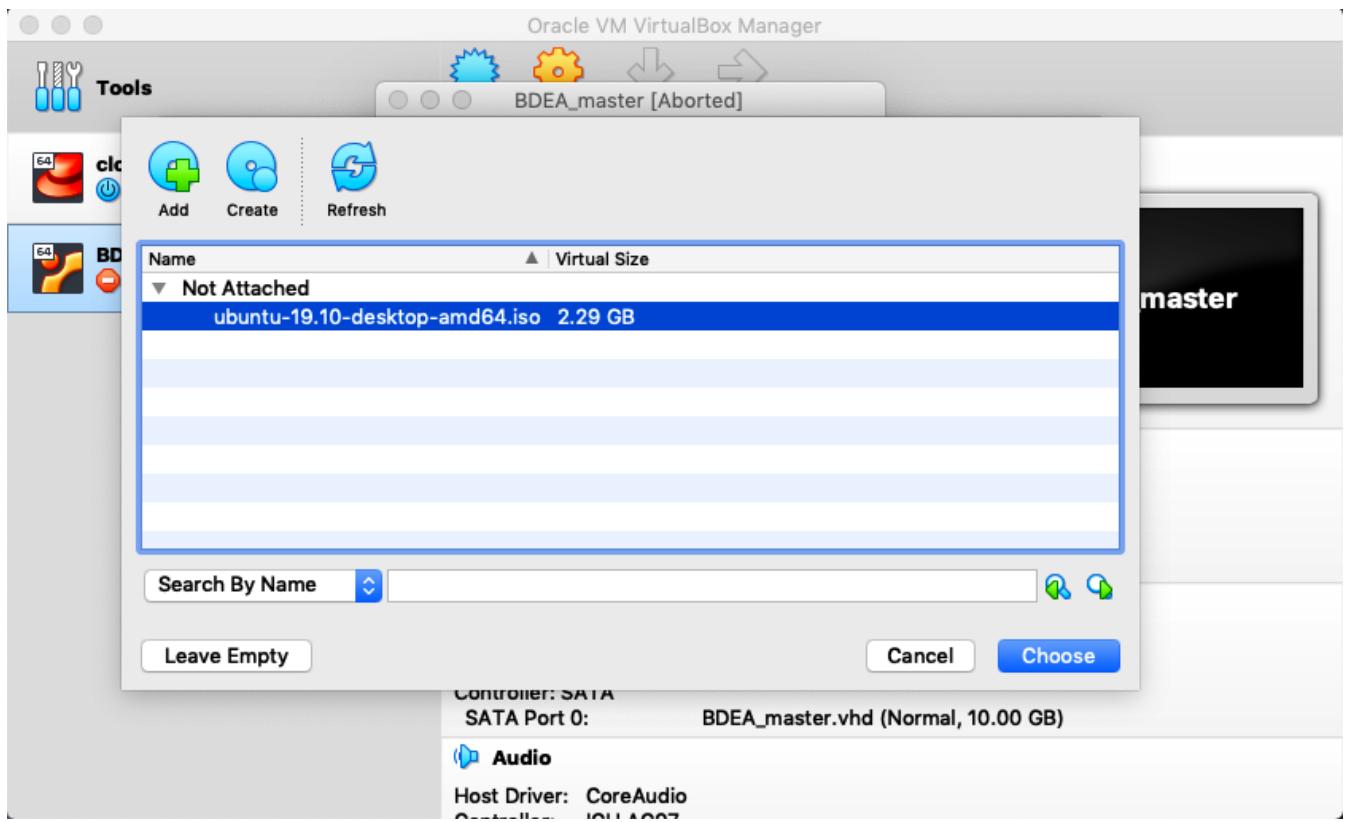


Figure 1.4: Ubuntu VM creation on virtual box - adding iso image 1

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.

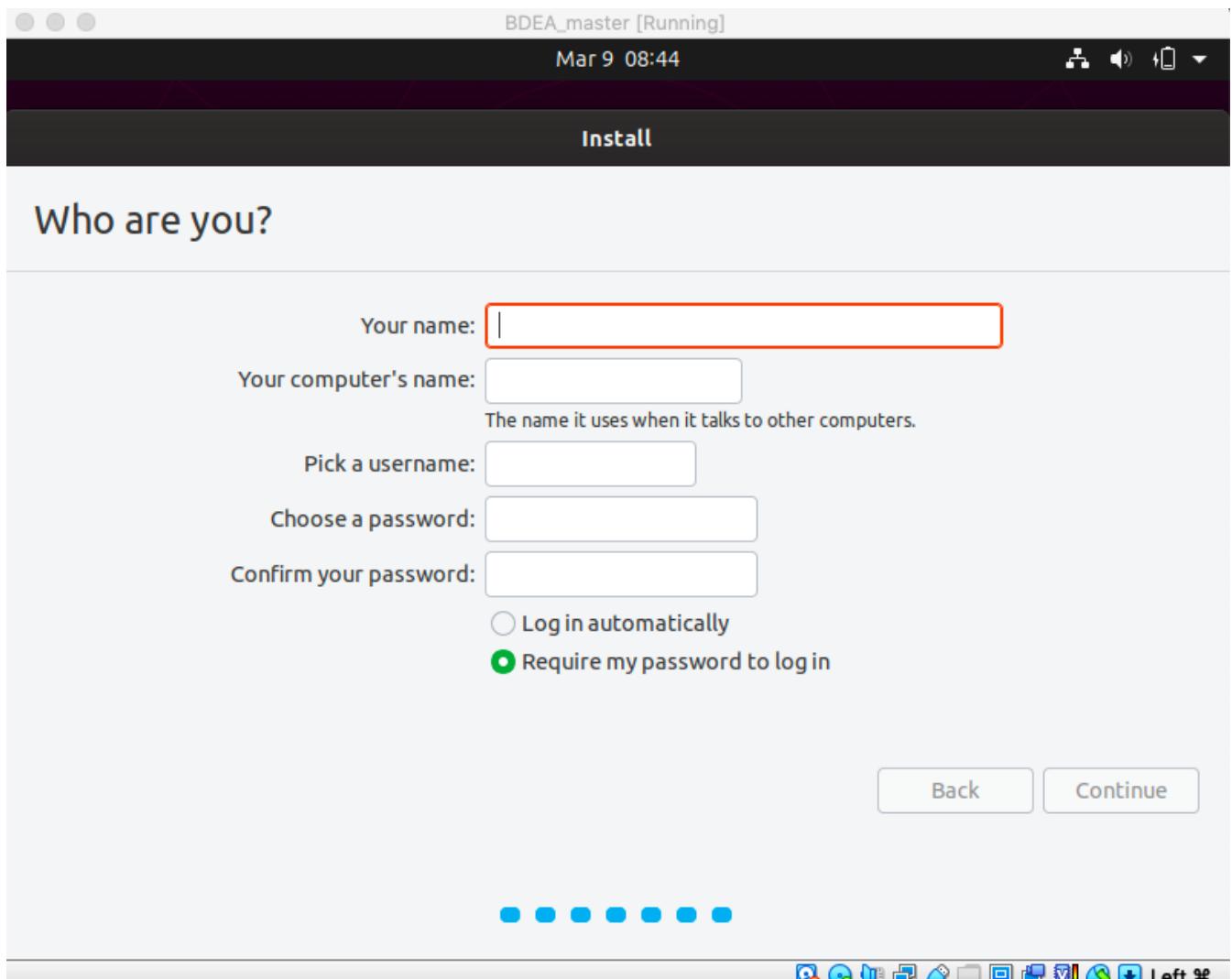


Figure 1.5: Ubuntu user creation

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 - × /etc
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 '\''$s/^\\s*[0-9]\\+\\s*//;s/[;&]\'$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

```

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.

```

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 '\''$/^s*/+\s*//;s/[;&]
\s*alert$'\\''\\''")'

# 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

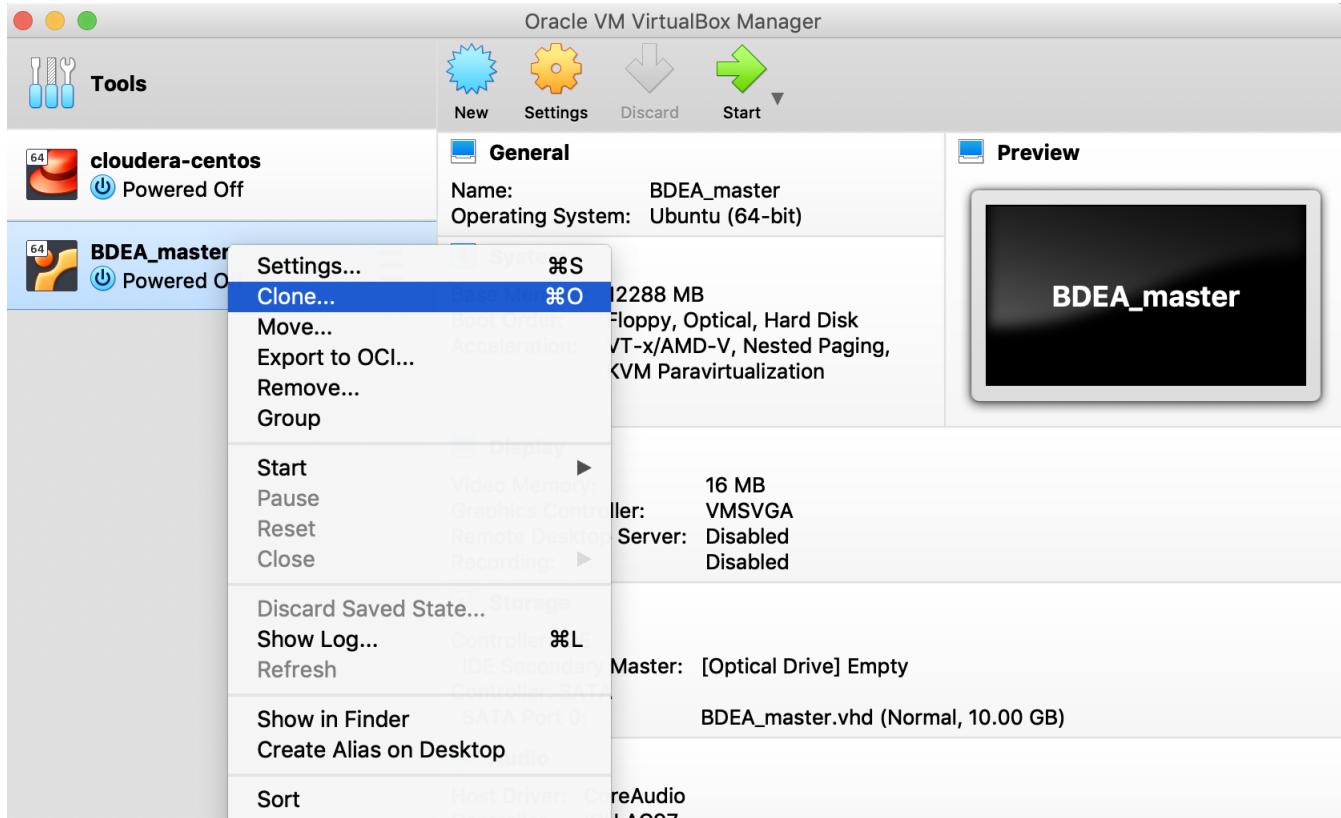


Figure 1.11: Cloning VM - 1

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.

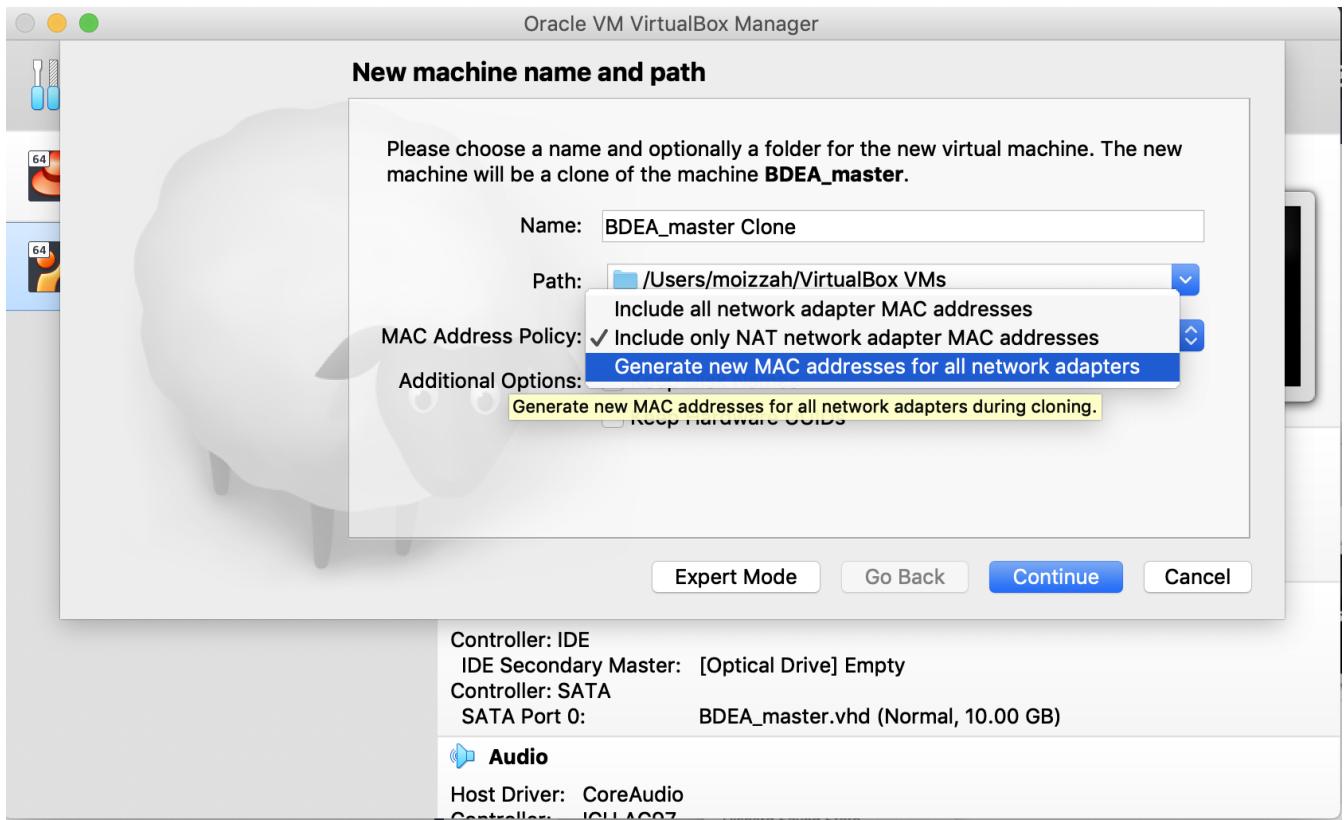


Figure 1.12: Cloning VM - Network adapter configuration

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.

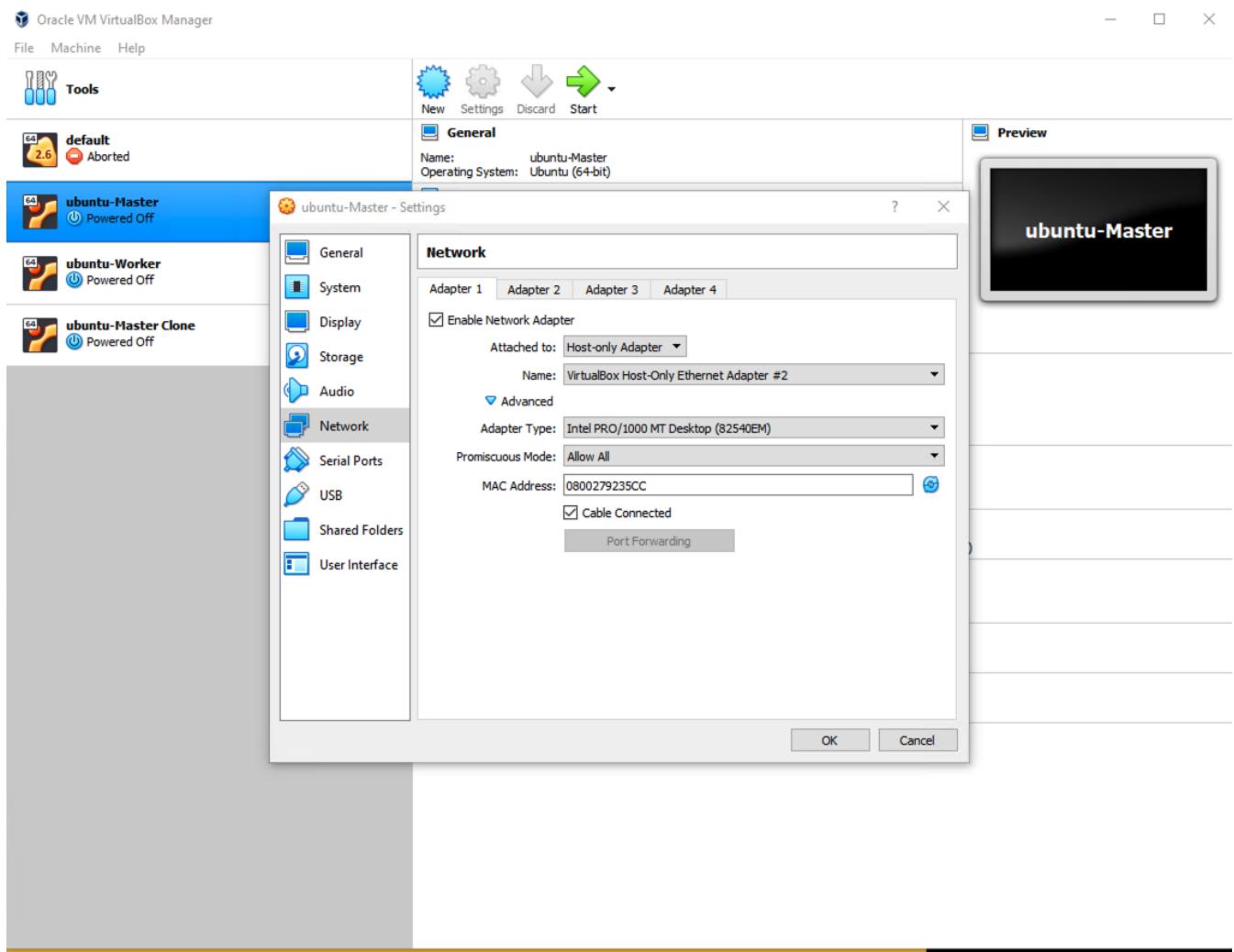


Figure 1.13: Switching to host-only adapter network mode on virtual box windows release

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.

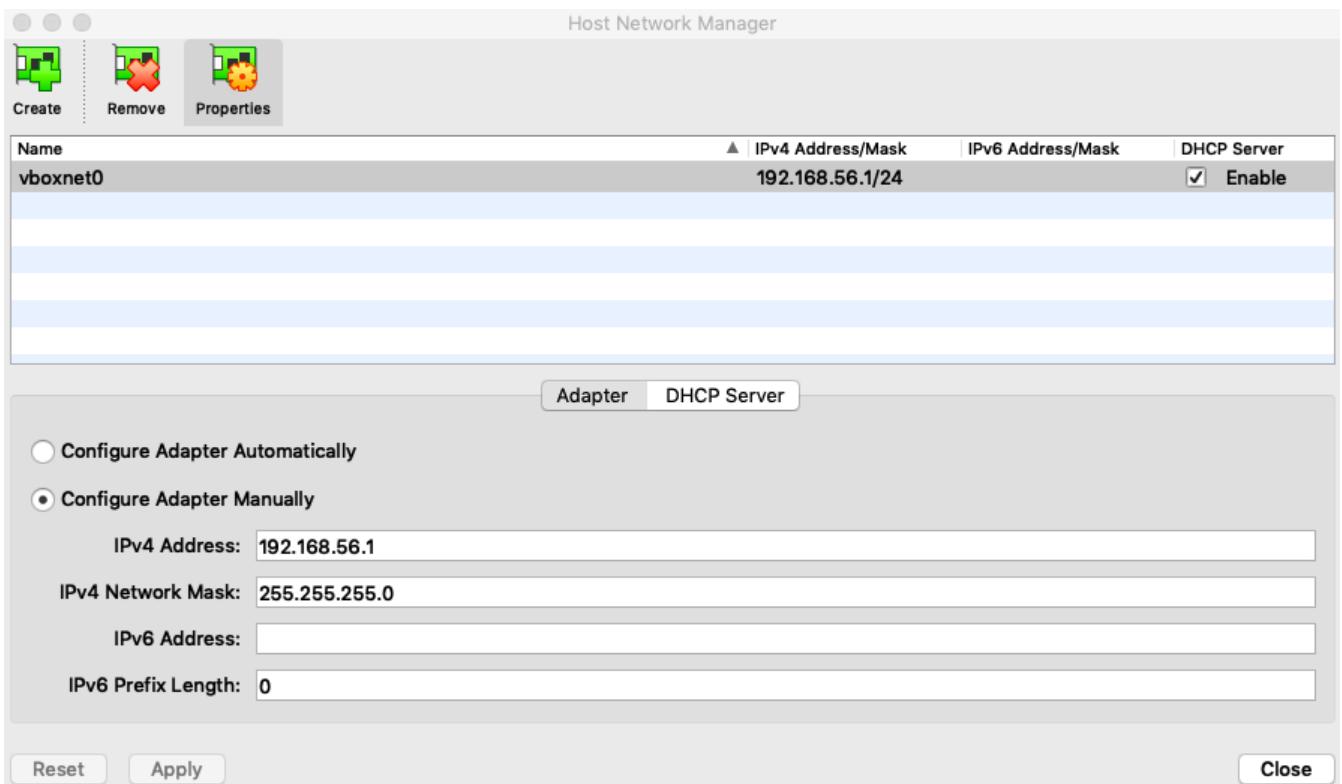


Figure 1.14: Adapter creation for host-only adapter network mode on virtual box macOS release

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.

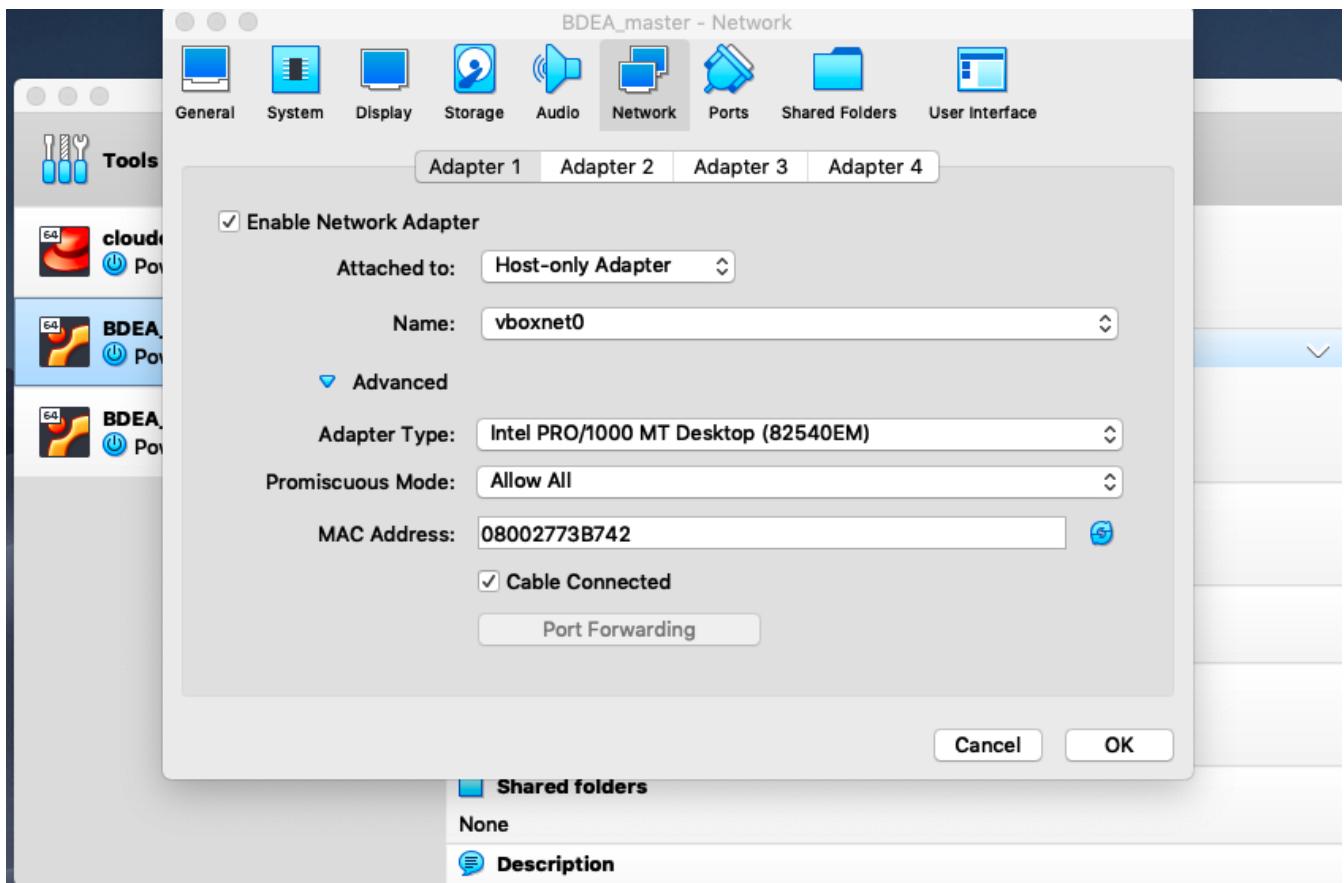


Figure 1.15: Selecting host-only adapter network mode for VMs on virtual box macOS release

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.

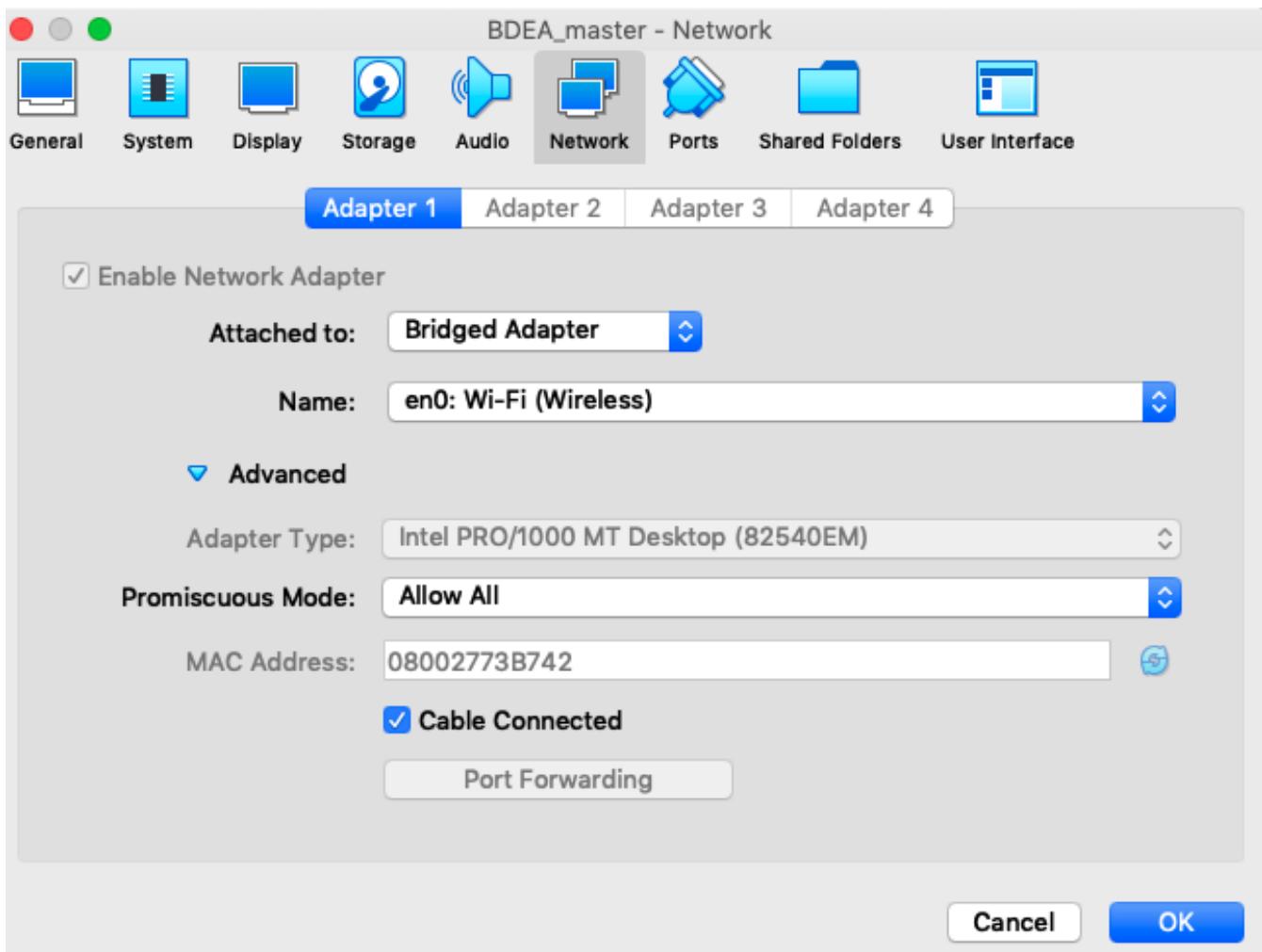


Figure 1.16: Selecting Bridged adapter network mode for VMs on virtual box macOS release

1.5.3 Revisiting system files

Since you have another node now and have chnaged 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: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
Save - 

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.

```
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/yHMf2Af26rVbeDcQ1FnBKRLpY bdea@BDEA-master
The key's randomart image is:
+---[RSA 3072]---+
|.. o oo .oo .|
|.. + o ....+ o |
|o . = o ...o + |
| o o o o..oEoo |
| . + =.S=.+.. |
| . = Oo * . |
| *oo. o |
| oo . |
| oo |
+---[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 out 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 n 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.

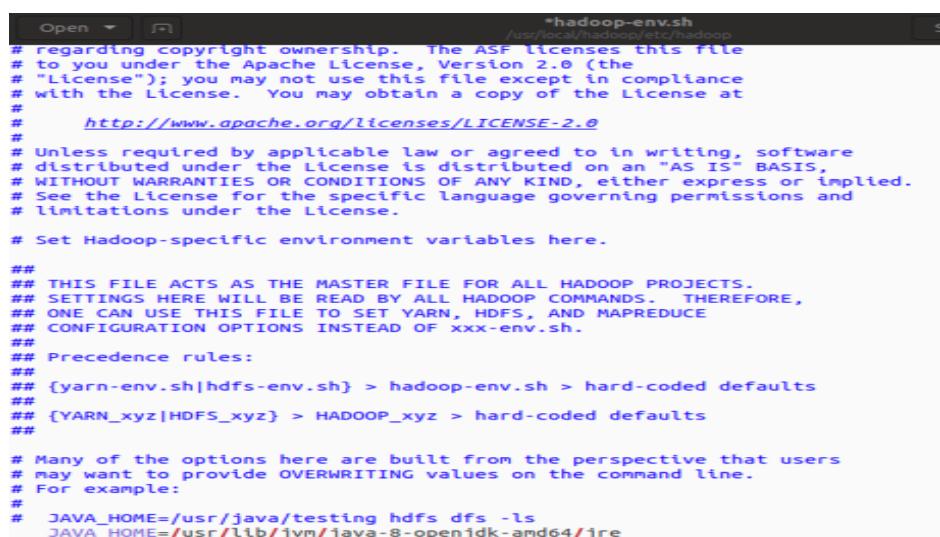
The following syntax maybe used to edit each of the following files:

```
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
/usr/local/hadoop/etc/hadoop
# 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

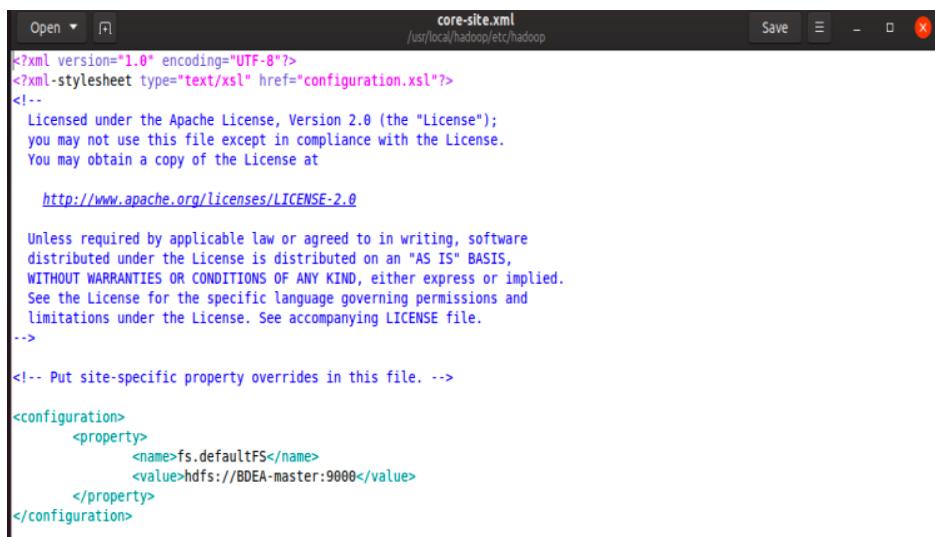
2. core-site.xml

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 text editor window titled "core-site.xml" with the path "/usr/local/hadoop/etc/hadoop". The content of the file is as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<!--
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.

```

*core-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

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.

```

yarn-site.xml
/usr/local/hadoop/etc/hadoop

<?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 formated.

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

```

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

```

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?

Mapreduce on Hadoop Cluster

The merits and demerits of the mapreduce paradigm have been discussed in detail in lectures, along with practical demonstrations. This chapter in the tutorial manual is provided as a reference guide on the practicalities and commands involved. Please use the material uploaded on blackboard to revisit mapreduce paradigm in detail.

2.1 Configuration on hadoop cluster

There are certain configurations required on all worker/data nodes and master node respectively. The following sub sections, walk through them.

2.1.1 Worker/data nodes configuration

Yarn-site.xml

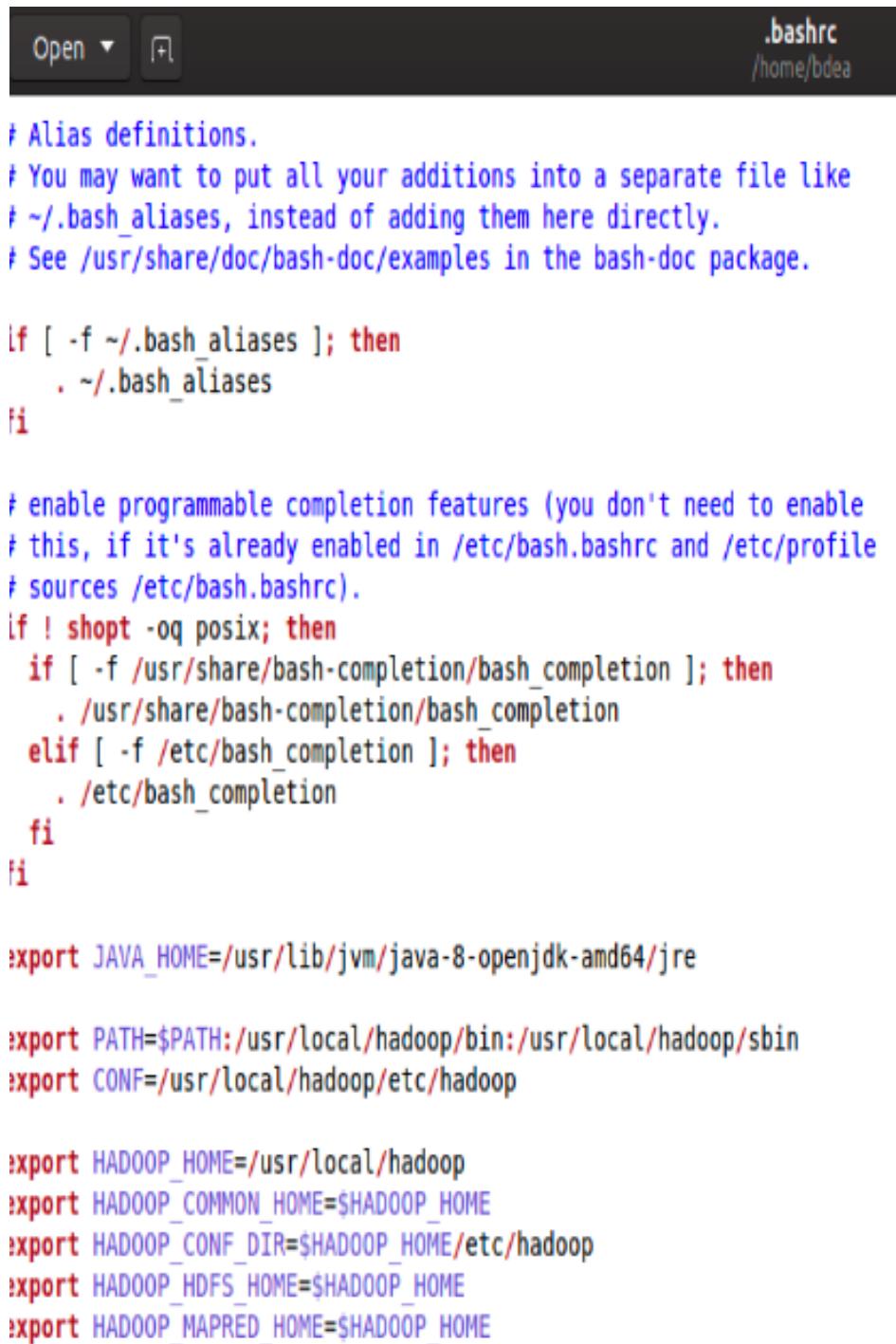
It is essential that all worker nodes in the hadoop cluster should have yarn-site.xml configured as shown previously in the lecture and Section 1.5.5 of this lab-tutorial manual. Note, that yarn-site.xml file in master/name node should contain empty configuration tags.

2.1.2 Master/name node configuration

In order to execute a mapreduce task successfully on the cluster certain variables need to be set in your master/data node's bash file. These variables are required by different hadoop shell scripts when hadoop daemons are executed. These variables and their values are as follows:

```
export HADOOP_HOME=/home/ubuntu/hadoop
export HADOOP_COMMON_HOME=$HADOOP_HOME
export HADOOP_CONF_DIR=$HADOOP_HOME/etc/hadoop
export HADOOP_HDFS_HOME=$HADOOP_HOME
export HADOOP_MAPRED_HOME=$HADOOP_HOME
```

The final outlook of the bash file should look similar to the one shown in Fig 2.1.



```

.bashrc
/home/bdea

# 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

export HADOOP_HOME=/usr/local/hadoop
export HADOOP_COMMON_HOME=$HADOOP_HOME
export HADOOP_CONF_DIR=$HADOOP_HOME/etc/hadoop
export HADOOP_HDFS_HOME=$HADOOP_HOME
export HADOOP_MAPRED_HOME=$HADOOP_HOME

```

Figure 2.1: Final version of bash file on master/name node

2.2 Executing a demo mapreduce job on the cluster

Before executing the demo job, you need to make sure that a password less ssh connection is established between all the servers in the cluster, including between localhost on master VM and master/data node on master VM. Additionally, please ensure that you format the master/name node before starting the cluster.

1. Start hadoop on master/name node - *start-dfs.sh*.
2. Type *jps* on your terminal; this commands lets you see the java processes in execution on your machine. The terminal output on name node should look similar to those shown in Fig 2.2(a) & 2.2(b).

```
bdea@BDEA-master:~$ start-dfs.sh
Starting namenodes on [BDEA-master]
Starting datanodes
Starting secondary namenodes [BDEA-master]
bdea@BDEA-master:~$ jps
5824 NameNode
6151 Jps
6057 SecondaryNameNode
bdea@BDEA-master:~$ █
```

(a) jps output on master VM

```
bdea@BDEA-worker:~$ jps
4720 Jps
4660 DataNode
bdea@BDEA-worker:~$ █
```

(b) jps output on worker VM

Figure 2.2: List of Java processes running on master and worker VMs of hadoop cluster after the cluster has been started.

3. It is essentially required to start Yarn before you can execute any mapreduce job. You can start yarn by executing the *start-yarn.sh* shell script from hadoop config directories. The output after calling this script and list of java processes on name and data nodes terminal should look similar to those shown in Fig

```
bdea@BDEA-master:~$ start-yarn.sh
Starting resourcemanager
Starting nodemanagers
bdea@BDEA-master:~$ jps
5824 NameNode
6057 SecondaryNameNode
9337 ResourceManager
9613 Jps
bdea@BDEA-master:~$
```

(a) jps output on master VM

```
bdea@BDEA-worker:~$ jps
4720 Jps
4660 DataNode
bdea@BDEA-worker:~$ jps
4660 DataNode
4660 DataNode
7385 Jps
7295 NodeManager
bdea@BDEA-worker:~$
```

(b) jps output on worker VM

Figure 2.3: List of Java processes running on master and worker VMs of hadoop cluster after starting yarn on cluster.

4. You can also view the list of yarn node manager daemons running on the worker/data nodes in your cluster by typing *yarn node -list* on the master/name node's terminal. The output list should contain details of resource managers on all the worker nodes and resemble to the one in Fig 2.4.

```
bdea@BDEA-master:~$ yarn node -list
2020-04-07 17:59:21,315 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032
Total Nodes:1
      Node-Id          Node-State Node-Http-Address      Number-of-Running-Containers
      BDEA-worker:43943      RUNNING   BDEA-worker:8042                      0
bdea@BDEA-master:~$
```

Figure 2.4: List of Resource managers running in the cluster

5. The hadoop cluster comes with an example mapreduce programme saved in a .jar file. The

programme calculates the value of pi based on the values of points on a circle's circumference and its radius. Executing this programme can ensure that your cluster is capable of executing a simple mapreduce job.

Type the following on master VM's terminal to run the demo map reduce task:

```
yarn jar /usr/local/hadoop/share/hadoop/mapreduce/hadoop-mapreduce-examples-3.2.1.jar pi 16 1000
```

Please watch the lecture recording to find out more about the command and its arguments. The beginning and end of this command's output in terminal should resemble to those shown in Fig

```
bdea@BDEA-master:~$ yarn jar /usr/local/hadoop/share/hadoop/mapreduce/hadoop-mapreduce-examples-3.2.1.jar pi 16  
100  
Number of Maps = 16  
Samples per Map = 100  
2020-04-07 18:20:58,129 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false,  
remoteHostTrusted = false  
Wrote input for Map #0  
2020-04-07 18:20:58,229 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false,  
remoteHostTrusted = false  
Wrote input for Map #1
```

(a)

```
Shuffled Maps =16  
Failed Shuffles=0  
Merged Map outputs=16  
GC time elapsed (ms)=423  
Total committed heap usage (bytes)=3242065920  
Shuffle Errors  
BAD_ID=0  
CONNECTION=0  
IO_ERROR=0  
WRONG_LENGTH=0  
WRONG_MAP=0  
WRONG_REDUCE=0  
File Input Format Counters  
Bytes Read=1888  
File Output Format Counters  
Bytes Written=97  
Job Finished in 3.761 seconds  
Estimated value of Pi is 3.1500000000000000000000000000000
```

(b) end

Figure 2.5: demo mapreduce job execution output

2.3 A word count mapreduce task locally with python

Recall, that a map step transforms the raw input data into key/value pairs and the reduce step transforms the key/value pairs into desired output. For a word count example, a map step may take a text file and

convert it into key/value pairs. The words will be keys and each word's value will be 1.

Likewise, the reduce step will combine all duplicate keys by adding their values. Since every key has value one, so the reducer will find the unique words and sum their values.

Figure 2.6 shows python code to execute the local implementation of a map task. The map task is reading a text file line by line, stripping its whitespace, splitting it into individual words, and then assigning the word to a key and initialising its value.

```
In [35]: txt = open('/Users/moizzah/Desktop/pg2701.txt', 'r')

In [36]: f = open('test.txt', 'a+')

for line in txt:

    print(line)
    print(keys)
    for key in keys:
        value = 1
        print("%s\t%d" % (key, value), file = f)

In [37]: txt.close()
f.close()
```

Figure 2.6: local implementation of word count mapper

If you run this implementation on the text file provided in blackboard, the output file, called test.txt should have similar initial output as shown in Figure 2.7

```

1 •The---*1
2 Project---*1
3 Gutenberg---*1
4 EBook---*1
5 of---*1
6 Moby---*1
7 Dick;---*1
8 or---*1
9 The---*1
10 Whale,---*1
11 by---*1
12 Herman---*1
13 Melville---*1
14 This---*1
15 eBook---*1
16 is---*1
17 for---*1
18 the---*1
19 use---*1
20 of---*1
21 anyone---*1
22 anywhere---*1
23 at---*1
24 no---*1
25 cost---*1
26 and---*1
27 with---*1
28 almost---*1
29 no---*1
30 restrictions---*1
31 whatsoever.*1
32 You---*1
33 may---*1
34 copy---*1
35 it,*1
36 give---*1
37 it---*1
38 away---*1
39 or---*1
40 re-use---*1
41 it---*1
42 under---*1
43 the---*1

```

Figure 2.7: Screenshot from local mapper's output file

2.4 Map reduce word count task on cluster

Given that you have the text file on your local machine which you want to process using a map reduce programme, you will have to send it to the cluster, so that it can be loaded into HDFS.

Assuming that the file to be sent to the cluster is ‘pg2701.txt’, you will have to share it with the master node using scp as follows:

```
scp -i <path to .pem file> <path to pg2701.txt> ubuntu@<master's public dns>:
```

However, as we are not working on a remote cluster we can download all the files required for this task from BB, including: mapper.py, reducer.py and pg2701.txt.

The logic behind the mapper code has already been discussed in live session as well video recording.

The working of the reducer script can be summarised as follows: When it receives a key it compares the key with previous key. If the key is similar, then the value of the key is added to the previous value of the key. Whereas, if the key is not similar the script prints the previous key and its last value; and then makes the new key previous key as well as restart the counter.

This logic may seem incomplete at first as the script loses its progress with previous key as soon as it encounters a new key. However, there is a hidden step that takes place between map and reduce tasks, called shuffle. The map reduce paradigm and hadoop services transparently takes care of executing the shuffle step.

The shuffle steps involves the following:

- sorting all key/value pairs before sending them to reducer
- key/value pairs with common keys are sent to the same reducer

Hence, as many reducer will run as many unique keys exist. So, it is very important that keys are created keeping this point in mind.

2.4.1 executing the map reduce task on cluster

Now you have the text file on which word count map reduce programme has to be executed on the Hadoop cluster as well as the mapper and reducer .py files which will execute this task. You need to follow the steps below to successfully execute this map reduce job on the server:

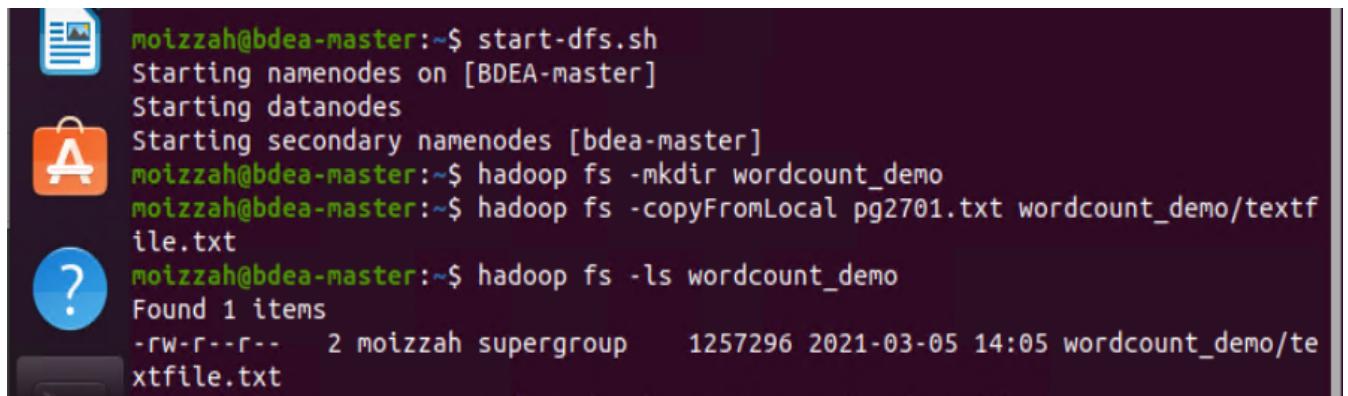
1. load the text file into HDFS by executing the following hadoop commands on master/name node's terminal:

```
hadoop fs -mkdir wordcount_demo - you have created a hdfs directory/folder called wordcount_demo.
```

```
hadoop fs -copyFromLocal pg2701.txt wordcount_demo/textfile.txt - you are saving the file from local file system to hdfs while renaming it as textfile.txt.
```

you can ensure that the file was loaded by typing the following command, user result should look similar to the output in Figure 2.8

```
hadoop fs -ls wordcount_demo/textfile.txt
```



A screenshot of a terminal window showing the execution of HDFS commands. The terminal is running on a Linux system with a dark theme. The commands entered are:

```
moizzah@bdea-master:~$ start-dfs.sh
Starting namenodes on [BDEA-master]
Starting datanodes
Starting secondary namenodes [bdea-master]
moizzah@bdea-master:~$ hadoop fs -mkdir wordcount_demo
moizzah@bdea-master:~$ hadoop fs -copyFromLocal pg2701.txt wordcount_demo/textfile.txt
moizzah@bdea-master:~$ hadoop fs -ls wordcount_demo
Found 1 items
-rw-r--r-- 2 moizzah supergroup 1257296 2021-03-05 14:05 wordcount_demo/textfile.txt
```

Figure 2.8: Ensuring that file has been copied to hdfs

2. execute the following command on the master's terminal to run the mapreduce job on cluster:

```
yarn jar /usr/local/hadoop/share/hadoop/tools/lib/hadoop-streaming-3.2.2.jar -files <path to mapper.py file on master node>,<path to reducer.py file on master node>-mapper '/path/to/mapper.py' -reducer '/path/to/reducer.py' -input 'wordcount_demo/textfile.txt' -output 'wordcount_demo/output_demo'
```

Signing up for AWS educate classrooms

You would have received an e-mail on the 26th of February 2021 about AWS Educate invitation. Before Coming to Week 4's scheduled live session you should have access to the AWS educate classroom.

The following guideline may help you to do so:

1. Click on the link provided in the e-mail with subject ‘Your AWS Educate Application’.
2. Accept the invitation.
3. You will be sent a verification e-mail, get verified using the link.
4. You will need to sign in to AWS educate to access the classroom. Navigate to AWS educate home using the following link: <https://aws.amazon.com/education/awseducate/>.
5. Click on the ‘sign in to AWS Educate’ link in blue. You can find this link just below the orange ‘Join Aws Educate button’.
6. Enter your e-mail address (University e-mail address as you have received invite on this address) and click on the ‘Forgot password’ link in blue. You will be navigated to a new page.
7. Enter your e-mail address in the ‘Username’ field on the new page and press the Blue ‘Send Password Reset Email’ button.
8. You will receive an e-mail with a subject line ‘Welcome To AWS Educate’. Click on the first link provided in this e-mail.
9. You will be navigated to a sign up page, where you will be required to enter personal details such as full name, DOB and etc. Please ensure that University of South Wales and your university e-mail address appear by default and remains in-editable on this page. Once done, submit the form.
10. You should now receive an e-mail with the following subject: ‘AWS Educate Application Approved’.
11. log into your AWS educate account and ensure that you can access the Big data classroom.

NoSQL Databases: DynamoDB

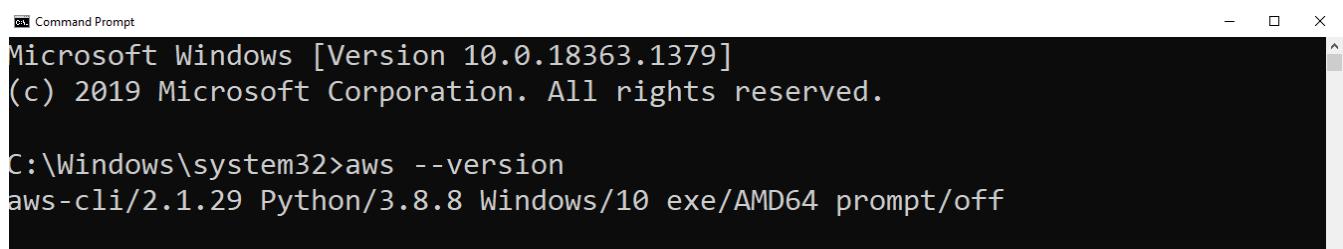
4.1 Working from your local jupyter notebook in AWS DynamoDB

Follow this section only after you have worked the Lab manual with hands on activity to create a Dynamo Table.

This section will introduce you to setting up a python API for dynamodb in your local jupyter notebook.

You have to ensure that you are logged into your AWS educate classroom and have internet connection before going forwards with this manual.

1. The python API that we will use to work on aws dynamodb is boto3. Please ensure that you have installed boto3. You can boto3's documentation here: [Boto3's documentation](#).
2. Now you need to configure AWS configuration on your system. We will use AWS CLI client. Please download the version suitable for your OS from this weblink: [AWS CLI Version 2](#).
3. After successfully installing the AWS CLI, open your OS's cmd/terminal and verify successful installation as shown in fig 4.1



```
Windows PowerShell
Copyright (c) Microsoft Corporation. All rights reserved.

C:\Windows\system32>aws --version
aws-cli/2.1.29 Python/3.8.8 Windows/10 exe/AMD64 prompt/off
```

Figure 4.1: Verifying installation success for AWS CLI

4. Navigate to your AWS classroom and click on the Account Details button from homepage as shown in fig 4.2

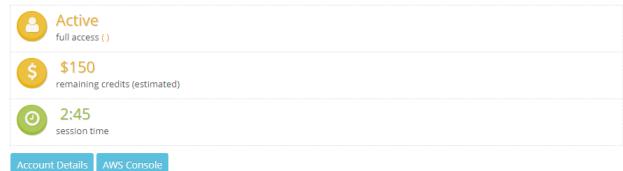
Welcome to your AWS Educate Account

AWS Educate provides you with access to a wide variety of AWS Services for you to get your hands on and build on AWS! To get started, click on the AWS Console button to log in to your AWS console.

Please read the FAQ below to help you get started on your Starter Account.

- What are the list of services supported?
- What regions are supported with Starter Accounts or Classroom Accounts?
- I can't start any resources. What happened?
- Can I create users within my Starter or Classroom Account for others to access?
- Can I create my own IAM policy within Starter Account or Classroom?
- Can I use marketplace software with my Starter Account or Classrooms?
- Are there any restrictions on AWS services in my AWS Educate Account?
- Are FPGA Instances Supported?
- How do I share image with my students?
- Can I access the billing and cost console?

Your AWS Account Status



Please use AWS Educate Account responsibly. Remember to shut down your instances when not in use to make the best use of your credits. And, don't forget to logout once you are done with your work!

Figure 4.2: AWS educate classroom homepage

5. Click on the show button placed in front of AWS CLI as shown in fig 4.3

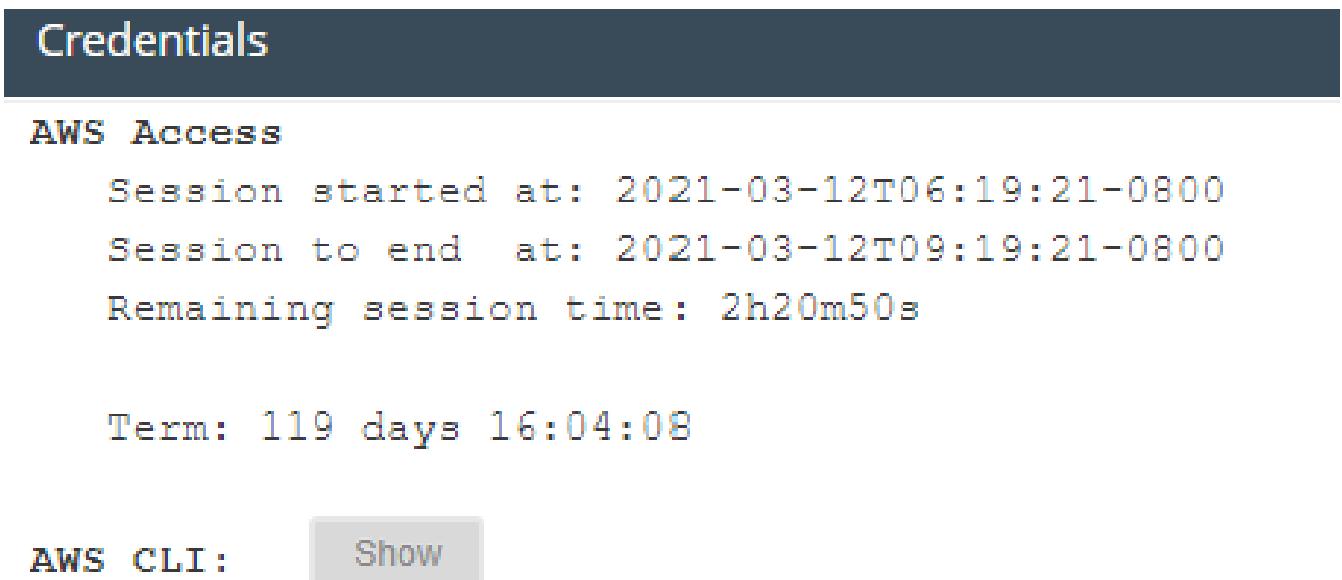


Figure 4.3: Acquiring AWS credentials for classroom console

6. Go back to your OS's cmd/terminal and type 'aws configure'. You will be able to setup the aws configuration required for boto3 to establish connection with dynamodb. Add the credentials from previous step as shown in fig 4.4. Please don't share your credentials publicly.

```
C:\Windows\system32>aws configure
AWS Access Key ID [None]: ASIAQQV...[REDACTED]
AWS Secret Access Key [None]: smWWk...[REDACTED]
Default region name [None]:
Default output format [None]:
```

Figure 4.4: Configuring AWS credentials

7. Now copy the session token an navigate to the user folder which is logged in to your OS. For example the path to my user folder in windows is as follows: ‘C:\Users\masif’.
8. You will find a folder called ‘.aws’, open this folder
9. Open the file called credentials with any text editor and copy paste the session key here.
10. now restart the kernel in your Jupyter notebook and see if the sample python code shown in fig ?? works in your jupyter notebook without any error. We will discuss this code next week.

```
In [1]: import boto3
In [3]: client = boto3.client('dynamodb')
In [4]: client.list_tables()
Out[4]: {'TableNames': ['Music'],
         'ResponseMetadata': {'RequestId': 'IC850PITQE9F8AT7NM1DJDAA6VVV4KQNS05AEMVJF66Q9ASUAAJG',
                             'HTTPStatusCode': 200,
                             'HTTPHeaders': {'server': 'Server',
                                             'date': 'Fri, 12 Mar 2021 14:38:47 GMT',
                                             'content-type': 'application/x-amz-json-1.0',
                                             'content-length': '24',
                                             'connection': 'keep-alive',
                                             'x-amzn-requestid': 'IC850PITQE9F8AT7NM1DJDAA6VVV4KQNS05AEMVJF66Q9ASUAAJG',
                                             'x-amz-crc32': '3485585'},
                             'RetryAttempts': 0}}
```

Figure 4.5: Sample boto3 code

11. You should be able to see the name of the table you have created using the dynamodb web console in the response to your query executed as shown in fig 4.5.

Twitter App creation for the module

Please follow the steps to create a Twitter app which you will use in week4 and 5 would come handy when attempting your coursework.

5.1 Twitter developer account and app creation

Required - intended for morning asynchronous hour study

Please follow the steps to create a Twitter app which you will use in the live session.

1. Sign in to twitter.com using your twitter credentials. If you don't have a twitter account, you may need to sign up first.
2. Open Twitter developers' website <https://developer.twitter.com/>, Click on tab located just before your profile picture called 'Developer Portal'.
3. Go to Projects and Apps section using the left side menu, and select overview.
4. Scroll down the page, you will find a button '+create App' at the bottom of the page.
5. Give your app a meaningful name and don't forget to copy paste API key, API secret Key and Bearer token in a text document.
6. Select appropriate reason i.e. student under academic category, and switch to individual developer account.
7. Please fill the form **How will you use Twitter API or Twitter Data** as follows:
 - (a) **first field:** To learn some topics of Data Science as a student of MS4S21 at the University of South Wales. The aim of using this app are as follows:
 - Using python API for twitter
 - Understanding the JSON format of response
 - Saving the data in a NoSQL database

(b) **second field specifics:** Will download public tweets to demonstrate: the use of python API for twitter
how to save public tweets downloaded from python API in DynamoDB
finding tweets containing trending hashtags to analyse their textual content, and other response fields, such as: location.

(c) Uncheck (**NO**) the rest of the questions

8. Now you can actually start creating an app. Click apps again, create an app and fill in the form **App Details** as follows:

(a) **App name:** MS4S21_USW_Learning

(b) **App description:** This App will be used to learn and teach some components of programming for data analysis at the University of south wales.
This App will be used to learn and teach some module contents of Big data engineering and applications at the University of South Wales.

(c) **URL:** <https://www.southwales.ac.uk/courses/msc-data-science/>

(d) **callback URL:** <https://www.southwales.ac.uk/courses/msc-data-science/>

Creating a simple hadoop cluster with AWS EMR

Reading and understanding the supplementary resource uploaded just before this manual in week 6 is a pre-requisite to the lab work for EMR.

6.1 Creating the cluster with GUI

You will create a simple EMR cluster using the AWS console provided in AWS educate classroom for MS4S21.

The following list provides you with the basic overview of how to create a cluster using the GUI. Each item from this option would be discussed in subsequent lists and subsections.

6.1.1 Overview of the main approach

1. Log on to aws educate with your university e-mail address.
2. Navigate to classroom for MS4S21.
3. click on the blue button which says AWS Console.
4. Scroll down through the services displayed on AWS dashboard to find the heading for Analytics services.
5. Select EMR, you should be navigated to EMR homepage which should look similar to webpage shown in Fig 6.1:

Figure 6.1: EMR AWS homepage

6. Click on the blue button which says ‘Create Cluster’.
7. You will be rendered to a form which basically is a AWS GUI to create a cluster. Initially most of the fields in the form will be empty with minimal options as shown in Fig 6.2. However you should wait until next step.

Figure 6.2: EMR Cluster creation GUI initial view

8. You will notice that after a few moments the GUI form has updated itself and had rendered more options as shown in Fig. 6.3

General Configuration

Cluster name [i](#)
 Logging [i](#)
S3 folder [i](#)
Launch mode Cluster [i](#) Step execution [i](#)

Software configuration

Release [i](#)
Applications Core Hadoop: Hadoop 2.10.1, Hive 2.3.7, Hue 4.9.0, Mahout 0.13.0, Pig 0.17.0, and Tez 0.9.2
 HBase: HBase 1.4.13, Hadoop 2.10.1, Hive 2.3.7, Hue 4.9.0, Phoenix 4.14.3, and ZooKeeper 3.4.14
 Presto: Presto 0.245.1 with Hadoop 2.10.1 HDFS and Hive 2.3.7 Metastore
 Spark: Spark 2.4.7 on Hadoop 2.10.1 YARN and Zeppelin 0.9.0
 Use AWS Glue Data Catalog for table metadata [i](#)

Hardware configuration

Instance type [i](#) The selected instance type adds 64 GiB of GP2 EBS storage per instance by default. [Learn more](#)
Number of instances (1 master and 2 core nodes)
Cluster scaling scale cluster nodes based on workload

Security and access

EC2 key pair [i](#) Learn how to create an EC2 key pair.
Permissions Default Custom
Use default IAM roles. If roles are not present, they will be automatically created for you with managed policies for automatic policy updates.
EMR role [i](#)
EC2 instance profile [i](#)

[Cancel](#) [Create cluster](#)

Figure 6.3: EMR Cluster creation GUI updated view

The next section will look at each section of the create cluster web page individually.

6.1.2 Deep dive into each section of the create cluster webpage

General configuration

General Configuration

Cluster name [i](#)
 Logging [i](#)
S3 folder [i](#)
Launch mode Cluster [i](#) Step execution [i](#)

Figure 6.4: EMR cluster creation - General Configuration

1. **Cluster name** - As usual provide a meaningful and intuitive name to your cluster.
2. **Logging** - If you enable this option, a log of your cluster activities would be maintained. The log has to be maintained in an AWS S3 bucket. Enabling

this option in turn enables the S3 address field. You may have noticed that a path to a new S3 bucket is automatically provided. However, you may choose a pre-existing bucket as well.

For this lab exercise select logging and don't edit the default S3 bucket uri.

3. **Launch mode** - As evident from the form design that there are two possible modes with you can launch and EMR cluster.

- (a) **Cluster**: This option enables you to create a cluster and add services from Apache's ecosystem on top of the cluster creation. You may use this cluster for any sort of analytics task with out any defined time frame of existence and condition to terminate. This type of cluster has to be terminated manually.
- (b) **Step Execution**: This type of cluster is created to perform very specific tasks which are defined in a sequence of steps. As soon as the cluster finishes its pre-defined tasks, the cluster terminates automatically. We will discuss this mode of cluster creation later in detail.

For this lab exercise select cluster as the launch mode.

Software configuration

<div style="border: 1px solid #ccc; padding: 5px;"> <p>Software configuration</p> <p>Release <input type="text" value="emr-5.33.0"/></p> <p>Applications <input checked="" type="radio"/> Core Hadoop: Hadoop 2.10.1, Hive 2.3.7, Hue 4.9.0, Mahout 0.13.0, Pig 0.17.0, and Tez 0.9.2 <input type="radio"/> HBase: HBase 1.4.13, Hadoop 2.10.1, Hive 2.3.7, Hue 4.9.0, Phoenix 4.14.3, and ZooKeeper 3.4.14 <input type="radio"/> Presto: Presto 0.245.1 with Hadoop 2.10.1 HDFS and Hive 2.3.7 Metastore <input type="radio"/> Spark: Spark 2.4.7 on Hadoop 2.10.1 YARN and Zeppelin 0.9.0</p> <p><input type="checkbox"/> Use AWS Glue Data Catalog for table metadata</p> </div>	<div style="border: 1px solid #ccc; padding: 5px;"> <p>Software configuration</p> <p>Release <input type="text" value="emr-6.2.0"/></p> <p>Applications <input checked="" type="radio"/> Core Hadoop: Hadoop 3.2.1 with Hive 3.1.2, Hue 4.8.0, Pig 0.17.0 and Tez 0.9.2 <input type="radio"/> HBase: HBase 2.2.6 with Hadoop 3.2.1, Hive 3.1.2, Hue 4.8.0, Phoenix 5.0.0, and ZooKeeper 3.4.14 <input type="radio"/> Presto: Presto 0.238.3 with Hadoop 3.2.1 HDFS and Hive 3.1.2 Metastore <input type="radio"/> PrestoSQL: PrestoSQL 343 with Hadoop 3.2.1 HDFS and Hive 3.1.2 Metastore <input type="radio"/> Spark: Spark 3.0.1 on Hadoop 3.2.1 YARN with and Zeppelin 0.9.0-preview1</p> <p><input type="checkbox"/> Use AWS Glue Data Catalog for table metadata</p> </div>
(a) default	(b) custom

Figure 6.5: EMR cluster creation - Software Configuration

Since we are choosing cluster launch mode, the software configuration section of this form will give us options to choose from different releases of EMR.

In simple terms EMR releases are similar to pre-packaged Apache hadoop clusters which come with pre-installed/configured softwares of the hadoop eco system. This is similar to choosing from various apache hadoop distributions when you are not

working in a cloud environment. Such distributions may include cloudera, hortonworks and etc (we have discussed these in initial lectures).

You may have already noticed that in Fig 6.5(a) & 6.5(b) show the screenshots when the automatically rendered EMR release was left as it is and when the latest EMR release was selected instead, respectively. The list of optional hadoop eco system software that you may add in your cluster change with the selection of EMR release.

For this lab exercise select EMR-6.20, which is the latest release and select the core hadoop option.

Hardware configuration

Hardware configuration

Instance type The selected instance type adds 64 GiB of GP2 EBS storage per instance by default. [Learn more](#)

Number of instances (1 master and 2 core nodes)

Cluster scaling scale cluster nodes based on workload

Figure 6.6: EMR cluster creation - Hardware Configuration

This section deals with picking the hardware back end that is fit for purpose in a given context for cluster creation. You may find it useful to revise the concepts about AWS EC2 compute and storage and revisit the type of compute and storage options available.

It would then be useful if you could skim through the weblink on [Storage instances for EMR](#). Once you are back after revisiting all the concepts, you would be ready to configure this section of the cluster creation form.

For this lab exercise leave the default options as they are in this section.

Security and access

Security and access

EC2 key pair

Permissions Default Custom
Use default IAM roles. If roles are not present, they will be automatically created for you with managed policies for automatic policy updates.

EMR role [EMR_DefaultRole](#)

EC2 instance profile [EMR_EC2_DefaultRole](#)

Figure 6.7: EMR cluster creation - Security and Access

Before we can go any further with this section, it is advised that you should revise your concepts on SSH and accessing remote computers from a local machine via this protocol.

In order to keep the EC2 machines secure, AWS uses EC2 key pairs which may then be used over the SSH protocol as public and private keys. It is recommended that you read through [AWS's guide on EC2](#) before following the rest of this lab manual.

Setup key pair

Create an Amazon EC2 Key Pair and PEM File

Amazon EMR uses an Amazon Elastic Compute Cloud (Amazon EC2) key pair to ensure that you alone have access to the instances that you launch. The PEM file associated with this key pair is required to ssh directly to the master node of the cluster.

To create an Amazon EC2 key pair:

1. Go to the [Amazon EC2 console](#)
2. In the Navigation pane, click Key Pairs
3. On the Key Pairs page, click Create Key Pair
4. In the Create Key Pair dialog box, enter a name for your key pair, such as, mykeypair
5. Click Create
6. Save the resulting PEM file in a safe location

Modify Your PEM File

Amazon Elastic MapReduce (Amazon EMR) enables you to work interactively with your cluster, allowing you to test cluster steps or troubleshoot your cluster environment. You use your PEM file to authenticate to the master node. The PEM file requires a modification based on the tool you use that supports your operating system.

To modify your credentials file:

Windows Mac / Linux

Set the permissions on the PEM file or your Amazon EC2 key pair. For example, if you saved the file as mykeypair.pem, the command looks like the following:

```
chmod og-rwx mykeypair.pem
```

Your credentials file is now modified to allow you to log in directly to the master node of your running cluster.

Windows Mac / Linux

1. Download PuTTYgen.exe to your computer from:
<http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html>

2. Launch PuTTYgen
3. Click Load
4. Select the PEM file you created earlier
5. Click Open
6. Click OK on the PuTTYgen Notice telling you the key was successfully imported
7. Enter a pass phrase in the Key passphrase field
8. Click Save private key to save the key in the PPK format
9. Enter a name for your PuTTY private key, such as, mykeypair.ppk
10. Click Save
11. Exit the PuTTYgen application

Your credentials file is now modified to allow you to log in directly to the master node of your running cluster.

(a) iOS

(b) Windows

Figure 6.8: EMR cluster creation - Creating and setting up EC2 keypairs

Creating EC2 Keypairs :

Since we have not created any EC2 pair in this module yet, we will have to create one. Fig 6.8 provides screenshots of the instructions that are available in the link adjacent to the EC2 key pair drop down menu in this section of the cluster creation form. Both the screenshots share instructions on how to setup an EC2 keypair for

windows and iOS platform. The instructions for iOS are also applicable on Linux distributions. Please note that you will have to access the EC2 dashboard via AWS console in MS4S21 AWS classroom.

- Once you are in the EC2 dashboard, the dashboard web page to keypairs can be found either in the EC2 dashboard stats section or the navigation bar on the left of the EC2 dashboard web page. Both of these locations are encircled with thick ink in Fig. 6.9. Open the Keypairs dashboard.

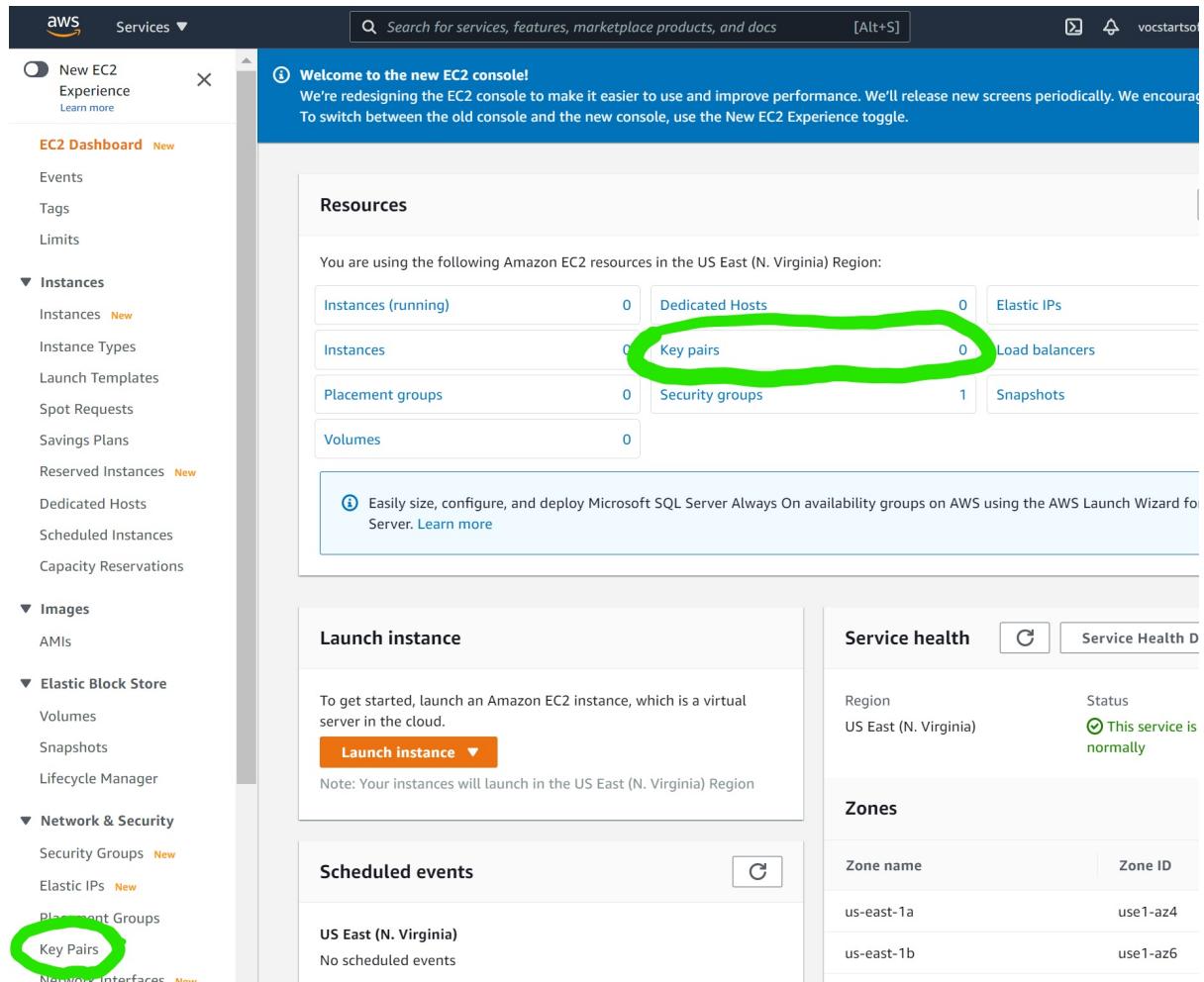


Figure 6.9: AWS Keypair Dashboard

- Click on the orange button titled as ‘create key pair’ on the top right corner of keypair dashboard as shown in Fig 6.10



Figure 6.10: AWS Keypair dashboard

- The create keypair button will take you to the keypair creation form. Fill the

fields and select options in this form as shown in Fig. 6.11. Note that an optional field ‘Add Tag’ has also been populated. This would help you recall which keypairs were created for which lab exercise or project. It is recommended that you select the .pem format as it can be converted to putty format easily while allowing you the option to use .pem or putty files of the keypair on any OS.

The screenshot shows the 'Create key pair' page in the AWS EC2 console. At the top, the navigation path is 'EC2 > Key pairs > Create key pair'. The main title is 'Create key pair'. A sub-section titled 'Key pair' defines it as 'A key pair, consisting of a private key and a public key, is a set of security credentials that you use to prove your identity when connecting to an instance.' Below this, the 'Name' field contains 'EC2keypair_initial-MS4S21'. A note says 'The name can include up to 255 ASCII characters. It can't include leading or trailing spaces.' Under 'File format', 'pem' is selected ('For use with OpenSSH'). 'ppk' is also listed ('For use with PuTTY'). The 'Tags (Optional)' section lists two tags: 'EC2' with value 'for initial cluster' and 'Classroom' with value 'MS4S21'. Both tags have 'Remove' buttons. An 'Add tag' button is available. A note at the bottom says 'You can add 48 more tags.' At the bottom right are 'Cancel' and 'Create key pair' buttons.

Figure 6.11: AWS Keypair dashboard

4. Don’t forget to hit the ‘create key pair’ button once you have filled the form according to the instructions provided here.
5. You will notice that after hitting the button, you have been navigated back to the keypair dashboard as shown in Fig. 6.12 and the .pem keypair file is also downloaded. Ensure that you have saved the file to a safe location.

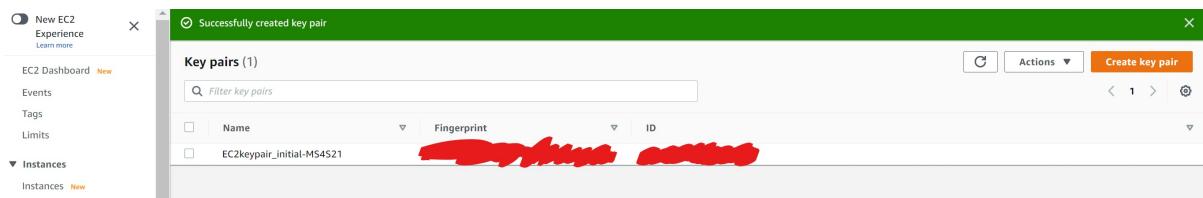


Figure 6.12: AWS Keypair dashboard

Modifying the EC2 Keypair :

Since we won't be using the EC2 keypair in this tutorial, the modification of the keypair won't be demonstrated unlike creation. However, if you would like you can modify the keypair following the instructions provided in Fig. 6.8 so that you have it prepared for later tutorials and courseworks.

1. **EC2 keypair** - You will notice that after the creation of keypairs, the drop down menu in cluster creation form is populated with this keypair as an option as well.

For this lab tutorial select they keypair that you have just created as shown in Fig. 6.13.

Security and access

This screenshot shows the 'Security and access' configuration for an EMR cluster. It includes fields for 'EC2 key pair' (set to 'EC2keypair_initial-MS4S21'), 'Permissions' (set to 'Default'), and 'EMR role' (set to 'EMR_DefaultRole'). Below these, there is a note about default IAM roles and a link to learn how to create an EC2 key pair. There is also a link to view the 'EMR_EC2_DefaultRole' policy.

Figure 6.13: EMR Cluster creation - filled Security and Access

2. **Permissions** - leave as default.

3. **EMR Default Role** - Allows EMR to call other AWS services on your behalf as required. Open the link provide beside the field. You will be navigated to this EMR role's page under IAM dashboard as shown in Fig. 6.14.

Identity and Access Management (IAM)

Roles > EMR_EC2_DefaultRole

Summary

Policy has been detached from the role EMR_EC2_DefaultRole

Role ARN	arn:aws:iam::035797753054:role/EMR_EC2_DefaultRole
Role description	Edit
Instance Profile ARNs	arn:aws:iam::035797753054:instance-profile/EMR_EC2_DefaultRole
Path	/
Creation time	2021-02-21 11:13 UTC+0100
Last activity	Not accessed in the tracking period
Maximum session duration	1 hour Edit

Permissions **Trust relationships** **Tags** **Access Advisor** **Revoke sessions**

▼ Permissions policies (1 policy applied)

Attach policies **Add inline policy**

Policy name	Policy type
AmazonElasticMapReduceforEC2Role	AWS managed policy

▶ Permissions boundary (not set)

► You need permissions

Figure 6.14: Roles webpage under IAM dashboard

You can click on the policy and find out more about the default IAM role for EMR. If you click on the policy you will be navigated to the page where all permissions with this role will be rendered on the policy summary webpage as shown in Fig. 6.15

Identity and Access Management (IAM)

Policies > AmazonElasticMapReduceforEC2Role

Summary

Policy ARN	arn:aws:iam::aws:policy/service-role/AmazonElasticMapReduceforEC2Role
Description	Default policy for the Amazon Elastic MapReduce for EC2 service role.

Permissions **Policy usage** **Policy versions** **Access Advisor**

Policy summary **{ JSON**

Q Filter

Service	Access level	Resource	Request condition
Allow (11 of 278 services) Show remaining 267			
CloudWatch	Full access	All resources	None
DynamoDB	Full access	All resources	None
EC2	Limited: List, Read	All resources	None
EMR	Limited: List, Read	All resources	None
Glue	Limited: Read, Write	All resources	None
Kinesis	Limited: Read, Write	All resources	None
RDS	Full: List	All resources	None
S3	Full access	All resources	None
SimpleDB	Full access	All resources	None
SNS	Full access	All resources	None
SQS	Full access	All resources	None

Figure 6.15: Policies for default IAM role for EMR

4. **EMR EC2 DefaultRole:** You can follow similar steps to EMR Default Role to find out more about the IAM policies and the default role here.
5. Finally don't forget to hit the create cluster button! The cluster will be discussed in the next lab tutorial manual.