INF729 Project: Setting up a Hadoop cluster

Sarah Boutigny, Guillaume Canat, Quang-Vinh Julien Ta - MS BGD 2021-2022

Summary

We did not encounter too many difficulties setting up the Hadoop cluster on 4 machines, after reading the documentation and a few tutorials (references below). However, during the configuration of Hbase and Hive we had an issue because of different versions of java on 2 machines (java 11). After setting all machines to java 8, we did not have any problems. Other small errors encountered are documented below.

Once our clustering was up and running, we ran some small MapReduce examples provided on the Apache Hadoop site, namely WordCount.jar. Later, with Spark, we ran a similar example using a python script to count words in a Sherlock Holmes book.

We also tried to run a more ambitious example using the fr-wiki database, but could not succeed in copying this 38Gb database on our cluster.

Configuring Hadoop

sources: - Apache[1] - Linode[2] - Medium[3]

"For a small cluster (on the order of 10 nodes), it is usually acceptable to run the namenode and the resource manager (yarn) on a single master machine" *source: Hadoop The Definitive Guide (T. White 2015)*

Need to decide which machine will be master (namenode and resource manager) and which will be workers (datanodes). We decide `tp-hadoop-25` will be master.

Install java, shh, pdsh, net-tools

```
sudo apt-get install default-jre ssh pdsh
```

Download hadoop, extract and rename the directory

```
wget https://dlcdn.apache.org/hadoop/common/stable/hadoop-3.3.1.tar.gz
tar zxf hadoop-3.3.1.tar.gz
mv hadoop-3.3.1 hadoop
```

Distribute authentication key-pairs

generate ssh key on master node (without passphrase):

```
ssh-keygen -t rsa
```

copy content of `.ssh/id_rsa.pub` and paste it in `.ssh/authorized_keys` on each non-master node

Configure master node

Edit `~/hadoop/etc/hadoop/hadoop-env.sh`

```
export JAVA_HOME=/usr/lib/jvm/default-java
```

[1] https://hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-common/ClusterSetup.html
[2] https://www.linode.com/docs/guides/how-to-install-and-set-up-hadoop-cluster/
[3] https://medium.com/@jootorres_11979/how-to-set-up-a-hadoop-3-2-1-multi-node-cluster-on-ubuntu-18-04-2-nodes-567ca44a3b12

- set namenode location File: `~/hadoop/etc/hadoop/core-site.xml`

```
<configuration>
        <property>
            <name>fs.default.name</name>
            <value>hdfs://tp-hadoop-25:9000</value>
        </property>
    </configuration>
```

- Set path for HDFS File: `~/hadoop/etc/hadoop/hdfs-site.xml`

```
<configuration>
    <property>
            <name>dfs.namenode.name.dir</name>
            <value>file:///home/ubuntu/data/namenode</value>
    </property>

    <property>
            <name>dfs.datanode.data.dir</name>
            <value>file:///home/ubuntu/data/datanode</value>
    </property>

    <property>
            <name>dfs.replication</name>
            <value>3</value>
    </property>
</configuration>
```

- Configure workers File: `~/hadoop/etc/hadoop/workers`

```
tp-hadoop-25
tp-hadoop-33
tp-hadoop-35
tp-hadoop-36
```

Duplicate config on each node

```
cd ~/

cat ~/hadoop/etc/hadoop/workers | while read worker; do
    scp hadoop-3.3.1.tar.gz $worker:~/
    ssh $worker
    tar -xzf hadoop-3.3.1.tar.gz
    mv hadoop-3.3.1 hadoop
    exit
    scp ~/hadoop/etc/hadoop/* $worker:~/hadoop/etc/hadoop/;
done
```

Format HDFS

on node master:

```
hdfs namenode -format
```

Start HDFS

```
start-all.sh
```

Put some data to HDFS

```
hadoop fs -mkdir -p /user/ubuntu
hadoop fs -mkdir books
cd ~/
wget -O alice.txt https://www.gutenberg.org/files/11/11-0.txt
wget -O holmes.txt https://www.gutenberg.org/files/1661/1661-0.txt
wget -O frankenstein.txt https://www.gutenberg.org/files/84/84-0.txt

hadoop fs -put alice.txt holmes.txt frankenstein.txt books
```

Monitoring the cluster

In the command line:

```
hdfs dfsadmin -report
```

With the web interface (only these 2 urls seem to work):

```
http://137.194.211.146/tp-hadoop-25/9870/dfshealth.html#tab-overview
http://137.194.211.146/tp-hadoop-25/8088/cluster/nodes
```

Running an example

source: MapReduce Tutorial[4]

create the WordCount.java file from the tutorial page

```
# install jdk to be able to run the jar command
sudo apt install default-jdk

export JAVA_HOME=/usr/lib/jvm/default-java
export PATH=${JAVA_HOME}/bin:${PATH}
export HADOOP_CLASSPATH=${JAVA_HOME}/lib/tools.jar

bin/hadoop com.sun.tools.javac.Main WordCount.java
jar cf wc.jar WordCount*.class
```

The sample text file `alice.txt` is located in `/user/ubuntu/books` on the cluster. We run the application with the following command

```
# run the application
bin/hadoop jar wc.jar WordCount /user/ubuntu/books /user/ubuntu/output
# inspect the output
bin/hadoop fs -head /user/ubuntu/output/part-r-00000
```

Zookeeper

Zookeeper configuration and startup

[4] https://hadoop.apache.org/docs/stable/hadoop-mapreduce-client/hadoop-mapreduce-client-core/MapReduceTutorial.html

Zookeeper tutorial[5]

```
wget https://downloads.apache.org/zookeeper/stable/apache-zookeeper-3.6.3-bin.tar.gz
tar zxf apache-zookeeper-3.6.3-bin.tar.gz
```

Edit ~/apache-zookeeper-3.6.3-bin/conf/zoo.cfg

```
tickTime=2000
initLimit=5
syncLimit=2
dataDir=/home/ubuntu/apache-zookeeper-3.6.3-bin/data
clientPort=2181
server.1=tp-hadoop-25:2888:3888
server.2=tp-hadoop-36:2888:3888
server.3=tp-hadoop-35:2888:3888
server.4=tp-hadoop-33:2888:3888
```

Edit `~/apache-zookeeper-3.6.3/data/myid` and set the server number (for example set it to "1" for Master)

Start zookeeper: `zkServer.sh start`

Running simple examples

We can run some examples based on the Apache Zookeeper documentation: Zookeeper examples[6]

HBase

File : conf/hbase-env.sh

```
export JAVA_HOME=/usr/lib/jvm/default-java
```

File : `conf/hbase-site.xml`

```
<configuration>
<property>
  <name>hbase.cluster.distributed</name>
  <value>true</value>
</property>
<property>
  <name>hbase.rootdir</name>
  <value>hdfs://tp-hadoop-25:9000/hbase</value>
</property>
  <property>
    <name>hbase.zookeeper.quorum</name>
    <value>tp-hadoop-25,tp-hadoop-33,tp-hadoop-35,tp-hadoop-36</value>
  </property>
  <property>
    <name>hbase.zookeeper.property.dataDir</name>
    <value>/home/ubuntu/apache-zookeeper-3.6.3-bin</value>
  </property>
</configuration>
```

Start HBase : `bin/start-hbase.sh`

[5] https://www.digitalocean.com/community/tutorials/how-to-install-and-configure-an-apache-zookeeper-cluster-on-ubuntu-18-04
[6] https://zookeeper.apache.org/doc/r3.4.13/zookeeperTutorial.html

Hive

Download Hive

```
wget https://dlcdn.apache.org/hive/hive-3.1.2/apache-hive-3.1.2-bin.tar.gz
```

Extract the archive

edit `.bashrc` to add these lines:

```
export HADOOP_HOME=/home/ubuntu/hadoop-3.3.1
export HIVE_HOME=/home/ubuntu/apache-hive-3.1.2-bin
export PATH=$PATH:$HIVE_HOME/bin
```

Edit `hadoop-3.3.1/etc/hadoop/core-site.xml`:

```
<configuration>
  <property>
    <name>hadoop.proxyuser.ubuntu.groups</name>
    <value>*</value>
  </property>
  <property>
    <name>hadoop.proxyuser.ubuntu.hosts</name>
    <value>*</value>
  </property>
  <property>
    <name>hadoop.proxyuser.server.hosts</name>
    <value>*</value>
  </property>
  <property>
    <name>hadoop.proxyuser.server.groups</name>
    <value>*</value>
  </property>
</configuration>
```

Copy default config to `hive-site.xml`:

```
cp hive-default.xml.template hive-site.xml
```

Add the following lines to the `hive-site.xml` file, to fix the "unable to impersonate error":

```
<property>
  <name>hive.server2.enable.doAs</name>
  <value>false</value>
</property>
```

Another issue we had to fix: `java.net.URISyntaxException: Relative path in absolute URI` We add this to the `hive-site.xml`

```
<property>
  <name>system:java.io.tmpdir</name>
  <value>/tmp/hive/java</value>
</property>
<property>
  <name>system:user.name</name>
  <value>${user.name}</value>
```

```
</property>
```

Create directories for hive in hdfs:

```
hadoop fs -mkdir /tmp
hadoop fs -mkdir /ubuntu
hadoop fs -mkdir /ubuntu/hive
hadoop fs -mkdir /ubuntu/hive/warehouse
hadoop fs -chmod g+w /tmp
hadoop fs -chmod g+w /ubuntu/hive/warehouse
```

Run the schematool command below as an initialization step. For example, we can use "derby" as db type.

```
bin/schematool -dbType derby -initSchema
```

Restart Hadoop, Zookeeper and HBase. Now we can start Hive.

```
bin/hiveserver2
```

In another screen, we can launch the beeline command shell:

```
bin/beeline -n ubuntu -u jdbc:hive2://localhost:10000
```

Verify installation by typing `show databases;` in the beeline command shell

Spark

Download Apache Spark

```
wget https://www.apache.org/dyn/closer.lua/spark/spark-3.2.0/spark-3.2.0-bin-hadoop3.2.tgz
```

Extract the archive and cwd into the directory

```
tar zxf spark-3.2.0-bin-hadoop3.2.tgz
cd spark-3.2.0-bin
```

We can test our cluster setup with a simple wordcount on the holmes.txt that we uploaded earlier on the HDFS

```
./bin/spark-submit --class org.apache.spark.examples.SparkPi --master yarn
--deploy-mode cluster --supervise --executor-memory 5G --total-executor-cores 100
/home/ubuntu/spark-3.2.0-bin-hadoop3.2/spark_wordcount.py /ubuntu/holmes.txt
```

For the above command to work we need to export HADOOP_CONF_DIR, to do that we add the following line in `.bashrc`:

```
export HADOOP_CONF_DIR=/home/ubuntu/hadoop-3.3.1/etc/hadoop/
```

The content of the wordcount script is as follows:

```
import sys
from operator import add
from pyspark import SparkContext

if __name__ == "__main__":
    if len(sys.argv) != 2:
        print("Usage: wordcount <file>", file=sys.stderr)
        exit(-1)
```

```
    sc = SparkContext(appName="PythonWordCount")
    lines = sc.textFile(sys.argv[1], 10)
    counts = lines.flatMap(lambda x: x.split(' ')).map(lambda x: (x, 1)).reduceByKey(add)
    output = counts.collect()
    for (word, count) in output:
        with open('output.txt', 'w+') as f:
            f.write("%s: %i" % (word, count))
            print(f'{word} : {count}')
    sc.stop()
```

To be able to debug more easily what is going on in the cluster we modify the yarn config to aggregate logs on the master node. We add the following property to `yarn-site.xml`

```
<property>
  <name>yarn.log-aggregation-enable</name>
  <value>true</value>
</property>
```

Now we can run the following command to display logs of our application

```
bin/hdfs dfs -cat
/tmp/logs/ubuntu/bucket-logs-tfile/0001/application_1634911618427_0001/tp-hadoop-36_41289
```

Project

We download the french wikimedia data to the hadoop bridge `tp-bridge-2`:

```
wget https://dumps.wikimedia.org/frwiki/20211101/frwiki-20211101-pages-meta-current.xml.bz2
bzip2 -d frwiki-20211101-pages-meta-current.xml.bz2
```

Then we can send it to our hdfs cluster with this command:

```
bin/hdfs dfs -fs tp-hadoop-25:9000 -put ../frwiki-20211101-pages-meta-current.xml /user/ubuntu/
```

We get the following error, even after several attemps where we reduce the replication factor to 2, and we manually remove all data in data/namenode, data/datanode on all machines and reformat HDFS.

```
2021-11-09 11:56:40,998 WARN hdfs.DataStreamer: DataStreamer Exception
org.apache.hadoop.ipc.RemoteException(org.apache.hadoop.hdfs.server.namenode.SafeModeException):
Cannot add block to /user/ubuntu/frwiki-20211101-pages-meta-current.xml._COPYING_.
Name node is in safe mode.
Resources are low on NN. Please add or free up more resourcesthen turn off safe mode manually.
NOTE:  If you turn off safe mode before adding resources, the NN will immediately return to
safe mode. Use "hdfs dfsadmin -safemode leave" to turn safe mode off.
NamenodeHostName:tp-hadoop-25
```