

HADOOP

HDFS and Pseudo Cluster

Prerequisites

- JAVA
 - OOPs concepts
 - Serialization
 - Data Structures (Hash Map, Lists)
 - FILE I/O
- UNIX Commands (mv, cp, ls etc, mkdir, ps, vi)

Development Environment

- Install jdk 1.6, jre 6, eclipse

Introduction

What is Big Data

- Typically we work on excel sheet, ppt, word docs, code files. They are of the order 1-2Mb. Even a movie is just 1–2 Gb size.
- The BIG DATA we want to deal with, is of the order of Petabytes. 10^{12} times size of ordinary files.

Confidential

What Happens in An Internet Minute?



Confidential

Where is this Data

- This data is generated from multiple sources. The data that goes in the logs of google, facebook, linkedin, yahoo servers is of billion users of all around the world.
- What are users accessing, how long the user remains in site. All the meta data sites visited, friend's list, status. Torrent downloads In Every 5 minutes granularity google gets Petabytes information in it's server logs. Same goes for facebook, yahoo, AT&T, Airtel.

Confidential

Why is this Data Needed

- In networks companies like airtel in India, AT&T in US monitor routers logs where information of subscriber phone id, call id is recorded to find top subscribers and install routers where traffic is more etc.
- Google, Amazon, Ebay they get logs so that ads and products can be recommended to customers.

Who can Use this Data

- Any organization / entity that has webserver or an information logging utility can utilize it's logs to find out information relevant for it's business, and take insights into data to increase revenue and decrease it's costs.

How do the Logs Look

```

192.168.156.95 - PuTTY
ERROR | 20/11/2013 3:01:40:000 | session | user | com.guavus.common.login.models.presentation.LoginPM | LoginModule_Login | 2CC20E64-A91E-6C77-B5C7-74D9CB27F696 | | Login Event failure for LoginModule_Login due to null
ERROR | 20/11/2013 3:01:40:000 | session | user | com.guavus.common.application.models.presentation.MainPM | LoginModule_Login | 2CC20E64-A91E-6C77-B5C7-74D9CB27F696 | | Login Event failure for LoginModule_Login
ERROR | 20/11/2013 3:01:40:000 | session | user | com.guavus.common.login.commands.LoginCommand | LoginModule_Login | 2CC20E64-A91E-6C77-B5C7-74D9CB27F696 | | Login Event failure for LoginModule_Login is (mx.rpc::Fault)#0
    content = (null)
    errorID = 0
    faultCode = "Server.Processing"
    faultDetail = (null)
    faultString = "com.guavus.rubix.user.management.exceptions.LoginException : CHANGE_PASSWORD_PROMPT"
    message = "faultCode:Server.Processing faultString:'com.guavus.rubix.user.management.exceptions.LoginException : CHANGE_PASSWORD_PROMPT' faultDetail:'null'"
    name = "Error"
    rootCause = (com.guavus.common.login.models.exception::LoginException)#1
    cause = (null)
    code = "CHANGE_PASSWORD_PROMPT"
    localizedMessage = "CHANGE_PASSWORD_PROMPT"
    loginRequest = (com.guavus.common.login.models.domain::LoginRequestVO)#2
    samlToken = (null)
    sessionId = (null)
    timeZone = (null)
    userName = "admin"
    message = "CHANGE_PASSWORD_PROMPT"
ERROR | 20/11/2013 3:01:40:000 | session | user | com.guavus.common.application.commands.Command | LoginModule_Login | 2CC20E64-A91E-6C77-B5C7-74D9CB27F696 | | Event failure for LoginModule_Login is CHANGE_PASSWORD_PROMPT
ERROR | 20/11/2013 3:02:48:000 | 8C890D2BB250088C86DD4D35AD5E2DF7 | admin | com.guavus.common.widgets.filterWidget.models.presentation.FilterSummaryWidgetPM | | | UserInteraction-Org_StTime:1363132800::Org_EndTime:1363219200~Sel_StTime:1363132800::Sel_EndTime:1363219200~Network_Module~FilterSummaryWidget~FilterSummaryWidgetDoughnut~Filters:ViewBy~DOWN_BYTES~Selections:-Total~Actions:NA
ERROR | 20/11/2013 3:02:50:000 | 8C890D2BB250088C86DD4D35AD5E2DF7 | admin | com.guavus.common.widgets.smartInsituGridWidget.models.presentation.CategoryGridPM | | | UserInteraction-Org_StTime:1363132800::Org_EndTime:1363219200~Sel_StTime:1363132800::Sel_EndTime:1363219200~Network_Module~CategoryWidget~CategoryGrid~Filters:ViewBy~TRAFFIC_TYPE$Page-1$ItemRank-1~Selections:Categories$Agg UpLink Bitrate~Selections:NA~Actions:NA
ERROR | 20/11/2013 3:02:50:000 | 8C890D2BB250088C86DD4D35AD5E2DF7 | admin | com.guavus.common.widgets.trafficTrendWidget.widget.multipletimeseries.models.presentation.MultipleTimeSeriesPM | | | UserInteraction-Org_StTime:1363132800::Org_EndTime:1363219200~Sel_StTime:1363132800::Sel_EndTime:1363219200~Network_Module~TrafficSubWidget~MultipleTimeSeries~Filters:ViewBy~All Categories$Agg UpLink Bitrate~Selections:NA~Actions:NA
ERROR | 20/11/2013 3:18:44:000 | 8C890D2BB250088C86DD4D35AD5E2DF7 | admin | com.guavus.common.widgets.filterWidget.models.presentation.FilterSummaryWidgetPM | | | UserInteraction-Org_StTime:1363132800::Org_EndTime:1363219200~Sel_StTime:1363132800::Sel_EndTime:1363219200~Network_Module~FilterSummaryWidget~FilterSummaryWidgetDoughnut~Filters:ViewBy~DOWN_BYTES~Selections:-Total~Actions:NA
ERROR | 20/11/2013 3:18:44:000 | 8C890D2BB250088C86DD4D35AD5E2DF7 | admin | com.guavus.common.widgets.smartInsituGridWidget.models.presentation.CategoryGridPM | | | U

```

Confidential

Challenges

- Problem is not getting this big data. Problem is **how to store, process and analyze this data.**

Case Study

Telecom Company

- Airmobile (50 million subscribers) wants to sell it's expensive \$500 monthly plan to it's customers, for this it wants to find out its top subscribers and the total bytes they have downloaded(Internet data) using it's services in last one month.
- Also it wants to advertise it's roaming plan of \$100 so that subscribers don't switch to other networks, when going to other cities. For this it wants to find out the minutes of usage (i.e., call duration) of top 10 thousand subscribers who have roamed in last 1 month .

Confidential

Issues

- Different subscribers in different cities have different data plans. Almost all the subscribers are active each day of month.
- Data collection is huge every minute almost 1 million people visit 5–6 sites.
- Every day tera bytes of information is collected in airmobile servers of each city, which get discarded because of unavailability of storage.

Confidential

Solution

- Introduced by Google was GFS (Google file system) and Map Reduce.
- Then Hadoop became open source and now is owned by apache.
- Hadoop is used by Facebook, Yahoo, Google, Twitter, linkedin, Rackspace.

How is HADOOP the Solution

- Storage -> HDFS A distributed file system where commodity hardware can be used to form clusters and store the huge data in distributed fashion. There is no need for high end hardwares.
- Process -> MAP Reduce Paradigm
- Analyze -> Hive, Pig MapReduce.
- It can easily scale to multiple nodes(1,500–2,000 nodes in a cluster), with just configuration change.

Applications of HADOOP

- Telecommunications -> To find out top subscribers for advertisement, find peak traffic rate to install routers at right places, for cost cutting.
- Recommendation systems -> Google Ads customized for all users.
- Data warehousing -> to store data and analyze it e.g., categorize data into http web or mobile, so that services by ISP can be customized accordingly.
- Market Research and Forecasting -> Forecast subscribers, traffic based on past data trend.
- Finance, social networking -> To predict trends and gain profit.

What it is Not

- Should be noted it's not OLAP (online analytical Processing) but batch / offline oriented
- It is not a database

Storage HDFS

Challenges

- Can this data be stored in 1 machine? Hard drives are approximately 500Gb in size. Even if you add external hard drives, you can't store the data in Peta bytes. Let's say you add external hard drives and store this data, you wouldn't be able to open or process that file because of insufficient RAM. And processing, it would take months to analyze this data.

Confidential

HDFS Features

- Data is distributed over several machines, and replicated to ensure their durability to failure and high availability to parallel applications
- Designed for very large files (in GBs, TBs)
- Block oriented
- Unix like commands interface
- Write once and read many times
- Commodity hardware
- Fault Tolerant when nodes fail
- Scalable by adding new nodes

Confidential

It is Not Designed for

- Small files.
- Multiple writes, arbitrary file modification -> Writes are always supported at the end of the file, modifications can't be made at random offsets of files.
- Low latency data access -> Since we are accessing huge amount of data it comes at the expense of time taken to access the data.

Confidential

HDFS Design

- Individual files are broken into blocks of fixed size (typically 64 Mb >> 4Kb block structured FS), and stored across cluster of nodes. (Not necessarily on same machine). These files can be more than the size of individual machine's hard drive.
- Individual machines are called Data Nodes.
- So access to a file requires cooperation of several nodes.

Design Details

- Challenge -> Since several machines are involved in serving a file, loss of machine can result in failure of servicing the file request.
- HDFS Solution : Replicate each block across a number of machines (3 in the screenshot). Each block 64 Mb in size, this is to keep smaller metadata at name node for each file (list will be small).
- HDFS expects large files sequentially laid on disk for processing by MAP REDUCE. Small number of very large files. So for smaller data HDFS is not an optimal solution.

Confidential

Name Node: Stores Meta Data

Meta Data:
/data/pristine/catalina.log.> 1, 2, 4
/data/pristine/myfile. >3,5

Data Node 1

1

2

4

5

Data Node 2

5

2

3

Data Node 3

4

1

3

Function of Name Node

- Name Node is controller and manager of HDFS. It knows the status and the metadata of all the files in HDFS.
- Metadata is -> file names, permissions, and locations of each block of file
- HDFS cluster can be accessed concurrently by multiple clients, even then this metadata information is never desynchronized. Hence, all this information is handled by a single machine.
- Since metadata is typically small, all this info. is stored in main memory of Name Node, allowing fast access to metadata.

Name Node HA

- Since function of name node is very critical for overall health of HDFS, even if individual data nodes fail, HDFS can recover and function with a little less capacity however crash of name node can lose all the information and the complete file system irrecoverably. That's why metadata and involvement of name node in data transfer is kept minimal.

Purpose of Secondary Name Node

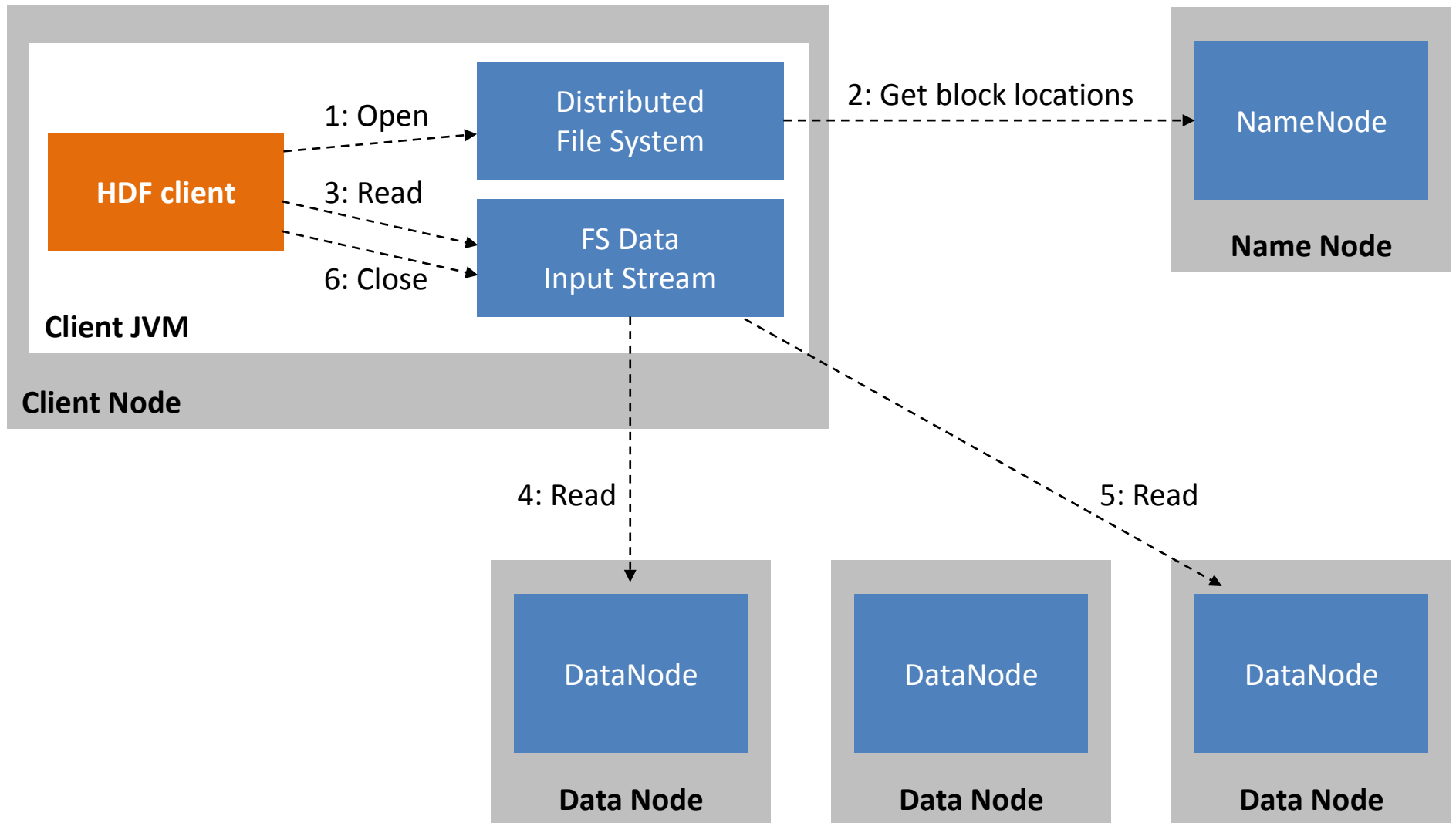
- It is not backup of name node nor data nodes connect to this. It is just a helper of name node.
- It only performs periodic checkpoints.
- It communicates with name node and to take snapshots of HDFS metadata.
- These snapshots help minimize downtime and loss of data.

Confidential

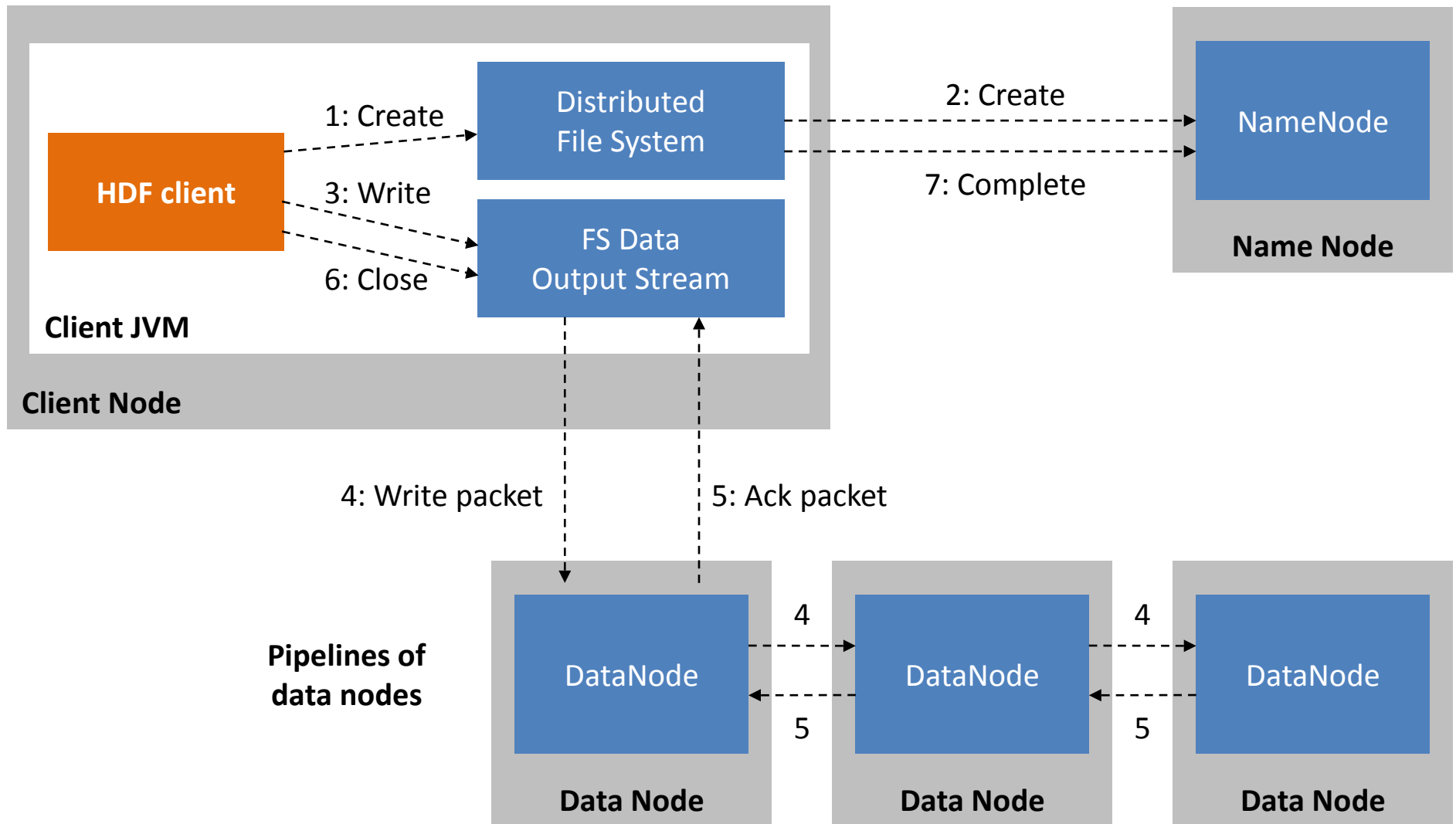
Overall Function

- To open a file, client contacts the name node retrieves list of locations of blocks of that file. These locations identify data nodes that hold the block. Clients then read the data directly from data nodes in parallel. Name Node is not involved in this stage.

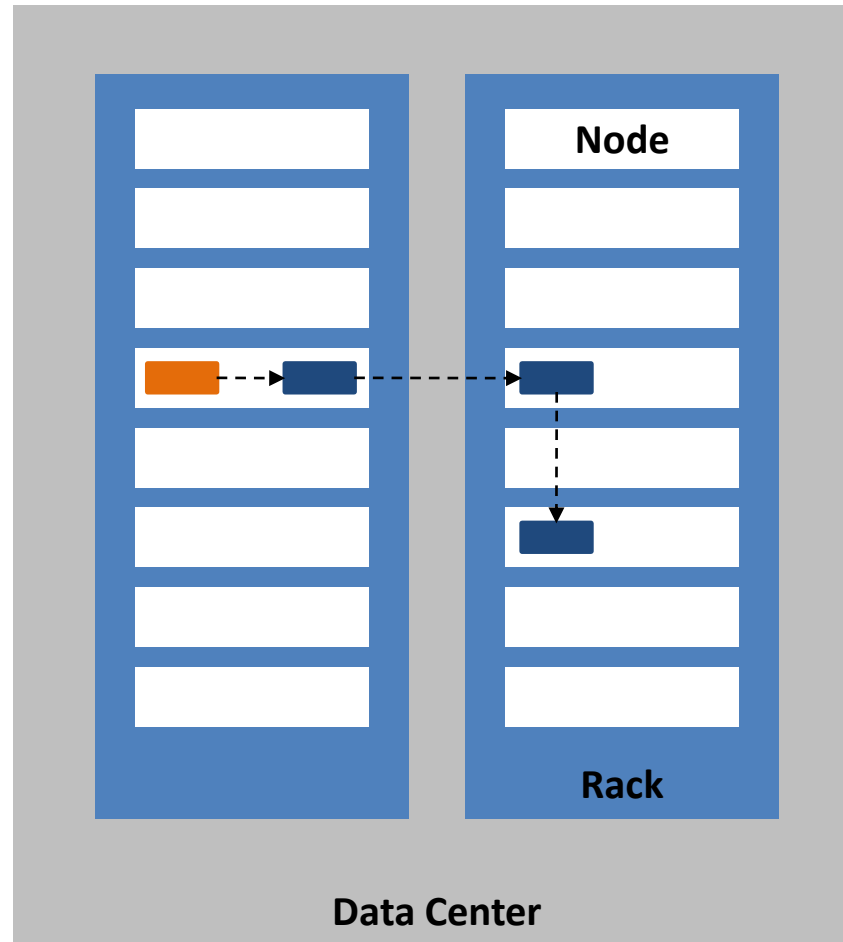
HDFS Read



HDFS Write



Replica Placement



How does it Run

- HDFS files are not part of ordinary file system. "ls" command will not list the files in HDFS.
- HADOOP runs as a separate process and in a different namespace isolated from normal FS on the OS.
- HDFS files/blocks are stored in a particular directory controlled by Datanode. Files are stored with block Ids. You can't interact with them using normal Unix commands.
- There are separate utilities similar to Unix that can be used to access the HDFS

```

hadoop-user@hadoop-desk: ~
hadoop-user@hadoop-desk:~$ hadoop dfs -ls /
Found 3 items
drwxr-xr-x - hadoop-user supergroup          0 2013-12-02 01:16 /hadoop
drwxr-xr-x - hadoop-user supergroup          0 2013-12-02 14:12 /jasleen
drwxr-xr-x - hadoop-user supergroup          0 2008-09-12 10:25 /user
hadoop-user@hadoop-desk:~$ hadoop dfs -mkdir /PRISTINE/
hadoop-user@hadoop-desk:~$ hadoop dfs -ls /
Found 4 items
drwxr-xr-x - hadoop-user supergroup          0 2013-12-02 18:27 /PRISTINE
drwxr-xr-x - hadoop-user supergroup          0 2013-12-02 01:16 /hadoop
drwxr-xr-x - hadoop-user supergroup          0 2013-12-02 14:12 /jasleen
drwxr-xr-x - hadoop-user supergroup          0 2008-09-12 10:25 /user
hadoop-user@hadoop-desk:~$ hadoop dfs -put /jasleen/newFile /PRISTINE/
hadoop-user@hadoop-desk:~$ hadoop dfs -ls /PRISTINE
Found 1 items
-rw-r--r--  1 hadoop-user supergroup          32 2013-12-02 18:28 /PRISTINE/newFile
hadoop-user@hadoop-desk:~$

```

Environment Needed for HADOOP

- The production and stable environment for HADOOP is Unix.
- You can run HADOOP on windows as well, but again you would need Unix like system (e.g., cygwin) to run HADOOP processes. For this cygwin should be run as a service in windows OS and can be a problem if user doesn't have admin rights
- Since most people have Windows OS, we will run HADOOP on preconfigured VM (given by Yahoo/CDH)

Confidential

Configuring and Setting up HDFS



- Download vm player from <http://www.vmware.com/support/download-player>

HADOOP vm from cloudera can be downloaded from <http://content.udacity-data.com/courses/ud617/Cloudera-Udacity-Training-VM-4.1.1.c.zip>

- HADOOP has introduced several versions the vm image present in the following link is 0.18 version of HADOOP <http://developer.yahoo.com/hadoop/tutorial/module3.html>
- Download putty from <http://www.putty.org/> and winscp <http://winscp.net/eng/download.php>

Steps

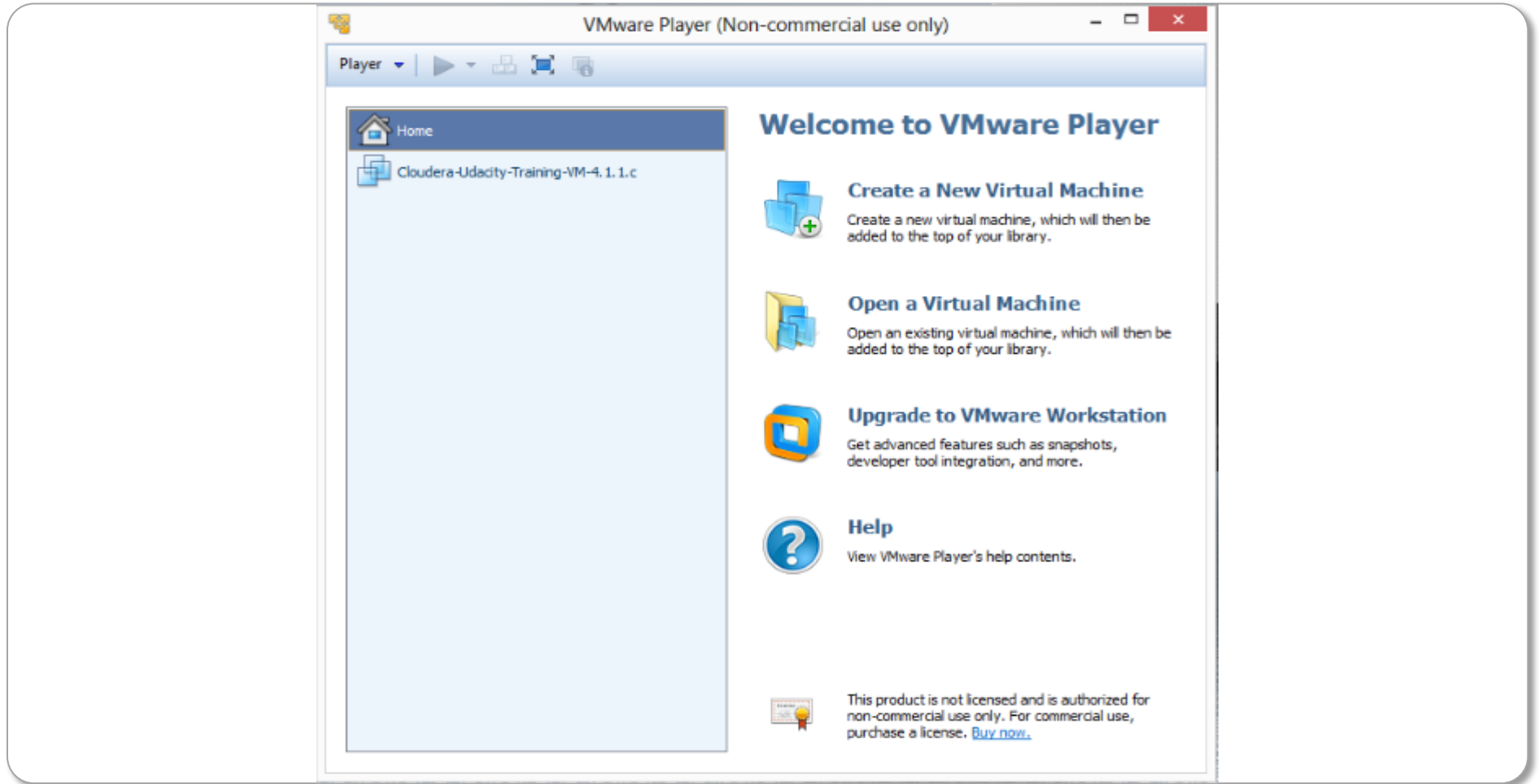
- After VMplayer is successfully installed, extract Cloudera-Udacity-4.1 zip to a folder

	Cloudera-Udacity-Training-VM-4.1.1.c	2/4/2014 1:41 PM	WinRAR ZIP archive	1,732,598 KB
	VMware-player-6.0.1-1379776	2/8/2014 5:47 PM	Application	96,200 KB

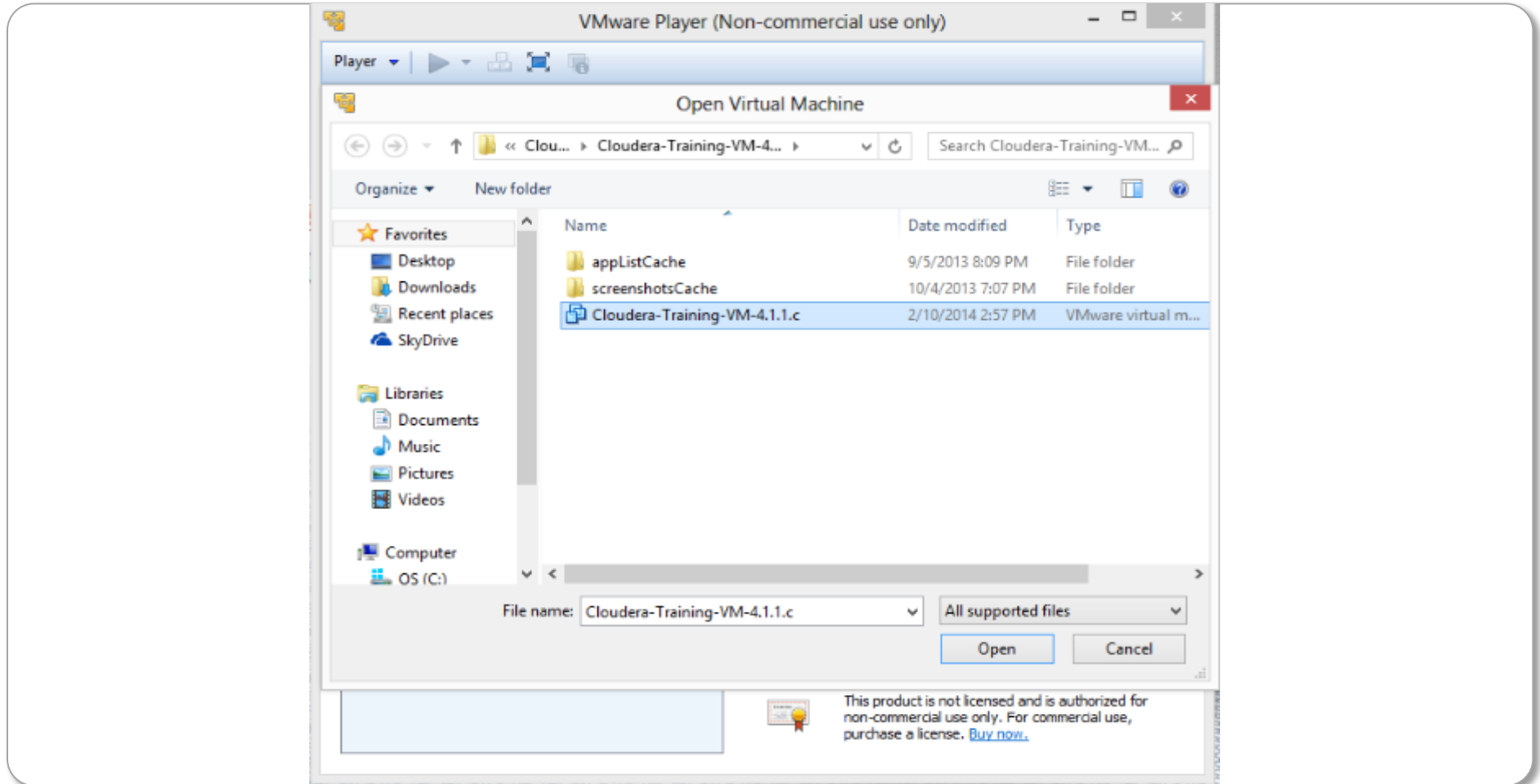
Confidential

VMware Setup

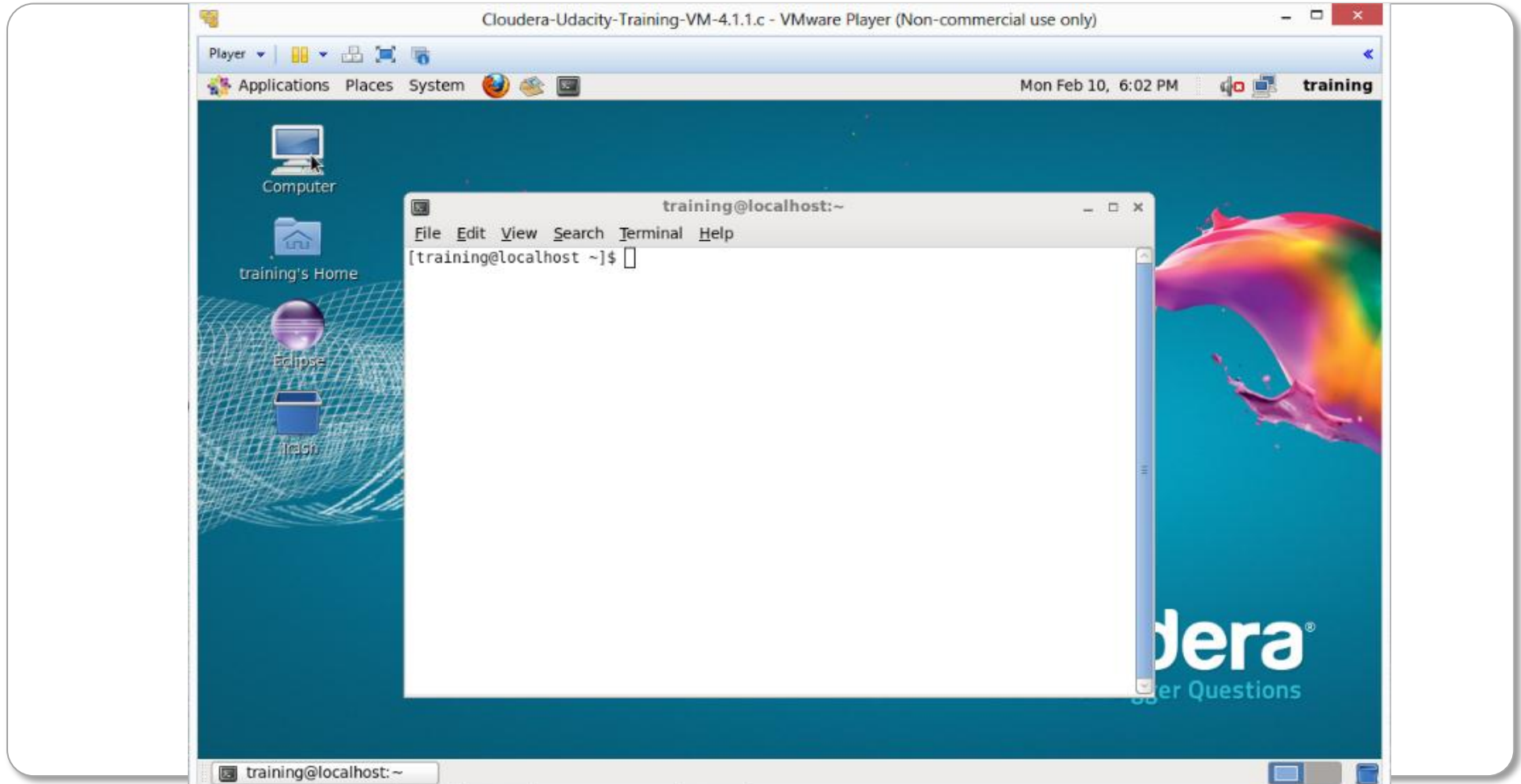
- Double click on VM player quick launcher, following screen will be opened



- Click on 'Open a Virtual Machine' and direct to extracted image folder.



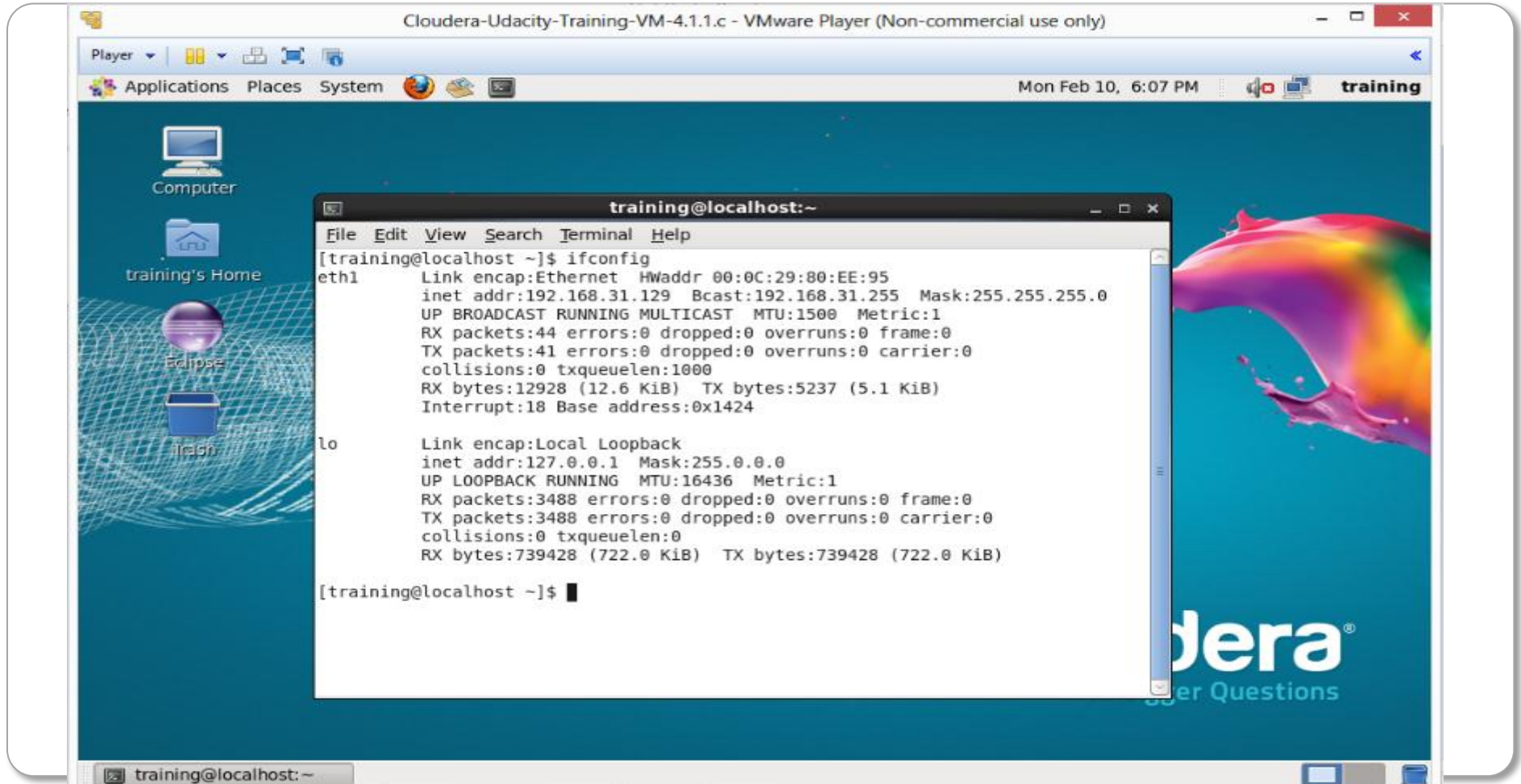
- After opening the image following information will be displayed in vm player



Cont.

- This vm image is configured with 2 users root and training. The password of training user is training.
- As soon as the vm is loaded correctly, an IP is assigned to it . In the screenshot attached the IP is 192.168.31.129.
- The user training is sudo user of the system.
- You can also connect to this vm using ssh.

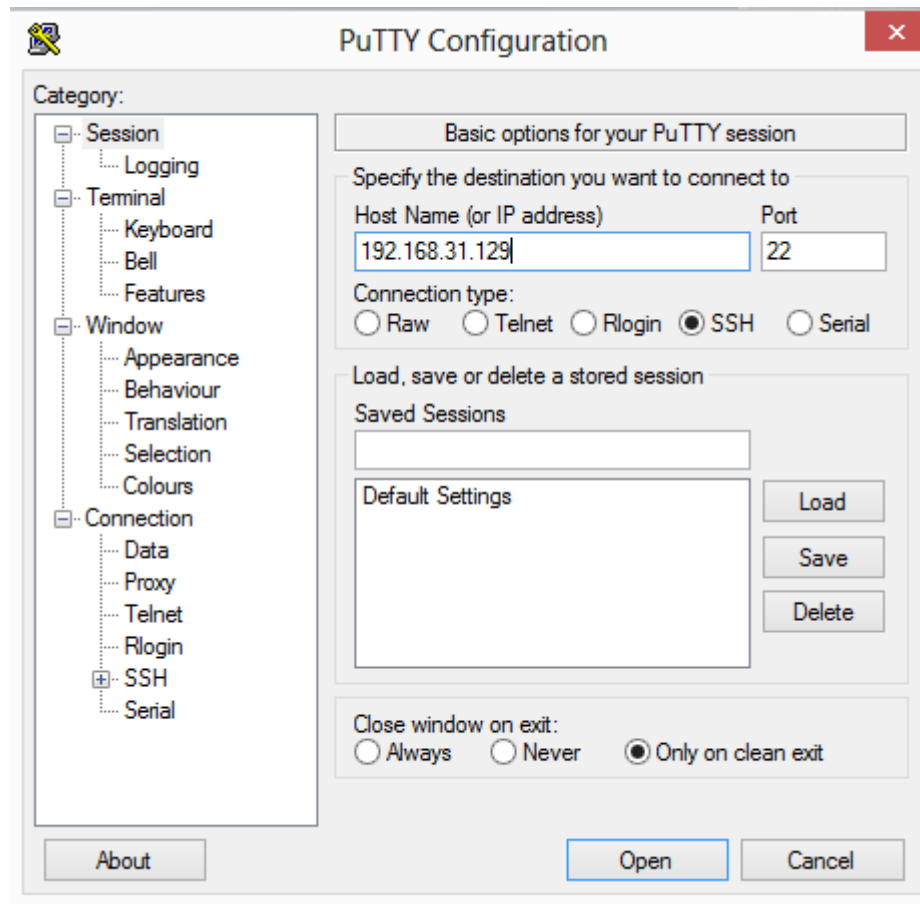
- The ip of vm can be found by typing ifconfig in terminal



Cont.

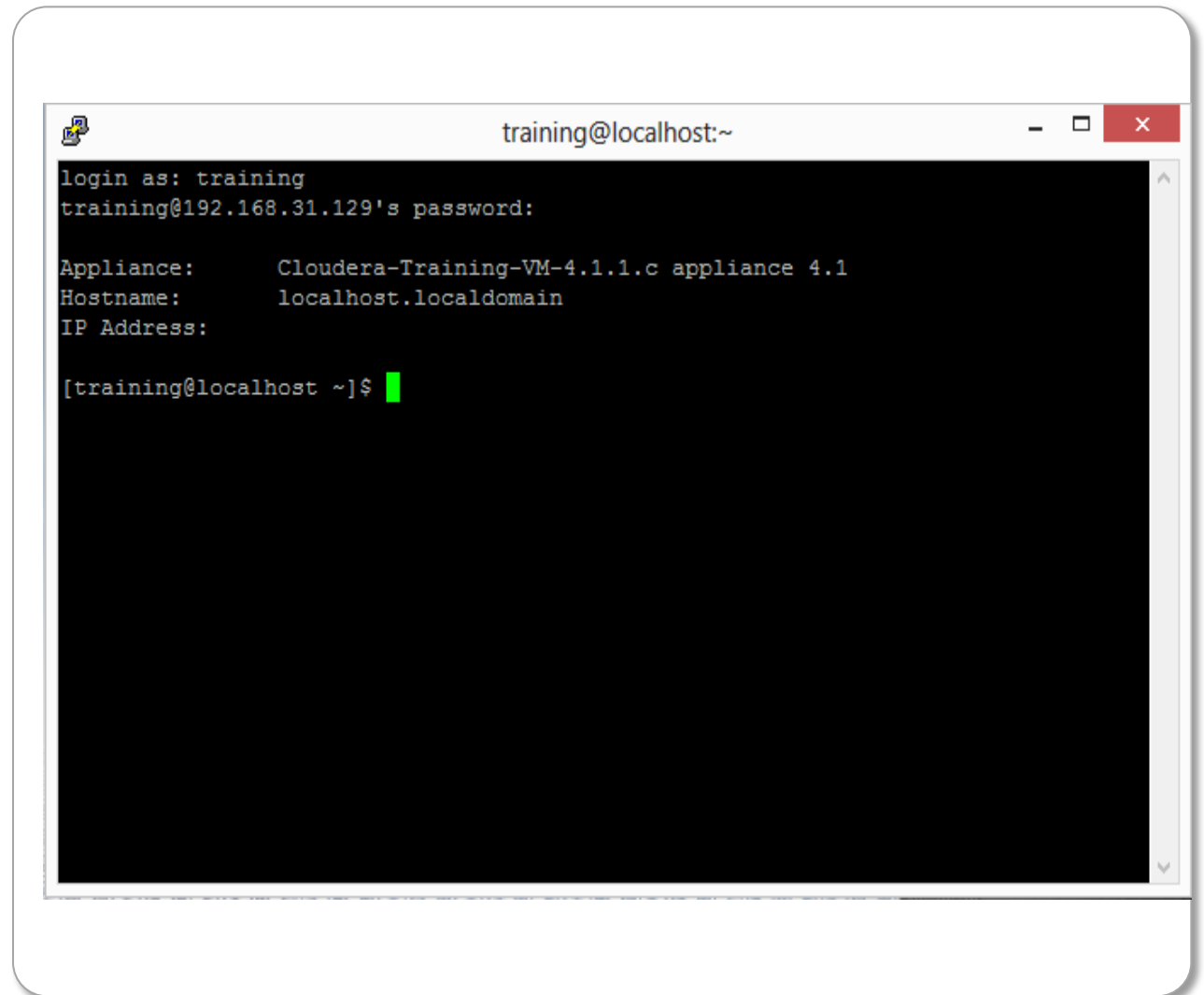
- To use the vm either directly use vm or putty.
- Whichever UI, student finds suitable can be used. Both are pointing to the same server.

- Double click on putty.exe and enter the IP of vm



Cont.

- Login training
- Password training
- After login following screen should appear



```
training@localhost:~  
login as: training  
training@192.168.31.129's password:  
  
Appliance:      Cloudera-Training-VM-4.1.1.c appliance 4.1  
Hostname:       localhost.localdomain  
IP Address:  
  
[training@localhost ~]$
```

- To check if HADOOP processes are running in the system, type
- `ps -ef | grep java`, name node, secondary name node, data node will be shown

```
cloudera@cloudera-vm:~$ ps -ef | grep java
hbase      1630      1 0 Dec07 ?        00:01:59 /usr/lib/jvm/java-6-sun/bin/java -Xmx1000m -ea -XX:+UseConcMarkSweepGC -XX:+CMSIncrementalMode -ea -XX:+UseConcMarkSweepGC -XX:+CMSIncrementalMode -ea -XX:+UseConcMarkSweepGC -XX:+CMSIncrementalMode -Dhbase.log.dir=/usr/lib/hbase/logs -Dhbase.log.file=hbase-hbase-master-cloudera-vm.log -Dhbase.home.dir=/usr/lib/hbase -Dhbase.id.str=hbase -Dhbase.root.logger=INFO,DRFA -classpath /usr/lib/hbase/conf:/usr/lib/jvm/java-6-sun/lib/tools.jar:/usr/lib/hbase:/usr/lib/hbase/hbase-0.90.1-cdh3u0.jar:/usr/lib/hbase/hbase-0.90.1-cdh3u0-tests.jar:/usr/lib/hbase/lib/activation-1.1.jar:/usr/lib/hbase/lib/asm-3.1.jar:/usr/lib/hbase/lib/avro-1.3.3.jar:/usr/lib/hbase/lib/commons-cli-1.2.jar:/usr/lib/hbase/lib/commons-codec-1.4.jar:/usr/lib/hbase/lib/commons-el-1.0.jar:/usr/lib/hbase/lib/commons-httpclient-3.1.jar:/usr/lib/hbase/lib/commons-lang-2.5.jar:/usr/lib/hbase/lib/commons-logging-1.1.1.jar:/usr/lib/hbase/lib/commons-net-1.4.1.jar:/usr/lib/hbase/lib/core-3.1.1.jar:/usr/lib/hbase/lib/guava-r06.jar:/usr/lib/hbase/lib/hadoop-core.jar:/usr/lib/hbase/lib/hbase-0.90.1-cdh3u0.jar:/usr/lib/hbase/lib/jackson-core-asl-1.5.2.jar:/usr/lib/hbase/lib/jackson-jaxrs-1.5.5.jar:/usr/lib/hbase/lib/jackson-mapper-asl-1.5.2.jar:/usr/lib/hbase/lib/jackson-xc-1.5.5.jar:/usr/lib/hbase/lib/jasper-compiler-5.5.23.jar:/usr/lib/hbase/lib/jasper-runtime-5.5.23.jar:/usr/lib/hbase/lib/jaxb-api-2.1.jar:/usr/lib/hbase/lib/jaxb-impl-2.1.12.jar:/usr/lib/hbase/lib/jersey-core-1.4.jar:/usr/lib/hbase/lib/jersey-json-1.4.jar:/usr/lib/hbase/lib/jersey-server-1.4.jar:/usr/lib/hbase/lib/jettison-1.1.jar:/usr/lib/hbase/lib/jetty-6.1.26.jar:/usr/lib/hbase/lib/jetty-util-6.1.26.jar:/usr/lib/hbase/lib/jruby-complete-1.0.3.jar:/usr/lib/hbase/lib/jsp-2.1-6.1.14.jar:/usr/lib/hbase/lib/jsp-api-2.1-6.1.14.jar:/usr/lib/hbase/lib/jsp-2.1.jar:/usr/lib/hbase/lib/jsr311-api-1.1.1.jar:/usr/lib/hbase/lib/log4j-1.2.16.jar:/usr/lib/hbase/lib/protobuf-java-2.3.0.jar:/usr/lib/hbase/lib/servlet-api-2.5-6.1.14.jar:/usr/lib/hbase/lib/servlet-api-2.5.jar:/usr/lib/hbase/lib/slf4j-api-1.5.8.jar:/usr/lib/hbase/lib/slf4j-log4j12-1.5.8.jar:/usr/lib/hbase/lib/stax-api-1.0.1.jar:/usr/lib/hbase/lib/thrift-0.2.0.jar:/usr/lib/hbase/lib/xmlenc-0.52.jar:/usr/lib/hbase/lib/zookeeper.jar org.apache.hadoop.hbase.master.HMaster start
hue         1772    1725 0 Dec07 ?        00:00:44 /usr/lib/jvm/java-6-sun/bin/java -Dproc.jar -Xmx1000m -Dlog4j.configuration=log4j.properties -Dhadoop.log.dir=/usr/lib/hadoop-0.20/logs -Dhadoop.log.file=hadoop.log -Dhadoop.home.dir=/usr/lib/hadoop-0.20 -Dhadoop.id.str=hadoop -Dhadoop.root.logger=INFO,console -Djava.library.path=/usr/lib/hadoop-0.20/lib/native/Linux-i386-32 -Dhadoop.policy.file=hadoop-policy.xml -classpath /etc/hive/conf:/usr/share/hue/apps/beeswax/src/beeswax/../../../../desktop/conf:/usr/lib/hadoop-0.20/conf:/usr/lib/jvm/java-6-sun/lib/tools.jar:/usr/lib/hadoop-0.20:/usr/lib/hadoop-0.20/hadoop-core-0.20.2-cdh3u0.jar:/usr/lib/hadoop-0.20/lib/ant-contrib-1.0b3.jar:/usr/lib/hadoop-0.20/lib/aspectjrt-1.6.5.jar:/usr/lib/hadoop-0.20/lib/aspectjtools-1.6.5.jar:/usr/lib/hadoop-0.20/lib/commons-cli-1.2.jar:/usr/lib/hadoop-0.20/lib/commons-codec-1.4.jar:/usr/lib/hadoop-0.20/lib/commons-daemon-1.0.1.jar:/usr/lib/hadoop-0.20/lib/commons-el-1.0.jar:/usr/lib/hadoop-0.20/lib/commons-httpclient-3.0.1.jar:/usr/lib/hadoop-0.20/lib/commons-logging-1.0.4.jar:/usr/lib/hadoop-0.20/lib/commons-logging-api-1.0.4.jar:/usr/lib/hadoop-0.20/lib/commons-net-1.4.1.jar:/usr/lib/hadoop-0.20/lib/core-3.1.1.jar:/usr/lib/hadoop-0.20/lib/hadoop-fairscheduler-0.20.2-cdh3u0.jar:/usr/lib/hadoop-0.20/lib/hsql-1.8.0.10.jar:/usr/lib/hadoop-0.20/lib/hue-plugins-1.2.0.jar:/usr/lib/hadoop-0.20/lib/jackson-core-asl-1.5.2.jar:/usr/lib/hadoop-0.20/lib/jackson-mapper-asl-1.5.2.jar:/usr/lib/hadoop-0.20/lib/jasper-compiler-5.5.12.jar:/usr/lib/hadoop-0.20/lib/jasper-runtime-5.5.12.jar:/usr/lib/hadoop-0.20/lib/jets3t-0.6.1.jar:/usr/lib/hadoop-0.20/lib/jetty-6.1.26.jar:/usr/lib/hadoop-0.20/lib/jetty-servlet-tester-6.1.26.jar:/usr/lib/hadoop-0.20/lib/jetty-util-6.1.26.jar:/usr/lib/hadoop-0.20/lib/jsch-0.1.42.jar:/usr/lib/hadoop-0.20/lib/junit-4.5.jar:/usr/lib/hadoop-0.20/lib/kfs-0.2.2.jar:/usr/lib/hadoop-0.20/lib/log4j-1.2.15.jar:/usr/lib/hadoop-0.20/lib/mockito-all-1.8.2.jar:/usr/lib/hadoop-0.20/lib/oro-2.0.8.jar:/usr/lib/hadoop-0.20/lib/servlet-api-2.5-20081211.jar:/usr/lib/hadoop-0.20/lib/servlet-api-2.5-6.1.14.jar:/usr/lib/hadoop-0.20/lib/slf4j-api-1.4.3.jar:/usr/lib/hadoop-0.20/lib/slf4j-log4j12-1.4.3.jar:/usr/lib/hadoop-0.20/lib/xmlenc-0.52.jar:/usr/lib/hadoop-0.20/lib/jsp-2.1/jsp-2.1.jar:/usr/lib/hadoop-0.20/lib/jsp-api-2.1.jar:/usr/lib/hadoop-0.20/lib/hive-shims-0.7.0-cdh3u0.jar:/usr/lib/hive/lib/hbase-0.90.1-cdh3u0.jar:/usr/lib/hive/lib/hive-service-0.7.0-cdh3u0.jar:/usr/lib/hive/lib/datanucleus-connectionpool-2.0.3.jar:/usr/lib/hive/lib/datanucleus-rdbms-2.0.3.jar:/usr/lib/hive/lib/hive-exec-0.7.0-cdh3u0.jar:/usr/lib/hive/lib/asm-3.1.jar:/usr/lib/hive/lib/commons-collect-ions-3.2.1.jar:/usr/lib/hive/lib/antlr-runtime-3.0.1.jar:/usr/lib/hive/lib/commons-lang-2.4.jar:/usr/lib/hive/lib/commons-logging-1.0.4.jar:/usr/lib/hive/lib/hive-serde-0.7.0-cdh3u0.jar:/usr/lib/hive/lib/hive-hwi-0.7.0-cdh3u0.jar:/usr/lib/hive/lib/hive/thrift-0.5.0.jar:/usr/lib/hive/lib/commons-logging-api-1.0.4.jar:/usr/lib/hive/lib/junit-3.8.1.jar:/usr/lib/hive/lib/hive-contrib-0.7.0-cdh3u0.jar:/usr/lib/hive/lib/hive-cli-0.7.0-cdh3u0.jar:/usr/lib/hive/lib/hive/line-0.9.94.jar:/usr/lib/hive/lib/log4j-1.2.15.jar:/usr/lib/hive/lib/guava-r06.jar:/usr/lib/hive/lib/hive-jdbc-0.7.0-cdh3u0.jar:/usr/lib/hive/lib/hive-common-0.7.0-cdh3u0.jar:/usr/lib/hive/lib/stringtemplate-3.1b1.jar:/usr/lib/hive/lib/hive/json.jar:/usr/lib/hive/lib/hive-anttasks-0.7.0-cdh3u0.jar:/usr/lib/hive/lib/jdo2-api-2.3-ec.jar:/usr/lib/hive/lib/slf4j-log4j12-1.6.1.jar:/usr/lib/hive/lib/hive/velocity-1.5.jar:/usr/lib/hive/lib/commons-cli-1.2.jar:/usr/lib/hive/lib/commons-dbc-1.4.jar:/usr/lib/hive/lib/hbase-0.90.1-cdh3u0-tests.jar:/usr/lib/hive/lib/commons-codec-1.3.jar:/usr/lib/hive/lib/datanucleus-core-2.0.3.jar:/usr/lib/hive/lib/commons-pool-1.5.4.jar:/usr/lib/hive/lib/zookeeper-3.3.1.jar:/usr/lib/hive/lib/datanucleus-enhancer-2.0.3.jar:/usr/lib/hive/lib/libthrift.jar:/usr/lib/hive/lib/derby.jar:/usr/lib/hive/lib/hive-metastore-0.7.0-cdh3u0.jar:/usr/lib/hive/lib/log4j-1.2.16.jar:/usr/lib/hive/lib/slf4j-api-1.6.1.jar:/usr/lib/hive/lib/ant-contrib-1.0b3.jar:/usr/lib/hive/lib/hive-hbase-oozie 1915      1 1 Dec07 ?        00:14:28 /usr/bin/java -Djava.util.logging.config.file=/var/lib/oozie/oozie-server/conf/logging.properties -Djava.util.logging.manager=org.apache.juli.ClassLoaderLogManager -Dderby.stream.error.file=/var/log/oozie/derby.log -Doozie.home.dir=/usr/lib/oozie -Doozie.config.dir=/etc/oozie -Doozie.log.dir=/var/log/oozie -Doozie.data.dir=/var/lib/oozie -Doozie.config.file=oozie-site.xml -Doozie.log4j.file=oozie-log4j.properties -Doozie.log4j.reload=10 -Doozie.http.hostname=cloudera-vm -Doozie.http.port=11000 -Doozie.base.url=http://cloudera-vm:11000/oozie -Djava.endorsed.dirs=/usr/lib/oozie/oozie-server/endorsed -classpath /usr/lib
```

Setup Cluster (Without Virtual machine on systems with Linux)

- In a typical installation, hadoop can be downloaded from <http://hadoop.apache.org/releases.html>
- Extract the gz file to your filesystem
- **tar xzf hadoop-x.y.z.tar.gz in some path /usr/pristine/hadoop-x.y.z**
- HADOOP needs java to run it's processes, and hence JAVA_HOME should be set in the path using `export JAVA_HOME=/usr/jre16/bin/`
- Otherwise JRE location can be set in `conf/hadoop-env.sh`
- Set HADOOP path `export PATH=$PATH:/usr/pristine/hadoopx.y.z/bin`
- Check if HADOOP is set by
- **HADOOP version**

Confidential

Configurations

- Each component is configured using an XML file.
- NOTE: In HADOOP version 0.18 all the confs are done in a single file `hadoop-site.xml`
- In newer versions from 0.20 onwards this file is split into 3 files (Properties names are same only the files have been split)
- `core-site.xml` -> Core properties go here
- `hdfs-site.xml` -> HDFS Properties go here
- `mapred-site.xml` -> Map reduce properties are configured here
- Since we are using CDH4 it uses HADOOP version 2.0.0

Modes

- Standalone mode-> No daemons run, everything runs in single process. Suitable for debugging MR programs
- Pseudo Distributed -> All HADOOP processes run on local machine, simulating small cluster of 1 node
- Fully Distributed -> HADOOP runs on a cluster of machines.

Properties

Component	Property	Standalone	Pseudo-distributed	Fully distributed
Core	fs.default.name	file:/// (default)	hdfs://localhost/	hdfs://namenode/
HDFS	dfs.replication	N/A	1	3 (default)
MapReduce	mapred.job.tracker	local(default)	localhost:8021	jobtracker:8021

mapred-site.xml

- mapred-site.xml location /usr/lib/hadoop-0.20-mapreduce/conf/

```

limitations under the License.
-->
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>

<configuration>
  <property>
    <name>mapred.job.tracker</name>
    <value>0.0.0.0:8021</value>
  </property>

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

  <property>
    <description>To set the value of tmp directory for map and reduce tasks.</description>
    <name>mapreduce.task.tmp.dir</name>
    <value>/var/lib/hadoop-mapreduce/cache/${user.name}/tasks</value>
  </property>

  <property>
    <name>jobtracker.thrift.address</name>
    <value>0.0.0.0:9290</value>
  </property>
  <property>
    <name>mapred.jobtracker.plugins</name>
    <value>org.apache.hadoop.thriftfs.ThriftJobTrackerPlugin</value>
    <description>Comma-separated list of jobtracker plug-ins to be activated.</description>
  </property>

  <property>
    <name>mapred.task.timeout</name>
    <value>60000</value>
  </property>

  <property>
    <name>mapred.map.tasks</name>
    <value>1</value>
  </property>
</configuration>
[training@localhost conf]$

```

Confidential

hdfs-site.xml

- hdfs-site.xml location /usr/lib/hadoop-0.20-mapreduce/conf

```

training@localhost:/usr/lib/hadoop-0.20-mapreduce/conf
On a multi-node cluster, these configurations must be removed. -->
<property>
  <name>dfs.safemode.extension</name>
  <value>0</value>
</property>
<property>
  <name>dfs.safemode.min.datanodes</name>
  <value>1</value>
</property>
<property>
  <name>dfs.permissions.enabled</name>
  <value>>false</value>
</property>
<property>
  <name>dfs.permissions</name>
  <value>>false</value>
</property>
<property>
  <name>dfs.safemode.min.datanodes</name>
  <value>1</value>
</property>
<property>
  <name>dfs.webhdfs.enabled</name>
  <value>true</value>
</property>
<property>
  <name>hadoop.tmp.dir</name>
  <value>/var/lib/hadoop-hdfs/cache/${user.name}</value>
</property>
<property>
  <name>dfs.namenode.name.dir</name>
  <value>/var/lib/hadoop-hdfs/cache/${user.name}/dfs/name</value>
</property>
<property>
  <name>dfs.namenode.checkpoint.dir</name>
  <value>/var/lib/hadoop-hdfs/cache/${user.name}/dfs/namesecondary</value>
</property>
<property>
  <name>dfs.datanode.data.dir</name>
  <value>/var/lib/hadoop-hdfs/cache/${user.name}/dfs/data</value>
</property>
</configuration>
[training@localhost conf]$

```

Confidential

core-site.xml

- core-site.xml location /usr/lib/hadoop-0.20-mapreduce/conf

```

training@localhost:/usr/lib/hadoop-0.20-mapreduce/conf
See the License for the specific language governing permissions and
limitations under the License.
-->
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>

<configuration>
  <property>
    <name>fs.default.name</name>
    <value>hdfs://0.0.0.0:8020</value>
  </property>

  <!-- OOZIE proxy user setting -->
  <property>
    <name>hadoop.proxyuser.oozie.hosts</name>
    <value>*</value>
  </property>
  <property>
    <name>hadoop.proxyuser.oozie.groups</name>
    <value>*</value>
  </property>

  <!-- HTTPFS proxy user setting -->
  <property>
    <name>hadoop.proxyuser.httpfs.hosts</name>
    <value>*</value>
  </property>
  <property>
    <name>hadoop.proxyuser.httpfs.groups</name>
    <value>*</value>
  </property>

  <!-- Hue proxy user setting -->
  <property>
    <name>hadoop.proxyuser.hue.hosts</name>
    <value>*</value>
  </property>
  <property>
    <name>hadoop.proxyuser.hue.groups</name>
    <value>*</value>
  </property>
</configuration>
[training@localhost conf]$ █

```

Confidential

VM Configurations

- The VM provided is preconfigured with these properties, the properties can be verified by these files present in location `/usr/lib/hadoop/conf`

VM Configurations

- After checking the configurations, to check the status and health of HDFS do
- HADOOP fsck / -files -blocks
- This command gives the blocks and metadata information of HDFS i.e., directory namespace information, size, block Id, permissions etc
- If the setup is successful status will be shown HEALTHY

```
training@localhost:usr/lib/hadoop-0.20-mapreduce/conf
999-127.0.0.1-1355878275444:blk_4126978484291124050_1604. Target Replicas is 10 but found 1 replica(s).
0. BP-2100437999-127.0.0.1-1355878275444:blk_4126978484291124050_1604 len=29532 repl=1

/var/lib/hadoop-hdfs/cache/mapred/mapred/staging/training/.staging/job_201402091753_0003/libjars/servlet-api-2.5-20081211.jar 133725 bytes, 1 block(s): Under replicate
d BP-2100437999-127.0.0.1-1355878275444:blk_3055206142112632530_1608. Target Replicas is 10 but found 1 replica(s).
0. BP-2100437999-127.0.0.1-1355878275444:blk_3055206142112632530_1608 len=133725 repl=1

/var/lib/hadoop-hdfs/cache/mapred/mapred/staging/training/.staging/job_201402091753_0003/libjars/snappy-java-1.0.4.1.jar 990097 bytes, 1 block(s): Under replicated BP-
2100437999-127.0.0.1-1355878275444:blk_9084453044116369509_1612. Target Replicas is 10 but found 1 replica(s).
0. BP-2100437999-127.0.0.1-1355878275444:blk_9084453044116369509_1612 len=990097 repl=1

/var/lib/hadoop-hdfs/cache/mapred/mapred/staging/training/.staging/job_201402091753_0003/libjars/sqljdbc4-3.0.jar 536204 bytes, 1 block(s): Under replicated BP-2100437
999-127.0.0.1-1355878275444:blk_1032327904439019437_1606. Target Replicas is 10 but found 1 replica(s).
0. BP-2100437999-127.0.0.1-1355878275444:blk_1032327904439019437_1606 len=536204 repl=1

/var/lib/hadoop-hdfs/cache/mapred/mapred/staging/training/.staging/job_201402091753_0003/libjars/sqoop-1.4.1-cdh4.1.1.jar 603201 bytes, 1 block(s): Under replicated BP
-2100437999-127.0.0.1-1355878275444:blk_-536931839949436293_1620. Target Replicas is 10 but found 1 replica(s).
0. BP-2100437999-127.0.0.1-1355878275444:blk_-536931839949436293_1620 len=603201 repl=1

/var/lib/hadoop-hdfs/cache/mapred/mapred/system <dir>
/var/lib/hadoop-hdfs/cache/mapred/mapred/system/jobtracker.info 4 bytes, 1 block(s): OK
0. BP-2100437999-127.0.0.1-1355878275444:blk_1896891380279557583_1709 len=4 repl=1

Status: HEALTHY
Total size: 21625622 B
Total dirs: 38
Total files: 80
Total blocks (validated): 76 (avg. block size 284547 B)
Minimally replicated blocks: 76 (100.0 %)
Over-replicated blocks: 0 (0.0 %)
Under-replicated blocks: 56 (73.68421 %)
Mis-replicated blocks: 0 (0.0 %)
Default replication factor: 1
Average block replication: 1.0
Corrupt blocks: 0
Missing replicas: 504 (86.89655 %)
Number of data-nodes: 1
Number of racks: 1
FSCK ended at Mon Feb 10 19:54:09 EST 2014 in 58 milliseconds

The filesystem under path '/' is HEALTHY
[training@localhost conf]$ mapred-site.xml
```


HDFS Interfaces / Clients

- HTTP Interfaces with browsers
- Using CLI (Command line) utility
- Programmatically using Java
- Using thrift APIs with other languages like C
- Most common use is with command line and Java APIs

HTTP Interfaces

HTTP APIs

- HTTP: HDFS defines read only interface to browse directory listings and data over HTTP. HTTP web server runs on top of name node on port 50070, while file data is streamed from data nodes by their web servers running on 50075

Hadoop NameNode 0.0.0.0:8020 - Mozilla Firefox

File Edit View History Bookmarks Tools Help

Hadoop NameNode 0.0.0.0:8020

localhost:50070/dfshealth.jsp

The Platform for Bi... Hadoop JobTracker Hadoop NameNode

NameNode '0.0.0.0:8020' (active)

Started: Mon Feb 10 19:17:04 EST 2014
Version: 2.0.0-cdh4.1.1, 581959ba23e4af85afd8db98b7687662fe9c5f20
Compiled: Tue Oct 16 11:07:59 PDT 2012 by jenkins from Unknown
Upgrades: There are no upgrades in progress.
Cluster ID: CID-5920c32c-c3f1-42e5-81a0-c93032e8442a
Block Pool ID: BP-2100437999-127.0.0.1-1355878275444

[Browse the filesystem](#)
[NameNode Logs](#)

Cluster Summary

Security is OFF
118 files and directories, 76 blocks = 194 total.
Heap Memory used 13.73 MB is 68% of Committed Heap Memory 20.06 MB. Max Heap Memory is 193.38 MB.
Non Heap Memory used 22.58 MB is 99% of Committed Non Heap Memory 22.62 MB. Max Non Heap Memory is 96 MB.

Configured Capacity	:	14.08 GB			
DFS Used	:	21.44 MB			
Non DFS Used	:	4.23 GB			
DFS Remaining	:	9.83 GB			
DFS Used%	:	0.15 %			
DFS Remaining%	:	69.79 %			
Block Pool Used	:	21.44 MB			
Block Pool Used%	:	0.15 %			
DataNodes usages	:	Min %	Median %	Max %	stdev %
	:	0.15 %	0.15 %	0.15 %	0 %
Live Nodes	:	1 (Decommissioned: 0)			
Dead Nodes	:	0 (Decommissioned: 0)			
Decommissioning Nodes	:	0			
Number of Under-Replicated Blocks	:	56			

NameNode Journal Status:
Current transaction ID: 2520

HDFS: / - Mozilla Firefox

File Edit View History Bookmarks Tools Help

HDFS: /

localhost.localdomain:50075/browseDirectory.jsp?namenodeInfoPort=50070&dir=/&nnaddr=0.0.0.0:8020

Google

The Platform for Bi... Hadoop JobTracker Hadoop NameNode

Contents of directory /

Goto : go

Name	Type	Size	Replication	Block Size	Modification Time	Permission	Owner	Group
tmp	dir				2014-02-07 19:00	rw-rw-rw-	hdfs	supergroup
user	dir				2013-09-10 10:37	rw-rw-rw-	hue	supergroup
var	dir				2013-09-05 20:08	rw-r-xr-x	hdfs	supergroup

[Go back to DFS home](#)

Local logs

[Log](#) directory

[Hadoop](#), 2014.

Command Line Utilities

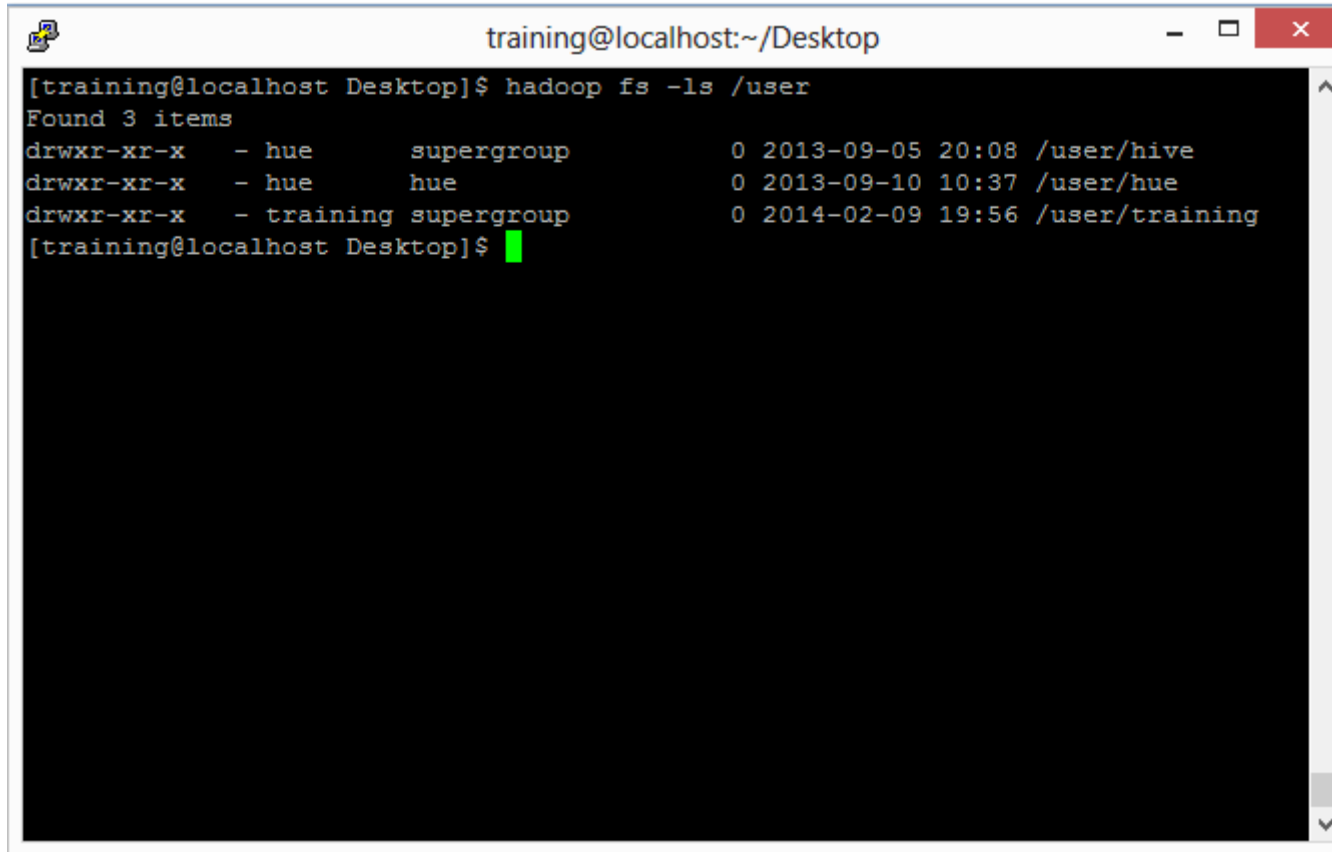
HADOOP Shell commands

- http://hadoop.apache.org/docs/r0.18.3/hdfs_shell.html

Command to List Files and DIRS

- ls
- Usage: HADOOP fs -ls <args>
- For a file returns stat on the file with the following format:
filename <number of replicas> filesize modification_date modification_time permissions userid
groupid
- For a directory it returns list of its direct children as in unix. A directory is listed as:
dirname <dir> modification_time modification_time permissions userid groupid

Example ls



```
training@localhost:~/Desktop
[training@localhost Desktop]$ hadoop fs -ls /user
Found 3 items
drwxr-xr-x  - hue      supergroup      0 2013-09-05 20:08 /user/hive
drwxr-xr-x  - hue      hue              0 2013-09-10 10:37 /user/hue
drwxr-xr-x  - training supergroup      0 2014-02-09 19:56 /user/training
[training@localhost Desktop]$
```

Practice Exercise

- Use command ls and see the output from command line and then do HADOOP ls

HADOOP mkdir

- mkdir – Takes path URI as argument and creates directory
- Usage: `hadoop fs -mkdir <paths>`
- Takes path uri's as argument and creates directories. The behavior is much like unix `mkdir -p` creating parent directories along the path
- Exit Code:
- Returns 0 on success and -1 on error

Example mkdir

```

training@localhost:~/Desktop
[training@localhost Desktop]$ hadoop fs -ls /user/training
Found 2 items
drwxr-xr-x  - training supergroup          0 2014-02-09 19:14 /user/training/ext1
drwxr-xr-x  - training supergroup          0 2014-02-09 19:55 /user/training/ext3
[training@localhost Desktop]$ hadoop fs -mkdir /user/training/trial_dir
[training@localhost Desktop]$ hadoop fs -ls /user/training
Found 3 items
drwxr-xr-x  - training supergroup          0 2014-02-09 19:14 /user/training/ext1
drwxr-xr-x  - training supergroup          0 2014-02-09 19:55 /user/training/ext3
drwxr-xr-x  - training supergroup          0 2014-02-10 20:15 /user/training/trial_dir
[training@localhost Desktop]$ █

```

Practice Exercise

- Make a HADOOP directory using mkdir command with your name and then do ls for that directory
- mkdir <yourName>

HADOOP put

- Usage: `hadoop fs -put <localsrc> ... <dst>`
- Copy single src, or multiple srcs from local file system to the destination filesystem. Also reads input from stdin and writes to destination filesystem.

Example Put

```

training@localhost:~/Desktop
[training@localhost Desktop]$ hadoop fs -ls /user/training
Found 3 items
drwxr-xr-x  - training supergroup          0 2014-02-09 19:14 /user/training/ext1
drwxr-xr-x  - training supergroup          0 2014-02-09 19:55 /user/training/ext3
drwxr-xr-x  - training supergroup          0 2014-02-10 20:15 /user/training/trial_direct
[training@localhost Desktop]$ ls
direc.png  Eclipse.desktop  namenode.png  new_file  sqljdbc4-3.0.jar.zip
[training@localhost Desktop]$ hadoop fs -put ./new_file /user/training/
[training@localhost Desktop]$ hadoop fs -ls /user/training
Found 4 items
drwxr-xr-x  - training supergroup          0 2014-02-09 19:14 /user/training/ext1
drwxr-xr-x  - training supergroup          0 2014-02-09 19:55 /user/training/ext3
-rw-r--r--  1 training supergroup        34 2014-02-10 20:26 /user/training/new_file
drwxr-xr-x  - training supergroup          0 2014-02-10 20:15 /user/training/trial_direct
[training@localhost Desktop]$

```

Practice Exercise

- Write a file in your local file system of vm using vi editor, save that file. Now put that file in your named directory in HDFS using put command.
- vi <myname>
- HADOOP dfs -put <myname> <hadoop destination>

HADOOP Get

- **Get**
- Usage: `hadoop fs -get [-ignorecrc] [-crc] <src> <localdst>`
- Copy files to the local file system. Files that fail the CRC check may be copied with the –ignorecrc option. Files and CRCs may be copied using the -crc option.

Example Get

```

training@localhost:~/Desktop
[training@localhost Desktop]$ ls
direc.png Eclipse.desktop namenode.png sqljdbc4-3.0.jar.zip
[training@localhost Desktop]$ hadoop fs -get /user/training/new_file ./
[training@localhost Desktop]$ ls
direc.png Eclipse.desktop namenode.png new_file sqljdbc4-3.0.jar.zip
[training@localhost Desktop]$

```

Practice Exercise

- Get the file you saved in HDFS to your local file system of VM, and read their contents and check if files are same.

HADOOP Copy to Local

- Usage: `hadoop fs -copyToLocal [-ignorecrc] [-crc] URI <localdst>`
- Similar to **get** command, except that the destination is restricted to a local file reference.

Confidential

HADOOP Copy to Local

```

training@localhost:~/Desktop
[training@localhost Desktop]$ ls
direc.png  Eclipse.desktop  namenode.png  sqljdbc4-3.0.jar.zip
[training@localhost Desktop]$ hadoop fs -copyToLocal /user/training/new_file ./file_copied
[training@localhost Desktop]$ ls
direc.png  Eclipse.desktop  file_copied  namenode.png  sqljdbc4-3.0.jar.zip
[training@localhost Desktop]$

```

Java APIs

Java API's

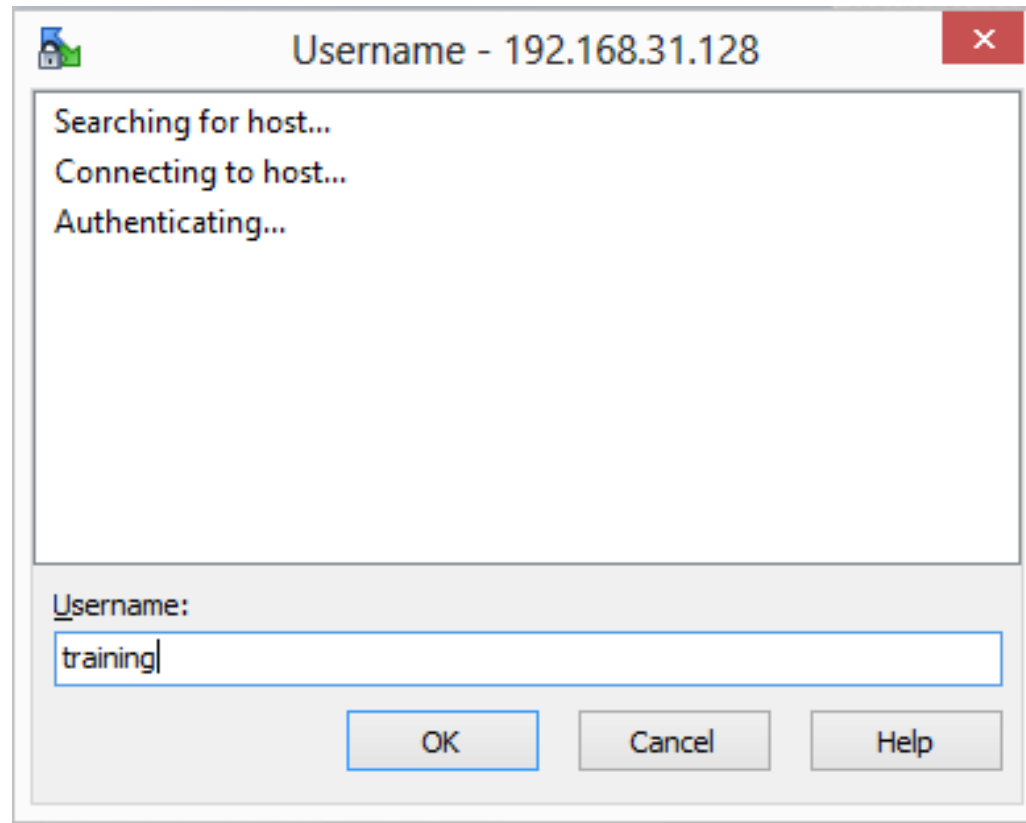
- HDFS can be accessed with JAVA API's as well
- You can add/delete/view file system from them

Environment

- To access hadoop API's hadoop jars should be there in build path of eclipse
- **From /usr/lib/hadoop/client-0.20**

copy this complete folder to your local system and add these jars into your build path

- Use WINSCP to copy the Jars to local system



- /usr/lib/hadoop/client-0.20

/usr/lib/hadoop/client-0.20		
Name	Ext	Size
..		
zookeeper-3.4.3-cdh4.1.1.jar		48 B
xmlenc-0.52.jar		35 B
snappy-java-1.0.4.1.jar		43 B
slf4j-log4j12-1.6.1.jar		43 B
slf4j-api-1.6.1.jar		39 B
servlet-api-2.5.jar		39 B
protobuf-java-2.4.0a.jar		44 B
paranamer-2.3.jar		37 B
oro-2.0.8.jar		48 B
mockito-all-1.8.5.jar		41 B
log4j-1.2.17.jar		36 B
junit-4.8.2.jar		35 B
jsr305-1.3.9.jar		36 B
jsp-api-2.1.jar		35 B
jsch-0.1.42.jar		35 B
...		...

Project Settings

- Make a new project in eclipse
- Add these jars in it's build path

Exercise

- We will put a file in HDFS and try to read it's contents with JAVA APIs

```

Applications  Places  System  Tue Feb 11, 10:26 PM  training
training@localhost:usr/lib/hadoop-0.20-mapreduce/conf
File Edit View Search Terminal Help
[training@localhost conf]$ hadoop fs -mkdir /user/training/cluster
[training@localhost conf]$ hadoop fs -ls /user/training
Found 5 items
drwxr-xr-x - training supergroup      0 2014-02-11 22:24 /user/training/cluster
drwxr-xr-x - training supergroup      0 2014-02-09 19:14 /user/training/ext1
drwxr-xr-x - training supergroup      0 2014-02-09 19:55 /user/training/ext3
-rw-r--r-- 1 training supergroup    34 2014-02-10 20:26 /user/training/new_file
drwxr-xr-x - training supergroup      0 2014-02-10 20:15 /user/training/trial_dirac
[training@localhost conf]$ hadoop fs -ls /user/training/cluster
[training@localhost conf]$

```

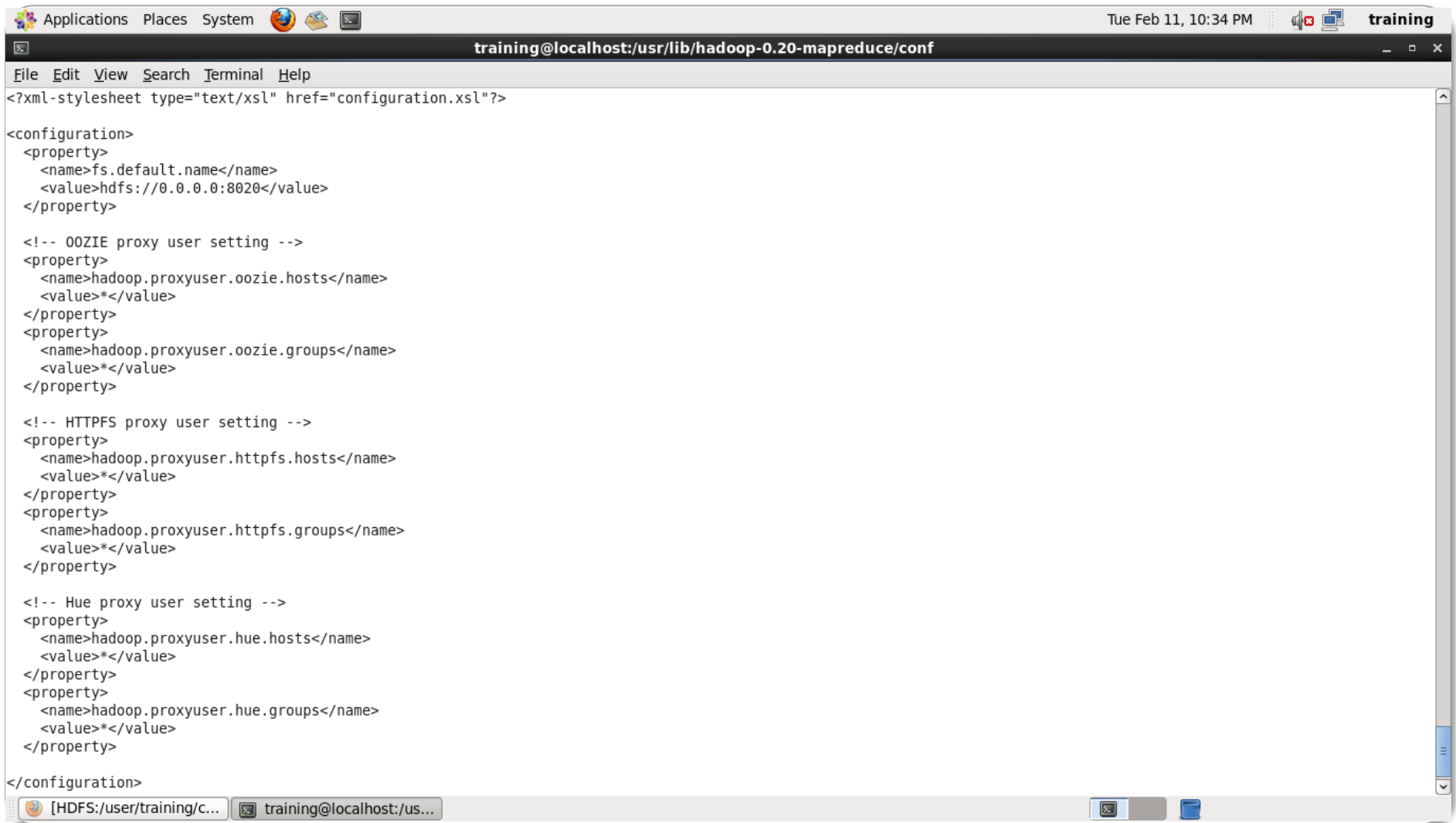
VM Environment

- The default configurations in vm are local host. With this configuration HADOOP doesn't allow external client connections.
- To ease the testing and our understanding we would like to connect the HADOOP from our local OS where eclipse is installed, so that we don't have to scp our jars or classes to vm.

Configurations Needed

- Change the NAMENODE/JOBTRACKER from local host to the IP of vm
- Files to be changed in /usr/lib/hadoop-0.20-mapreduce/conf are mapred-site.xml, hdfs-site.xml, core-site.xml

Editing Read Only File



The screenshot shows a terminal window titled "training@localhost:usr/lib/hadoop-0.20-mapreduce/conf". The window displays the contents of a configuration file, which is an XML document. The file is being edited, and the terminal shows the following XML content:

```
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>

<configuration>
  <property>
    <name>fs.default.name</name>
    <value>hdfs://0.0.0.0:8020</value>
  </property>

  <!-- OOZIE proxy user setting -->
  <property>
    <name>hadoop.proxyuser.oozie.hosts</name>
    <value>*</value>
  </property>
  <property>
    <name>hadoop.proxyuser.oozie.groups</name>
    <value>*</value>
  </property>

  <!-- HTTPFS proxy user setting -->
  <property>
    <name>hadoop.proxyuser.httpfs.hosts</name>
    <value>*</value>
  </property>
  <property>
    <name>hadoop.proxyuser.httpfs.groups</name>
    <value>*</value>
  </property>

  <!-- Hue proxy user setting -->
  <property>
    <name>hadoop.proxyuser.hue.hosts</name>
    <value>*</value>
  </property>
  <property>
    <name>hadoop.proxyuser.hue.groups</name>
    <value>*</value>
  </property>
</configuration>
```

The terminal window also shows the system menu (Applications, Places, System) and the date and time (Tue Feb 11, 10:34 PM). The window title bar indicates the file path: "training@localhost:usr/lib/hadoop-0.20-mapreduce/conf".

Confidential

- To edit the file you will have to do `sudo su` so that with root access you can change the file

```
[training@localhost conf]$ sudo su  
[root@localhost conf.pseudo.mr1]#
```

- You can now edit the file and change the localhost to IP of vm

```
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>  
  
<configuration>  
  <property>  
    <name>fs.default.name</name>  
    <value>hdfs://192.168.31.128:8020</value>  
  </property>
```

Changes to be Done in Local OS

- If you are running MR jobs/HDFS access from outside vm, host name entry needs to be done in the local OS as well, so that IP corresponding to the name can be pinged
- In running from windows C:\Windows\System32\drivers\etc\hosts, add the mapping
192.168.31.128 cloudera-vm


```

change.log x findImmediateCousin.java x black_jack.py.txt x pom.xml x settings.xml x hosts x
1 # Copyright (c) 1993-2009 Microsoft Corp.
2 #
3 # This is a sample HOSTS file used by Microsoft TCP/IP for Windows.
4 #
5 # This file contains the mappings of IP addresses to host names. Each
6 # entry should be kept on an individual line. The IP address should
7 # be placed in the first column followed by the corresponding host name.
8 # The IP address and the host name should be separated by at least one
9 # space.
10 #
11 # Additionally, comments (such as these) may be inserted on individual
12 # lines or following the machine name denoted by a '#' symbol.
13 #
14 # For example:
15 #
16 #      102.54.94.97      rhino.acme.com      # source server
17 #      38.25.63.10      x.acme.com          # x client host
18
19 # localhost name resolution is handled within DNS itself.
20 #   127.0.0.1      localhost
21 #   ::1            localhost
22 192.168.191.70 atlas41.guavus.com:q!
23
24 192.168.191.70 bs41.guavus.com
25 192.168.191.70 rge41.guavus.com
26 192.168.156.95 machine-156-95
27 192.168.87.139 cloudera-vm
28
29

```

Changes to be Done

- Change the fs.default.name to hdfs://<IP OF VM>:8020 in /etc/hadoop-0.20/conf/core-site.xml
- In mapred-site.xml mapred.job.tracker to <IP OF VM>:8021
- In /etc/hosts make sure that there is no entry of 127.0.0.1 and there is a hostname for your VM IP


File Edit View Search Terminal Help
[root@localhost etc]# cat hosts
192.168.31.128 cloudera-vm
#127.0.0.1 localhost.localdomain localhost
::1 localhost6.localdomain6 localhost6
[root@localhost etc]# █

Restart HADOOP Services

- To make your changes effective in hadoop you will have to restart namenode, data node and jobtracker

Use commands

- `sudo service hadoop-hdfs-namenode restart`
- `sudo service hadoop-hdfs-datanode restart`
- `sudo service hadoop-0.20-mapreduce-jobtracker restart`
- `sudo service hadoop-0.20-mapreduce-tasktracker restart`

Restart Name Node and Data Node

File Edit View Search Terminal Help

```
[training@localhost etc]$ sudo service hadoop-hdfs-namenode restart
Stopping Hadoop namenode:          [ OK ]
no namenode to stop
Starting Hadoop namenode:          [ OK ]
starting namenode, logging to /var/log/hadoop-hdfs/hadoop-hdfs-namenode-localhost.localdomain.out
[training@localhost etc]$
```

```
[training@localhost etc]$ sudo service hadoop-hdfs-datanode restart
Stopping Hadoop datanode:          [ OK ]
stopping datanode
Starting Hadoop datanode:          [ OK ]
starting datanode, logging to /var/log/hadoop-hdfs/hadoop-hdfs-datanode-localhost.localdomain.out
[training@localhost etc]$
```

Health Check

- After doing all the changes check if HDFS is accessible
- HADOOP dfs -ls /

Troubleshooting Guide

- You can always check logs of namenode, datanode to see if any errors have occurred while starting namenode/datanode or hadoop services in `/var/log/hadoop-0.20-mapreduce/logs` and `/usr/lib/hadoop-hdfs/logs`
- In case you get the error

```
cloudera@cloudera-vm:/etc/hadoop-0.20/conf.pseudo$ hadoop dfs -ls /  
13/12/08 05:44:57 INFO ipc.Client: Retrying connect to server: /192.168.87.138:8020. Already tried 0 time(s).  
13/12/08 05:44:58 INFO ipc.Client: Retrying connect to server: /192.168.87.138:8020. Already tried 1 time(s).  
13/12/08 05:44:59 INFO ipc.Client: Retrying connect to server: /192.168.87.138:8020. Already tried 2 time(s).  
13/12/08 05:45:00 INFO ipc.Client: Retrying connect to server: /192.168.87.138:8020. Already tried 3 time(s).  
13/12/08 05:45:01 INFO ipc.Client: Retrying connect to server: /192.168.87.138:8020. Already tried 4 time(s).  
13/12/08 05:45:02 INFO ipc.Client: Retrying connect to server: /192.168.87.138:8020. Already tried 5 time(s).
```

- Issue must be with your host mapping, please check `/etc/hosts` to see if there is any entry for localhost IP `127.0.0.1`. If it is there comment or delete that and restart your hdfs services

Distscp

- Tool for large inter/intra cluster copying.
- Can be used to copy large data from one hdfs cluster to another. Provided both are running on same HADOOP version
- HADOOP distscp hdfs://namenode1/foo hdfs://namenode2/foo2
- This will copy /foo directory in foo2
- This is implemented as MR job.(Will be covered in MR section)

Confidential

Java APIs

APIs

- <http://hadoop.apache.org/docs/current/api/org/apache/hadoop/fs/Path>
- **Path** (URI aUri)
 - Construct a path from a URI. This URI is the path of HDFS
- The file system can be obtained by:
 - <http://hadoop.apache.org/docs/current/api/org/apache/hadoop/fs/FileSystem.html>

```
import java.io.BufferedReader;
import java.io.InputStreamReader;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.FileSystem;
import org.apache.hadoop.fs.Path;

public class ReadFile {

    public static void main(String[] args) throws Exception {

        try{

            Path path = new Path("hdfs://cloudera-vm:8020/pristine/data");
            Configuration conf = new Configuration();
            FileSystem fs = FileSystem.get(path.toUri(), conf);
            BufferedReader br = new BufferedReader(new InputStreamReader(fs.open(path)));
            String line ;
            line = br.readLine();
            while(line!=null)
            {
                System.out.println(line);
                line = br.readLine();
            }

        }catch(Exception e)
        {
            System.out.println(e);
            e.printStackTrace();
        }
    }
}
```

HADOOP CLASSES FOR HDFS

HDFS PATH, Expects host name of hadoop machine. Port 8020 of namenode.

FILE SYSTEM HADOOP

Confidential

[illegible]

- Read from offset

- <http://hadoop.apache.org/docs/current/api/org/apache/hadoop/fs/FSDatInputStream.html>

```
import org.apache.hadoop.conf.Configuration;

public class Seek {

    public static void main(String[] args) throws Exception{

        Path path = new Path("hdfs://cloudera-vm:8020/pristine/data");
        Configuration conf = new Configuration();
        FileSystem fs = FileSystem.get(path.toUri(), conf);
        FSDataInputStream in = null;
        try{

            in = fs.open(path);
            IOUtils.copyBytes(in, System.out, 4096, false);
            in.seek(0);
            IOUtils.copyBytes(in, System.out, 4096, false);

        }finally{
            IOUtils.closeStream(in);
        }

    }
}
```

Seek to desired offset


```
import java.io.BufferedInputStream;

public class FileCopy {

    public static void main(String[] args) throws Exception { Path of File to be copied

        String localSrc = "C:\\\\Users\\jasleen.kaur\\Desktop\\hdfsconcepts.txt";
        Path path = new Path("hdfs://cloudera-vm:8020/jasleen/hdfsconcepts.txt");
                                                                    Path of hdfs

        Configuration conf = new Configuration();

        FileSystem fs = FileSystem.get(path.toUri(), conf);

        InputStream in = new BufferedInputStream(new FileInputStream(localSrc));
        OutputStream out = fs.create(path, true);
        IOUtils.copyBytes(in, out, 4096, true);

    }
}
```

```
cloudera@cloudera-vm: /etc/hadoop-0.20/conf.pseudo$  
cloudera@cloudera-vm:/etc/hadoop-0.20/conf.pseudo$ hadoop dfs -ls /jasleen/  
Found 1 items  
-rw-r--r--   3 jasleen.kaur supergroup      540 2013-12-08 04:44 /jasleen/hdfsconcepts.txt  
cloudera@cloudera-vm:/etc/hadoop-0.20/conf.pseudo$
```

Case Study Exercise

- STORAGE: Put the provided sample logs of airmobile in HDFS directory with name airmobile with java APIs.
- READ: Try to read the file using FS commands.
- The fields in the given file represent field following msidn= "<subscriber Id>", databytes = "Bytes Downloaded", city="<city">, sitevisited="<Site Visited">. For analysis write a Java program to find out subscribers and their corresponding downloaded bytes.

Next Agenda

- Next we will try to find out how the same Java program implemented in last exercise can be done with Map Reduce.

Thank you!

Contact:

EduPristine

702, Raaj Chambers, Old Nagardas Road, Andheri (E), Mumbai-400 069. INDIA

www.edupristine.com

Ph. +91 22 3215 6191