Hadoop Commands (2.4.0)

CSE Workshop **VM** • Lubuntu 14.04 Trusty Tahr

User names: `cse-user` (for general exercises in MATLAB, Python, R); `hduser` (for Hadoop)
Password: `cse-user1` (all users)

This VM has a dedicated Hadoop user account, a recommended practice to separate the Hadoop installation from other software applications and user accounts on the same machine. To log in for Hadoop exercises:

```
  cse-user@csevm$ sudo login hduser
  Enter password for sudo: cse-user1
  Enter password: cse-user1
  hduser@csevm$
```

FS Shell Commands

`hadoop fs <args>` *or* `hdfs dfs <args>`

All FS shell commands take paths as arguments in the format `scheme://authority/path`. For HDFS the scheme is `hdfs`, and for the local filesystem the scheme is `file`. `scheme` and `authority` are optional. An HDFS file or directory such as /parent/child can be specified as `hdfs://namenodehost/parent/child` or simply as `/parent/child`. Most commands in FS shell behave like corresponding Unix commands.

`cat`	Copies source path to `stdout`. `hadoop fs -cat /user/script.sh`
`copyFromLocal`	Like `put`, except source restricted to local file reference. `hadoop fs -copyFromLocal ./data/file01.csv /user/data`
`copyToLocal`	Like `get`, except that destination is restricted to local file reference. `hadoop fs -copyToLocal [-ignorecrc] [-crc] URI <localdst>`
`cp`	Copy files from source to destination. `hadoop fs -cp /user/oldfile /user/newfile`
`du`	Displays aggregate length of files contained in directory or length of file. `hadoop fs -du /user/data`
`expunge`	Empty the Trash. (See *HDFS Design* for more information.) `hadoop fs -expunge`
`get`	Copy files to the local file system. `hadoop fs -get /user/hdfsfile localfile`
`ls`	For file, returns stat on the file; for directory, returns list of direct children `hadoop fs -ls /user/file`
`mkdir`	Takes path as argument and creates directories. `hadoop fs -mkdir /user/userdir`
`mv`	Moves files from source to destination (but not across filesystems). `hadoop fs -mv /user/hadoop/file1 /user/hadoop/file2`
`put`	Copy single or multiple sources from local file system to destination filesystem. `hadoop fs -put localfile /user/hdfsfile`
`rm`	Delete files specified as args. Only deletes non empty directory and files. `hadoop fs -rm /user/emptydir`
`rmr`	Recursive version of delete. `hadoop fs -rmr /user/nonemptydir`
`tail`	Displays last kilobyte of the file to stdout. `hadoop fs -tail /user/data/01.dat`

`test -e`
`-z`
`-d`
	Check if file exists. (0 if true)
	Check if zero length file. (0 if true)
	Check if directory (return 1) or not.
	`hadoop fs -test -e filename`

Running Hadoop **Locally**

1. Format the filesystem:

```
$ hdfs namenode -format
```

2. Start NameNode daemon and DataNode daemon:

```
$ start-all.sh
```

3. Browse the web interface for the NameNode, available at `http://localhost:50070`.
4. Make the HDFS directories required to execute MapReduce jobs:

```
$ hdfs dfs -mkdir /user
$ hdfs dfs -mkdir /user/hduser
$ hdfs dfs -mkdir /user/hduser/input
```

5. Copy the input files into the distributed filesystem:

```
$ hdfs dfs -copyFromLocal $HADOOP_HOME/etc/hadoop /user/hduser/input
```

6. Run an example, one which here counts the number of instances of each string matching the regular expression:

```
$ hadoop jar $HADOOP_HOME/share/hadoop/mapreduce/hadoop-mapreduce-examples-2.4.0.jar
  grep /user/hduser/input/hadoop /user/hduser/output '<[a-z.]+>'
```

7. Examine the output files.
   a. Copy output files from the distributed filesystem to the local filesystem and examine them:

```
$ hdfs dfs -get output output
$ cat output/*
```

   b. View output files on the distributed filesystem:

```
$ hdfs dfs -cat output/*
```

8. When done, stop the daemons with:

```
$ stop-dfs.sh
```

Hadoop Daemon **Ports** (http://localhost:#####)

MAP/REDUCE			
50030	job tracker	50060	task tracker
HDFS			
8020 50070	name node	50090	secondary name node
50010 50020 50075	data nodes	50100 50105	backup/checkpoint node