

ISIT312 Big Data Management

SIM, Session 4, 2022

Exercise 1

Hadoop and HDFS

In this exercise, you will get familiar with using the Linux Shell and Zeppelin to interact with Hadoop.

DO NOT attempt to copy the Linux commands in this document to your working Terminal, because it is error-prone. Type those commands by yourself.

Laboratory Instructions.

First you must install download and install VirtualBox on your system.

You can download VirtualBox from here:

<https://www.virtualbox.org/wiki/Downloads>

You can find installation procedure here:

<https://documents.uow.edu.au/~jrg/115/cookbook/e1-1-A1-frame.html>

Connect to Moodle and to download ova image of virtual machine use a link

Image of Virtual Machine for ISIT312

available in WEB LINKS section.

When downloaded start VirtualBox and use option File->Import to import a file BigDataVM-07-SEP-2020.ova into VirtualBox.

After the importation is completed, a VM named BigDataVM-07-SEP-2020 appears in a column on the left-hand side of VirtualBox window. Use Settings option and later on System option and allow the virtual machine to use all available Base Memory on the "green side" of a scale. In Display option grant to the virtual machine all available Video Memory. Finally, in Network option, check (if necessary, and change) the Attached to option to NAT.

Use Start option to run BigDataVM-07-SEP-2020.

Note. There is no need for user name and password.

Note. Next time you use the VM a process of changing Settings does not need to be repeated.

(1) Start Shell and Zeppelin

Start a Shell window with Ctrl + Alt + T or use the second from bottom icon in a sidebar on the left-hand side of a screen.

You can use the Linux Shell to interact with Hadoop. However, it is recommended that you use Zeppelin, which provides a better interface. To start Zeppelin, enter in the Shell window:

```
$ZEPPELIN_HOME/bin/zeppelin-daemon.sh start
```

Then use the fourth from bottom icon in the sidebar to start the Firefox browser and use access the following link.

127.0.0.1:8080

You may need to refresh the browser until you see the following information:

The screenshot shows the Zeppelin web interface. At the top, there is a blue header bar with the Zeppelin logo and navigation links for 'Notebook' and 'Job'. Below the header, a large white area displays the text 'Welcome to Zeppelin!'. Underneath this, there is a brief introduction: 'Zeppelin is web-based notebook that enables interactive data analytics. You can make beautiful data-driven, interactive, collaborative document with SQL, code and even more!'. On the left side, there is a sidebar with sections for 'Notebook' (containing 'Import note', 'Create new note', 'Filter', and 'Zeppelin Tutorial' buttons), 'Help' (link to documentation), and 'Community' (text about contributing to Zeppelin). The main content area is currently empty, showing a large white space.

Then, use a link `Create new note` to create a new note and named it as, say, `Lab1`. At the moment you can use `sh` as a default interpreter. Do not create any folders and click at `Create` button to create a note.

A note comprises of many paragraphs. In the first line of each paragraph, you need to indicate the interpreter by inputting `%<interpreter>` where a metasymbol `<interpreter>` should be replaced with interpreter name. In this laboratory class, we use the Shell interpreter with command is `%sh`. Type

```
%sh  
pwd
```

inside a note window (see a picture below). We would like to process `pwd` command (print working directory) in the Shell window.

When you are ready to run your code in a Zeppelin paragraph, click the Run button (a small triangle next to `READY` text on the right -hand side of a note window) or use `Shift + Return` keyboard shortcut.

Zeppelin processes `pwd` command in the Shell window and displays a text:

```
/home/bigdata
```

You can also try to use the Markdown interpreter to write down some text.

```
%sh  
pwd  
  
/usr/local  
  
Took 0 sec. Last updated by anonymous at July 29 2019, 12:20:34 PM.
```

```
%md  
Write down your test here ....|
```

For example, type the following lines inside Zeppelin paragraph.

```
%md  
<b>Hello</b>, <i>Hello</i>, Hello !!!
```

Can you guess from the outcomes what does Markdown do ?

It is always possible top re-edit the contents of Zeppelin's paragraphs and re-run a new code many times. If you do not provide a name of interpreter, for example %sh or %md then Zeppelin uses a default interpreter that has been determined as %sh when we created a new note.

Now you can interact with Hadoop.

(2) Hadoop files and scripts

Have a look at what are contained in the \$HADOOP_HOME. Type and process the following lines one by one in the separate Zeppelin's paragraphs.

```
ls $HADOOP_HOME      # view the root directory  
ls $HADOOP_HOME/bin # view the bin directory  
ls $HADOOP_HOME/sbin # view the sbin directory
```

Note that # denotes a start of inline comment and ls is a Shell command that list the contents of a folder. In another paragraph try the following command to learn what does \$HADOOP_HOME mean.

```
echo $HADOOP_HOME
```

The bin and sbin folders contain scripts for Hadoop.

(3) Hadoop Initialization

Now you can start all Hadoop processes. First, start HDFSNameNode and DataNode processes.

```
$HADOOP_HOME/sbin/hadoop-daemon.sh start namenode  
$HADOOP_HOME/sbin/hadoop-daemon.sh start datanode
```

Next, start YARN ResourceManager and NodeManager.

```
$HADOOP_HOME/sbin/yarn-daemon.sh start resourcemanager  
$HADOOP_HOME/sbin/yarn-daemon.sh start nodemanager
```

Finally, start the Job History Server.

```
$HADOOP_HOME/sbin/mr-jobhistory-daemon.sh start historyserver
```

To view the running daemon processes the following command.

```
jps
```

The following Hadoop processes should be listed (note that the process numbers may be different).

```
2897 JobHistoryServer  
2993 Jps  
2386 NameNode  
2585 ResourceManager  
2654 NodeManager  
2447 DataNode
```

(4) HDFS Shell commands

Create a folder `myfolder` in HDFS process the following command.

```
$HADOOP_HOME/bin/hadoop fs -mkdir myfolder
```

To check if the folder has been created process the following command.

```
$HADOOP_HOME/bin/hadoop fs -ls
```

Copy a file from the local filesystem to HDFS. The following command copy all files with the `.txt` extension in `$HADOOP_HOME` to a folder `myfolder` in HDFS:

```
$HADOOP_HOME/bin/hadoop fs -put $HADOOP_HOME/*.txt myfolder
```

To verify the results, list all `*.txt` files in `$HADOOP_HOME`.

```
ls $HADOOP_HOME/*.txt
```

Next, list the files in `myfolder` folder in HDFS:

```
$HADOOP_HOME/bin/hadoop fs -ls myfolder
```

To view a file in HDFS process the following command.

```
$HADOOP_HOME/bin/hadoop fs -cat myfolder/README.txt
```

Copy a file from HDFS to the Desktop folder in a local filesystem process the following command.

```
$HADOOP_HOME/bin/hadoop fs -copyToLocal myfolder/README.txt /home/bigdata/Desktop
```

To verify the results, process the following command.

```
ls /home/bigdata/Desktop
```

To remove a file `README.txt` from HDFS process the following command.

```
$HADOOP_HOME/bin/hadoop fs -rm myfolder/README.txt
```

(5) HDFS user interface

Open another tab in the Firefox Web Browser, and use a link `localhost:50070`.

You will see `localhost:8020`. This is the location of the HDFS. It is specified in a configuration file named `core-site.xml`.

Check this file in a folder `$HADOOP_HOME/etc/hadoop`, that contains Hadoop's configuration files.

Process the following Shell command to view the contents of a file `core-site.xml` in Zeppelin paragraph.

```
cat $HADOOP_HOME/etc/hadoop/core-site.xml
```

Return to and browse the Web user interface, for example, you can see the location of the Datanode. Use the options Utilities and then Browser the file system. Check the `.txt` files uploaded to HDFS previously. Note that the root directory of bigdata is in the user directory.

To view all root folders in HDFS in Terminal, you can also enter the following command in a new Zeppelin paragraph.

```
$HADOOP_HOME/bin/hadoop fs -ls /
```

(6) HDFS's Java Interface

A Java program listed below retrieves the contents of a file in the HDFS. This program is equivalent to a command `hadoop fs -cat`. A source code of the program (`FileSystemCat.java`) is available in `exercise-1` folder and it is also provided below. Read and understand the source code.

```
// cc FileSystemCat
// Displays files from a Hadoop filesystem on standard output
// by using the FileSystem directly

import java.io.InputStream;
import java.net.URI;

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

// vv FileSystemCat
public class FileSystemCat {

    public static void main(String[] args) throws Exception {
        String uri = args[0];
        Configuration conf = new Configuration();
        FileSystem fs = FileSystem.get(URI.create(uri), conf);
        InputStream in = null;
        try {
            in = fs.open(new Path(uri));
            IOUtils.copyBytes(in, System.out, 4096, false);
        } finally {
            IOUtils.closeStream(in);
        }
    }
} // ^~ FileSystemCat
```

Now, create a file `FileSystemCat.java` with the source code and try to compile it. It is important that all commands must be processed in the same Zeppelin paragraph. First, use the following command to define environment variable `HADOOP_CLASSPATH`.

```
export HADOOP_CLASSPATH=$(($HADOOP_HOME/bin/hadoop classpath))
```

This environment variable points to all basic Hadoop libraries. Note that each time you open the Terminal and you want to use the environment variable then you must `export` it. To view the libraries, process in the same Zeppelin paragraph the following command.

```
echo $HADOOP_CLASSPATH
```

Now, make sure that the file `FileSystemCat.java` is in your current folder. Now, compile the program and create `.jar` file in the following way (still in the same Zeppelin paragraph).

```
javac -cp $HADOOP_CLASSPATH FileSystemCat.java  
jar cvf FileSystemCat.jar FileSystemCat*.class
```

The first above command moves to the current folder that contains the Java source (so that the compilation does not create any package namespace for the main class). The second command compiles the source code. The last command creates a `.jar` file that includes the Java class(es).

Now, you can run the `.jar` file by using the `hadoop` script and `jar` command (still in the same Zeppelin paragraph).

```
$HADOOP_HOME/bin/hadoop jar /home/bigdata/Desktop/FileSystemCat.jar FileSystemCat myfolder/LICENSE.txt
```

Check whether the uploaded file is same as the local file.

(7) Shut down Hadoop

When finishing your practice with Hadoop, it is good practice to terminate the Hadoop daemons before turning off the VM.

Use the following command to terminate the Hadoop daemons.

```
$HADOOP_HOME/sbin/hadoop-daemon.sh stop namenode  
$HADOOP_HOME/sbin/hadoop-daemon.sh stop datanode  
$HADOOP_HOME/sbin/yarn-daemon.sh stop resourcemanager  
$HADOOP_HOME/sbin/yarn-daemon.sh stop nodemanager  
$HADOOP_HOME/sbin/mr-jobhistory-daemon.sh stop historyserver
```

(8) Make a typescript of information in Terminal

If you work in Shell but not Zeppelin, you can use the `script` command to make a typescript of everything printed in your Terminal.

```
script a-file-name-you-want-to-save-the-typescript-to.txt  
<work with your Terminal..>  
exit
```

Check the contents of `a-file-name-you-want-to-save-the-typescript-to.txt`.

It is good idea to export Zeppelin note, such that you can import it before the next exercise and reuse the commands processed up to now.

End of Exercise 1