

EDUREKA

Apache Hadoop 2.0 Installation and Single Node Cluster Configuration on Ubuntu

A guide to install and setup Single-Node
Apache Hadoop 2.0 Cluster

edureka!

11/12/2013

APACHE HADOOP 2.0 INSTALLATION AND SINGLE NODE CLUSTER CONFIGURATION

A guide to install and setup Single-Node Apache Hadoop 2.0 Cluster

Table of Contents

Introduction	2
1. Setting up the Ubuntu Server	3
1.1 Creating an Ubuntu VMPlayer instance.....	3
1.1.1 Download the VMware image	3
1.1.2 Open the image file.....	3
1.1.3 Play the Virtual Machine.....	5
1.1.4 Update the OS packages and their dependencies	7
1.1.5 Install the Java for Hadoop 2.2.0	7
1.2 Download the Apache Hadoop 2.0 binaries	7
1.2.1 Download the Hadoop package.....	7
2. Configure the Apache Hadoop 2.0 Single Node Server	9
2.1 Update the Configuration files.....	9
2.1.1 Update “.bashrc” file for user ‘ubuntu’	9
2.2 Setup the Hadoop Cluster	11
2.2.1 Configure JAVA_HOME	11
2.2.2 Create namenode and datanode directory	12
2.2.3 Configure the Default Filesystem.....	12
2.2.4 Configure the HDFS.....	13
2.2.5 Configure YARN framework	14
2.2.6 Configure MapReduce framework.....	15
2.2.6 Start the DFS services.....	16
2.2.7 Perform the Health Check.....	18

edureka!

Introduction

This setup and configuration document is a guide to setup a Single-Node Apache Hadoop 2.0 cluster on an Ubuntu virtual machine on your PC. If you are new to both Ubuntu and Hadoop, this guide comes handy to quickly setup a Single-Node Apache Hadoop 2.0 Cluster on Ubuntu and start your Big Data and Hadoop learning journey.

The guide describes the whole process in two parts:

[Section 1: Setting up the Ubuntu OS for Hadoop 2.0](#)

This section describes step by step guide to download, configure an Ubuntu Virtual Machine image in VMPlayer, and provides steps to install pre-requisites for Hadoop Installation on Ubuntu.

[Section 2: Installing Apache Hadoop 2.0 and Setting up the Single Node Cluster](#)

This section explains primary Hadoop 2.0 configuration files, Single-Node cluster configuration and Hadoop daemons start and stop process in detail.

Note

The configuration described here is intended for learning purposes only.

1. Setting up the Ubuntu Server

This section describes the steps to download and create an Ubuntu image on VMPlayer.

1.1 Creating an Ubuntu VMPlayer instance

The first step is to download an Ubuntu image and create an Ubuntu VMPlayer instance.

1.1.1 Download the VMware image

Access the following link and download the 12.0.4 Ubuntu image:

<http://www.trafficool.net/vmware/ubuntu1204t.html>

1.1.2 Open the image file

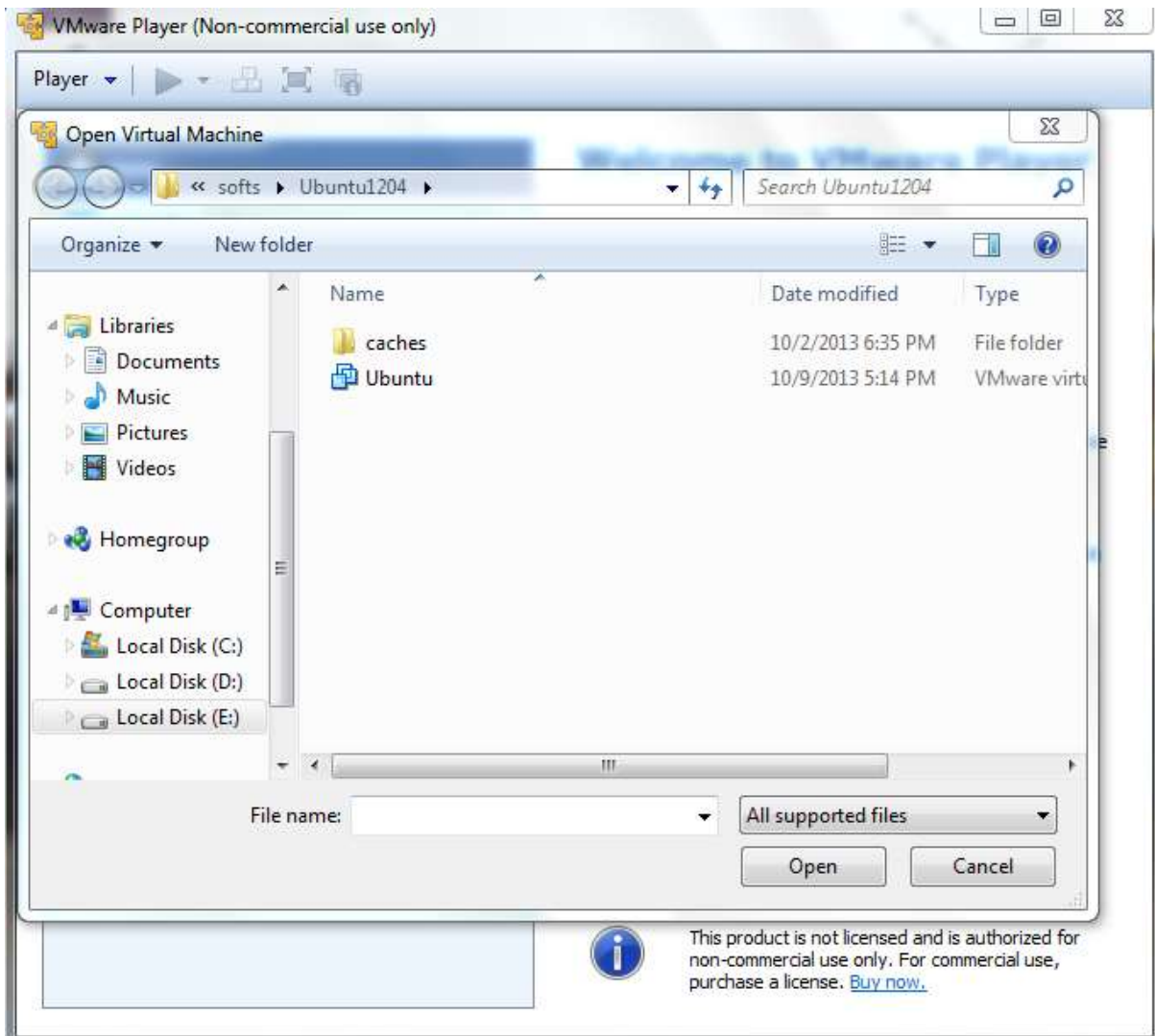
Extract the Ubuntu VM image and Open it in VMware Player.

Click open virtual machine and select path where you have extracted the image.

Select the **‘.vmx’** file and click **‘ok’**.

edureka!

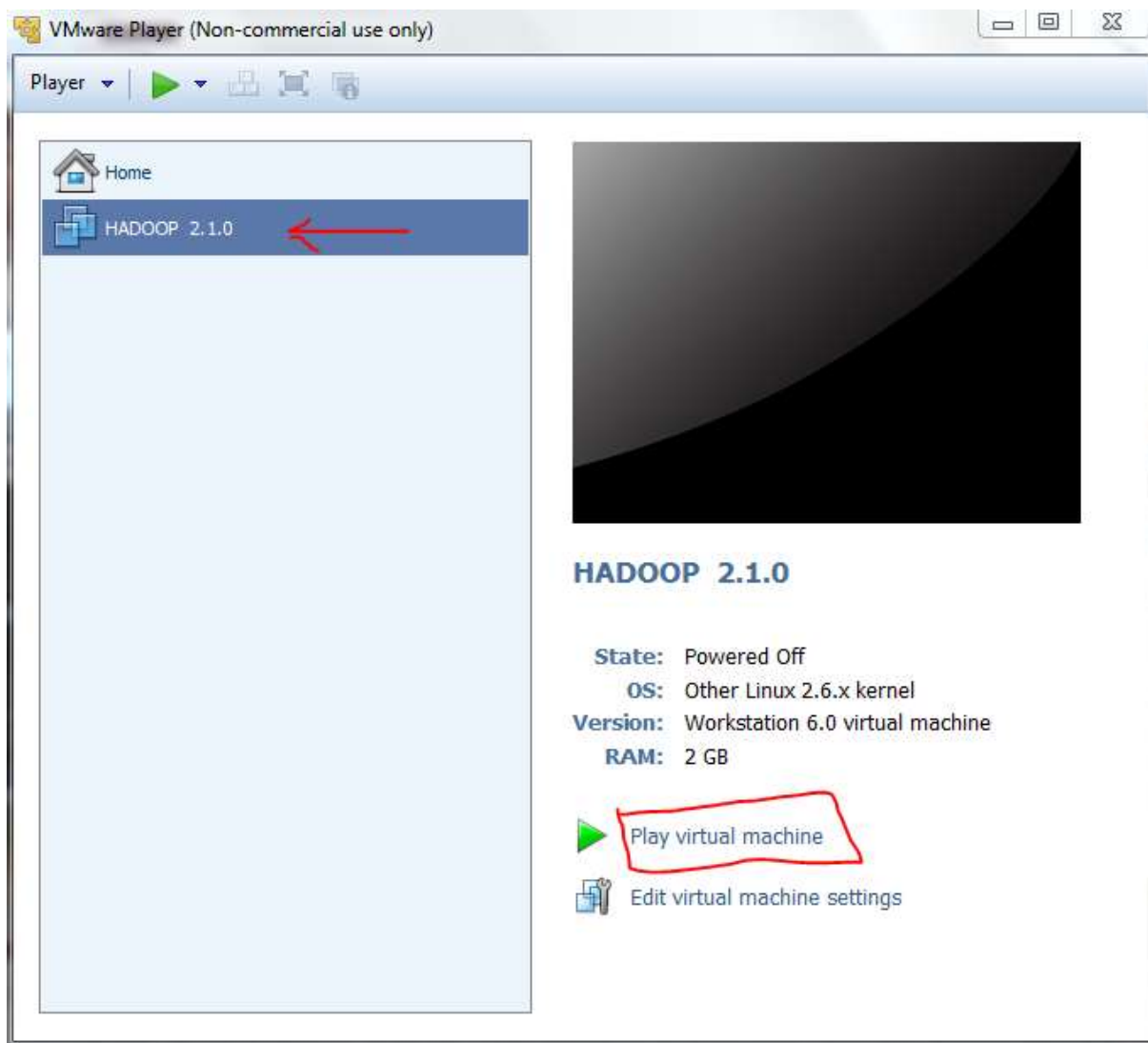
FIGURE 1-1 OPEN THE VM IMAGE



1.1.3 Play the Virtual Machine

You would see the below screen in VMware Player after the VM image creation completes.

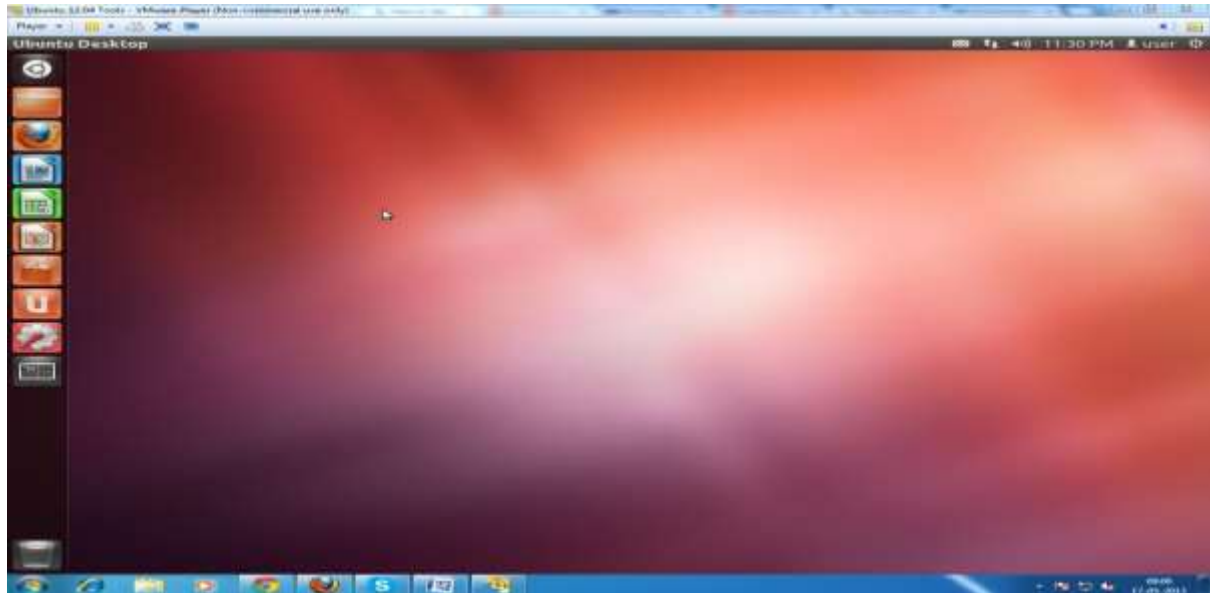
FIGURE 1-2 PLAY THE VIRTUAL MACHINE



Double click on the link.

You will get the home screen with the following image.

FIGURE 1-3 UBUNTU HOME SCREEN



The user details for the Virtual instance is:

Username : user

Password : password

Open the terminal to access the file system.

FIGURE 1-4 OPEN A TERMINAL



1.1.4 Update the OS packages and their dependencies

The first task is to run **'apt-get update'** to download the package lists from the repositories and "update" them to get information on the newest versions of packages and their dependencies.

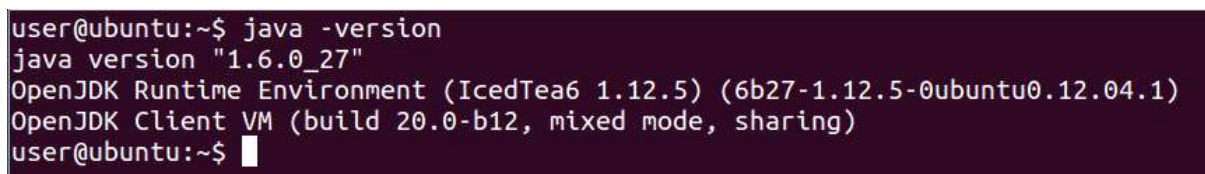
```
$sudo apt-get update
```

1.1.5 Install the Java for Hadoop 2.2.0

Use apt-get to install the JDK 6 on the server.

```
$sudo apt-get install openjdk-6-jdk
```

FIGURE 1-5 INSTALL JDK



1.2 Download the Apache Hadoop 2.0 binaries

1.2.1 Download the Hadoop package

Download the binaries to your home directory. Use the default user 'user' for the installation.

In Live production instances a dedicated Hadoop user account for running Hadoop is used. Though, it's not mandatory to use a dedicated Hadoop user account but is recommended because this helps to separate the Hadoop installation from other software applications and user accounts running on the same machine (separating for security, permissions, backups, etc.).


```
$wgethttp://apache.mirrors.lucidnetworks.net/hadoop/common/stable2/hadoop-2.2.0.tar.gz
```

FIGURE 1-6. DOWNLOAD HADOOP 2.2.0

```
user@ubuntu:~$ wget http://apache.mirrors.lucidnetworks.net/hadoop/common/stable2/hadoop-2.2.0.tar.gz
--2013-11-11 08:20:19-- http://apache.mirrors.lucidnetworks.net/hadoop/common/stable2/hadoop-2.2.0.tar.gz
Resolving apache.mirrors.lucidnetworks.net (apache.mirrors.lucidnetworks.net)...
 108.166.161.136
Connecting to apache.mirrors.lucidnetworks.net (apache.mirrors.lucidnetworks.net)|108.166.161.136|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 109229073 (104M) [application/x-gzip]
Saving to: `hadoop-2.2.0.tar.gz'

0% [          ] 805,868    177K/s  eta 10m 51s
```

```
user@ubuntu:~$ ls
Desktop    Downloads    hadoop-2.2.0.tar.gz  Pictures  Templates
Documents  examples.desktop  Music                Public    Videos
user@ubuntu:~$
```

Unzip the files and review the package content and configuration files.

```
$star -xvf hadoop-2.2.0.tar.gz
```

FIGURE 1-7. HADOOP PACKAGE CONTENT

```
user@ubuntu:~$ ls
Desktop    Downloads    hadoop-2.2.0    Music    Public    Videos
Documents  examples.desktop  hadoop-2.2.0.tar.gz  Pictures  Templates
user@ubuntu:~$
```

```

user@ubuntu:~$ cd hadoop-2.2.0/
user@ubuntu:~/hadoop-2.2.0$ ls
bin  include  libexec  NOTICE.txt  sbin
etc  lib      LICENSE.txt  README.txt  share
user@ubuntu:~/hadoop-2.2.0$ cd etc
user@ubuntu:~/hadoop-2.2.0/etc$ ls
hadoop
user@ubuntu:~/hadoop-2.2.0/etc$ cd hadoop/
user@ubuntu:~/hadoop-2.2.0/etc/hadoop$ ls
capacity-scheduler.xml  nttps-site.xml
configuration.xsl      log4j.properties
container-executor.cfg  mapred-env.cmd
core-site.xml           mapred-env.sh
hadoop-env.cmd         mapred-queues.xml.template
hadoop-env.sh          mapred-site.xml.template
hadoop-metrics2.properties  slaves
hadoop-metrics.properties  ssl-client.xml.example
hadoop-policy.xml         ssl-server.xml.example
hdfs-site.xml            yarn-env.cmd
httpfs-env.sh            yarn-env.sh
httpfs-log4j.properties  yarn-site.xml
httpfs-signature.secret
user@ubuntu:~/hadoop-2.2.0/etc/hadoop$ █

```

Review the Hadoop configurations files.

After creating and configuring your virtual servers, the Ubuntu instance is now ready to start installation and configuration of Apache Hadoop 2.0 Single Node Cluster. This section describes the steps in details to install Apache Hadoop 2.0 and configure a Single-Node Apache Hadoop cluster.

2. Configure the Apache Hadoop 2.0 Single Node Server

This section explains the steps to configure the Single Node Apache Hadoop 2.0 Server on Ubuntu.

2.1 Update the Configuration files

2.1.1 Update “.bashrc” file for user ‘ubuntu’.

Move to ‘user’ \$HOME directory and edit ‘.bashrc’ file.

FIGURE 2-1 FILE '.BACHRC' LOCATION

```
user@ubuntu:~$ cd
user@ubuntu:~$ ls -al .b*
-rw----- 1 user user  29 Nov 11 08:01 .bash_history
-rw-r--r-- 1 user user 220 Apr 28 2012 .bash_logout
-rw-r--r-- 1 user user 3486 Apr 28 2012 .bashrc
user@ubuntu:~$
```

Update the **'.bashrc'** file to add important Apache Hadoop environment variables for user.

- a) Change directory to home.

```
$ cd
```

- b) Edit the file

```
$ vi .bashrc
```

```
-----Set Hadoop environment Variables - Begin-----
# Set Hadoop-related environment variables
export HADOOP_HOME=$HOME/hadoop-2.2.0
export HADOOP_CONF_DIR=$HOME/hadoop-2.2.0/etc/hadoop
export HADOOP_MAPRED_HOME=$HOME/hadoop-2.2.0
export HADOOP_COMMON_HOME=$HOME/hadoop-2.2.0
export HADOOP_HDFS_HOME=$HOME/hadoop-2.2.0
export YARN_HOME=$HOME/hadoop-2.2.0

# Set JAVA_HOME (we will also configure JAVA_HOME for Hadoop
execution later on)
export JAVA_HOME=/usr/lib/jvm/java-6-openjdk-amd6
```

```
# Add Hadoop bin/ directory to PATH
export PATH=$PATH:$HOME/hadoop-2.2.0/bin
```

-----Set Hadoop environment Variables – End -----

FIGURE 2-2 EDIT .BASHRC

```
# Set Hadoop-related environment variables
export HADOOP_HOME=$HOME/hadoop-2.2.0
export HADOOP_MAPRED_HOME=$HOME/hadoop-2.2.0
export HADOOP_COMMON_HOME=$HOME/hadoop-2.2.0
export HADOOP_HDFS_HOME=$HOME/hadoop-2.2.0
export YARN_HOME=$HOME/hadoop-2.2.0
export HADOOP_CONF_DIR=$HOME/hadoop-2.2.0/etc/hadoop

# Set JAVA_HOME (we will also configure JAVA_HOME for Hadoop execution later on)
export JAVA_HOME=/usr/lib/jvm/java-6-openjdk-amd64

# Add Hadoop bin/ directory to PATH
export PATH=$PATH:$HOME/hadoop-2.2.0/bin
```

- c) Source the .bashrc file to set the hadoop environment variables without having to invoke a new shell:

```
$. ~/.bashrc
```

Execute the all the steps of this section on all the remaining cluster servers.

2.2 Setup the Hadoop Cluster

This section describes the detail steps needed for setting up the Hadoop Cluster and configuring the core Hadoop configuration files.

2.2.1 Configure JAVA_HOME

Configure JAVA_HOME in '**hadoop-env.sh**'. This file specifies environment variables that affect the JDK used by Apache Hadoop 2.0 daemons started by the Hadoop start-up scripts:

```
$cd $HADOOP_CONF_DIR
```

```
$vi hadoop-env.sh
```

Update the JAVA_HOME to:

```
export JAVA_HOME=/usr/lib/jvm/java-6-openjdk-amd64
```

FIGURE 2-3 JAVA HOME SETUP

```
# The java implementation to use.
#export JAVA_HOME=${JAVA_HOME}
export JAVA_HOME=/usr/lib/jvm/java-6-openjdk-1.386
```

2.2.2 Create NameNode and DataNode directory

Create DataNode and NameNode directories to store HDFS data.

```
$ mkdir -p $HOME/hadoop2_data/hdfs/namenode
```

```
$ mkdir -p $HOME/hadoop2_data/hdfs/datanode
```

2.2.3 Configure the Default File system

The 'core-site.xml' file contains the configuration settings for Apache Hadoop Core such as I/O settings that are common to HDFS, YARN and MapReduce. Configure default files-system (Parameter: fs.default.name) used by clients in **core-site.xml**

```
$cd $HADOOP_CONF_DIR
```

```
$vi core-site.xml
```

Add the following line in between the configuration tag:

```
<configuration>
```

```
<property>
```

```
  <name>fs.default.name</name>
```

```
  <value>hdfs://localhost:9000</value>
```

```
</property>
```

```
</configuration>
```

FIGURE 2-4 CONFIGURE THE DEFAULT FILE SYSTEM

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

<configuration>
<property>
  <name>fs.default.name</name>
  <value>hdfs://localhost:9000</value>
</property>
</configuration>
```

Where *hostname* and *port* are the machine and port on which Name Node daemon runs and listens. It also informs the Name Node as to which IP and port it should bind. The commonly used port is 9000 and you can also specify IP address rather than hostname.

Note

For the simplicity of understanding the cluster setup, we have updated changed only necessary parameters to start a cluster. You can research more on Apache Hadoop 2.0 page and experiment the configuration for different features.

2.2.4 Configure the HDFS

This file contains the configuration settings for HDFS daemons; the Name Node and the data nodes.

Configure **hdfs-site.xml** and specify default block replication, and NameNode and DataNode directories for HDFS. The actual number of replications can be specified when the file is created. The default is used if replication is not specified in create time.

```
$cd $HADOOP_CONF_DIR
```

```
$vi hdfs-site.xml
```

Add the following line in between the configuration tag:

```
<configuration>
<property>
  <name>dfs.replication</name>
  <value>1</value>
```

</property>

<property>

<name>dfs.namenode.name.dir</name>

<value>file:/home/user/hadoop-2.2.0/hadoop2_data/hdfs/namenode</value>

</property>

<property>

<name>dfs.datanode.data.dir</name>

<value>file:/home/user/hadoop-2.2.0/hadoop2_data/hdfs/datanode</value>

</property>

</configuration>

FIGURE 2-5 CONFIGURE THE DEFAULT FILESYSTEM

```
-->
<!-- Put site-specific property overrides in this file. -->
<configuration>
  <property>
    <name>dfs.replication</name>
    <value>1</value>
  </property>
  <property>
    <name>dfs.namenode.name.dir</name>
    <value>file:/home/user/hadoop-2.2.0/hadoop2_data/hdfs/namenode</value>
  </property>
  <property>
    <name>dfs.datanode.data.dir</name>
    <value>file:/home/user/hadoop-2.2.0/hadoop2_data/hdfs/datanode</value>
  </property>
</configuration>
```

2.2.5 Configure YARN framework

This file contains the configuration settings for YARN; the NodeManager.

```
$cd $HADOOP_CONF_DIR
```

```
$vi yarn-site.xml
```

Add the following line in between the configuration tag:

```

<configuration>
  <property>
    <name>yarn.nodemanager.aux-services</name>
    <value>mapreduce_shuffle</value>
  </property>
  <property>
    <name>yarn.nodemanager.aux-
services.mapreduce.shuffle.class</name>
    <value>org.apache.hadoop.mapred.ShuffleHandler</value>
  </property>
</configuration>

```

FIGURE 2-6 CONFIGURE THE DEFAULT FILESYSTEM

```

-->
<configuration>
<!-- Site specific YARN configuration properties -->
<property>
  <name>yarn.nodemanager.aux-services</name>
  <value>mapreduce_shuffle</value>
</property>
<property>
  <name>yarn.nodemanager.aux-services.mapreduce.shuffle.class</name>
  <value>org.apache.hadoop.mapred.ShuffleHandler</value>
</property>
</configuration>

```

2.2.6 Configure MapReduce framework

This file contains the configuration settings for MapReduce.

Configure `mapred-site.xml` and specify framework details.

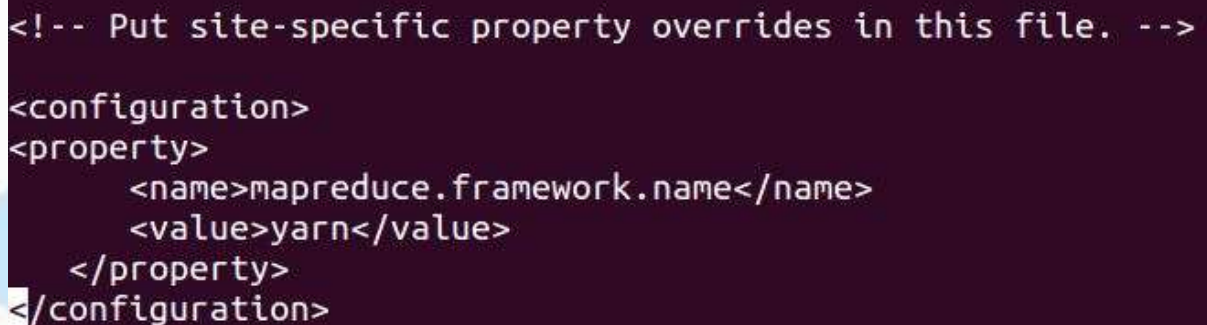
```
$cd $HADOOP_CONF_DIR
```

```
$vi mapred-site.xml
```

Add the following line in between the configuration tag:


```
<configuration>  
<property>  
    <name>mapreduce.framework.name</name>  
    <value>yarn</value>  
</property>  
</configuration>
```

FIGURE 2-7 CONFIGURE THE JOBTRACKER DETAILS



```
<!-- Put site-specific property overrides in this file. -->  
<configuration>  
<property>  
    <name>mapreduce.framework.name</name>  
    <value>yarn</value>  
</property>  
</configuration>
```

2.2.6 Start the DFS services

The first step in starting up your Hadoop installation is formatting the Hadoop file-system, which is implemented on top of the local file-systems of your cluster. This is required on the first time Hadoop installation. Do not format a running Hadoop file-system, this will cause all your data to be erased.

To format the file-system, run the command:

```
$hadoop namenode -format
```

You are now all set to start the HDFS services i.e. Name Node, Resource Manager, Node Manager and Data Nodes on your Apache Hadoop Cluster.

FIGURE 2-8 START THE SERVICES

```
user@ubuntu:~/hadoop-2.2.0/sbin$ ./hadoop-daemon.sh start namenode
starting namenode, logging to /home/user/hadoop-2.2.0/logs/hadoop-user-namenode-ubuntu.out
user@ubuntu:~/hadoop-2.2.0/sbin$ jps
8410 NameNode
8468 Jps
user@ubuntu:~/hadoop-2.2.0/sbin$ ./hadoop-daemon.sh start datanode
starting datanode, logging to /home/user/hadoop-2.2.0/logs/hadoop-user-datanode-ubuntu.out
user@ubuntu:~/hadoop-2.2.0/sbin$ jps
8410 NameNode
8525 Jps
8488 DataNode
user@ubuntu:~/hadoop-2.2.0/sbin$
```

Start the YARN daemons i.e. Resource Manager and Node Manager. Cross check the service start-up using JPS (Java Process Monitoring Tool).

FIGURE 2-9 START THE YARN DAEMONS

```
user@ubuntu:~/hadoop-2.2.0/sbin$ ./yarn-daemon.sh start resourcemanager
starting resourcemanager, logging to /home/user/hadoop-2.2.0/logs/yarn-user-resourcemanager-ubuntu.out
user@ubuntu:~/hadoop-2.2.0/sbin$ jps
8410 NameNode
8620 Jps
8575 ResourceManager
8488 DataNode
user@ubuntu:~/hadoop-2.2.0/sbin$ ./yarn-daemon.sh start nodemanager
starting nodemanager, logging to /home/user/hadoop-2.2.0/logs/yarn-user-nodemanager-ubuntu.out
user@ubuntu:~/hadoop-2.2.0/sbin$ jps
8410 NameNode
8575 ResourceManager
10960 Jps
8488 DataNode
10928 NodeManager
8880 JobHistoryServer
user@ubuntu:~/hadoop-2.2.0/sbin$
```

Start the History server.

FIGURE 2-10 START THE HISTORY SERVER

```
user@ubuntu:~/hadoop-2.2.0/sbin$ ./mr-jobhistory-daemon.sh start historyserver
starting historyserver, logging to /home/user/hadoop-2.2.0/logs/mapred-user-historyserver-ubuntu.out
user@ubuntu:~/hadoop-2.2.0/sbin$ jps
8410 NameNode
8575 ResourceManager
8488 DataNode
8880 JobHistoryServer
8916 Jps
user@ubuntu:~/hadoop-2.2.0/sbin$
```

2.2.7 Perform the Health Check

- a) Check the NameNode status:

<http://localhost:50070/dfshealth.jsp>

FIGURE 2-11 NAME NODE STATUS

The screenshot shows a web browser window with the address bar containing 'localhost:50070/dfshealth.jsp'. The page title is 'NameNode 'localhost:9000' (active)'. Below the title is a table with the following data:

Started:	Mon Nov 11 09:46:36 EST 2013
Version:	2.2.0, 1529768
Compiled:	2013-10-07T06:28Z by hortonmu from branch-2.2.0
Cluster ID:	CID-42557cfe-be4c-4798-b9ae-f9d7fc2fb778
Block Pool ID:	BP-1143686600-127.0.1.1-1384181141421

Below the table are two links: 'Browse the filesystem' and 'NameNode Logs'. The 'Cluster Summary' section indicates that security is OFF, there are 7 files and directories, and 0 blocks. It also shows memory usage statistics: Heap Memory used 40.42 MB (59% of 67.91 MB committed), and Non Heap Memory used 32.54 MB (81% of 40.03 MB committed). A table at the bottom of the summary shows:

Configured Capacity	:	39.28 GB
DFS Used	:	24 KB
Non DFS Used	:	5.00 GB
DFS Remaining	:	34.28 GB

b) JobHistory status:

<http://localhost:19888/jobhistory.jsp>

FIGURE 2-12 JOBHISTORY STATUS

The screenshot shows the Hadoop JobHistory web interface. The browser address bar displays "localhost:19888/jobhistory". The page title is "JobHistory" and the user is logged in as "dr.who". On the left, there is a navigation menu with "Tools" expanded, showing links for "Configuration", "Local logs", "Server stacks", and "Server metrics". The main content area is titled "Retired Jobs" and features a search bar and a table. The table has columns for Start Time, Finish Time, Job ID, Name, User, Queue, State, Maps Total, Maps Completed, Reduces Total, and Reduces Completed. Below the table, it states "No data available in table". At the bottom of the table area, it says "Showing 0 to 0 of 0 entries" and includes navigation links for "First", "Previous", "Next", and "Last".

Start Time	Finish Time	Job ID	Name	User	Queue	State	Maps Total	Maps Completed	Reduces Total	Reduces Completed
No data available in table										

Showing 0 to 0 of 0 entries

edureka!