Hadoop 2.7.3 Setup Walkthrough

This document provides information about working with Hadoop 2.7.3.

1 Setting Up Configuration Files

- Download and extract the configuration files from the course website (Link: http://www.cs.colostate.edu/~cs455/hadoop-conf.tar.gz).

```
wget http://www.cs.colostate.edu/~cs455/hadoop-conf.tar.gz

tar –xvzf hadoop-conf.tar.gz
```

- Modify your configuration files as per the following instructions.
    - masters
        - Specify the secondary name node for your HDFS cluster.
    - slaves
        - List of nodes used as data nodes for the HDFS cluster as well as Yarn resources. It is recommended to use between 10-15 nodes.
    - core-site.xml
        - Update the name node hostname and port. Name node will start on the machine where you run '`start-dfs.sh`' command as explained later.
    - hdfs-site.xml
        - Use unique port numbers where the place holder 'PORT' appears.
        - Update the secondary name node host (hostname mentioned in the 'masters' file) and a unique port for its web console.
    - mapred-site.xml
        - Replace 'PORT' with a unique port number.
    - yarn-site.xml
        - Update the RESOURCE-MANAGER-HOST with the hostname of the node you are planning to run the Yarn resource manager.
        - Use unique values for ports marked with the 'PORT' place holder.
    - hadoo-env.sh and yarn-env.sh
        - It is not necessary to modify these scripts

Hadoop will be using the local disks of the machines you specify in your 'masters' and 'slaves' files. The location is /s/${HOSTNAME}/a/nobackup/cs455/${USER} where ${HOSTNAME} is the machine name (e.g.: denver) and ${USER} is your CS department login (e.g.: johndoe).

The log files will also be available in the 'tmp' directory created in the local disks. The location is /s/${HOSTNAME}/a/tmp/${USER}. There will be separate directories for Hadoop logs and Yarn logs.

2 Setting Up The Environment

- Determine which shell you are using

```
echo $SHELL
```

- For tcsh:
    - Add the following lines into your ~/.cshrc file.

```
setenv HADOOP_HOME /usr/local/hadoop
setenv HADOOP_CONF_DIR <path to your Hadoop conf dir>
```

```
setenv JAVA_HOME /usr/local/jdk1.8.0_51
```

- For bash:
  - Copy the environment variables for Bash (from the course wiki) into your ~/.bashrc file

```
export HADOOP_HOME=/usr/local/hadoop
export HADOOP_CONF_DIR=<path to your Hadoop conf dir>
export JAVA_HOME=/usr/local/jdk1.8.0_51
```

- Make sure to update the HADOOP_CONF_DIR to point to the directory with configuration files created in section 1.
- Close any open terminals, resume with a fresh terminal or source the .*rc file.
- **Make sure to use jdk 1.8.0_51 (instead of 1.8.0_60) to avoid containers getting killed due to virtual memory issues.**
- **Also make sure that you have setup SSH keys to log into CS department machines. Instructions to setup a SSH key pair are available in http://www.cs.colostate.edu/~info/faq.html - 4.08**

3 Additional Notes

- For additional help, visit http://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-common/ClusterSetup.html

4 Selecting Ports

You need to select a set of available ports from the non-privileged port range. Do not select any ports between 56,000 and 57,000. These are dedicated for the shared HDFS cluster.

5 Running HDFS

5.1 First Time Configuration

- Log into your namenode
  - You will have specified which machine this is in your core-site.xml file:
    - ```
      <property>
      <name>fs.default.name</name>
      <value>hdfs://host:port</value>
      </property>
      </configuration>
      ```
- Format your namenode

```
$HADOOP_HOME/bin/hdfs namenode -format
```

5.2 Starting HDFS

- Log into your namenode
- Run the start script:

```
$HADOOP_HOME/sbin/start-dfs.sh
```

- Check the web portal to make sure that HDFS is running:
  - http://<namenode>:<port given for the property `dfs.namenode.http-address` >

5.3 Stopping HDFS

- Log into your namenode
- Run the stop script:

```
$HADOOP_HOME/sbin/stop-dfs.sh
```
```

6 Running Yarn

6.1 Starting Yarn

- Log into your resource manager
- Run the start script:

```
$HADOOP_HOME/sbin/start-yarn.sh
```

- Check the web portal to make sure yarn is running:
  - http://<resourcemanager>:<port given for the property `yarn.resourcemanager.webapp.address`>

6.2 Stopping Yarn

- Log into your resource manager
- Run the stop script:

```
$HADOOP_HOME/sbin/stop-yarn.sh
```

7 Running a Job Locally

7.1 Setup

- Open mapred-site.xml
  - Change the value of property `mapreduce.framework.name` to `local`
- To be safe, restart HDFS and ensure yarn is stopped

7.2 Running the Job

- From any node:

```
$HADOOP_HOME/bin/hadoop jar your.jar yourClass args
```

8 Running a Job in Yarn

8.1 Setup

- Open mapred-site.xml
  - Change the value of property `mapreduce.framework.name` to `yarn`
- To be safe, restart HDFS and yarn

8.2 Running the Job

- From any node:

```
$HADOOP_HOME/bin/hadoop jar your.jar yourClass args
```

9 Accessing Shared Data Sets

We are using ViewFS based federation for sharing data sets due space constraints. Your MapReduce programs will deal with two name nodes in this setup.

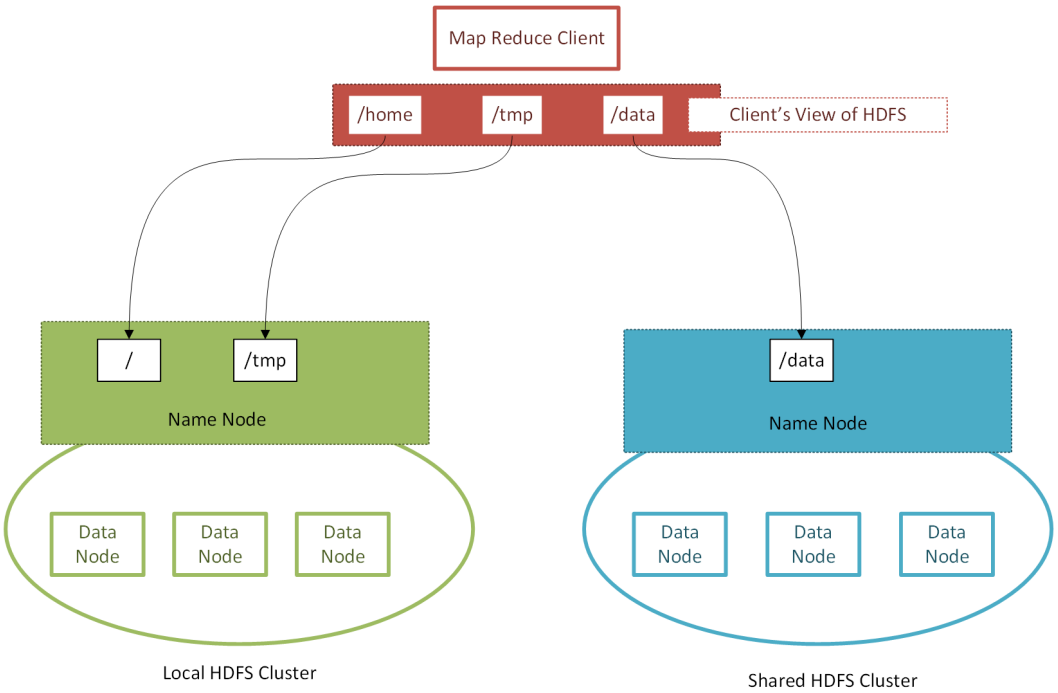
1. **Name node of your own HDFS cluster**. This name node is used for writing the output generated by your program. We will refer to this as the local HDFS.
2. **Name node of the shared HDFS cluster**. This name node hosts inputs for your data and it is made read only. **You should not attempt to write to this cluster. This is only for staging input data.**

Datasets are hosted in `/data` directory of shared file system. You will get r+x permission for this directory. This will allow your MapReduce program to traverse inside this directory and read files.

ViewFS based federation is conceptually similar to mounts in Linux file systems. Different directories visible to the client are mounted to different locations in namespaces of name nodes. This provides a seamless view to the client as if it is accessing a single file system. There are three mounts created in the core-site.xml of the client's end.

Mount Point	Name Node	Directory in Name Node
/home	Local	/
/tmp	Local	/tmp
/data	Shared	/data

Following figure provides a graphical representation of these mount points.



For example, if you want to read the main dataset, the input file path should be provided as `/data/main` which points to the `/data/main` in the shared HDFS. If you write the output to `/home/output-1`, then it will be written `/output-1` of the local HDFS.

Running a MapReduce Client

Usually a client uses the same Hadoop configuration used for creating the cluster when running MapReduce programs. In this setup, the Hadoop configuration for a client is different mainly due to the mount points discussed above. Also your MapReduce program will be running on the MapReduce runtime based on the shared HDFS. In other words, your local HDFS cluster will be used only for storing the output. You will read input and run your MapReduce programs on the shared Hadoop cluster. This is to ensure the data locality for input data.

Follow the following steps when running a MapReduce program.

1. Download the client-config.tar.gz directory from http://www.cs.colostate.edu/~cs455/client-config.tar.gz.
2. Extract the tarball.
3. This configuration directory contains 3 files. You only need to update the core-site.xml file. Other files should not be changed. Update the NAMENODE_HOST and NAMENODE_PORT information in the core-site.xml with hostname and port of the name node of your local cluster. You can find this information in the core-site.xml of your local HDFS cluster configuration. There will be an entry with 'augusta' as the name node hostname in the core-site.xml file.. Do not change this entry. It points to the name node of the shared cluster.
4. Now you need to update the environment variable 'HADOOP_CONF_DIR'. This has already being set in your .*rc file of your shell (.bashrc for Bash and .rcsh for Tcsh). Do not update the environment variable set in the .*rc file, because you are running your local Hadoop cluster based on that parameter. Instead, just overwrite this environment variable for the current shell instance.

   For tcsh;

   ```
   setenv HADOOP_CONF_DIR "/path/to/extracted-client-config"
   ```

   For Bash;

   ```
   export HADOOP_CONF_DIR=/path/to/extracted-client-config
   ```

   This overwritten value is valid only for that particular instance of the shell. You need to overwrite this environment variable every time you spawn a new shell to run a MapReduce program.

5. Now you can run your Hadoop program as usual. Make sure to include the paths using mount points.

   For example:
   Suppose that you need to run your census data analysis program. The output should be written to `/census/output` directory in your local cluster. Let's assume this job only takes two parameters, the input path and the output path respectively.

   ```
   $HADOOP_HOME/bin/hadoop jar airline-analysis.jar
   cs555.hadoop.airline.AnalysisJob /data/census /home/census/output
   ```