Deploying and configuring *Yarn* on a single node

Shadi Ibrahim

March 3rd, 2017

Exercise 1: Installing Yarn platform on **Single** machine

The goal of this exercise is to learn how to set up and configure a single-node Yarn installation so that you can quickly perform simple operations using Hadoop MapReduce.

Question 1.1

Required software for Yarn on Linux include:

- Java must be installed (check and install in needed).
  To install java use.

```
$sudo apt-get update
$sudo apt-get install default-jdk
```

- ssh must be installed and sshd must be running to use the Hadoop scripts that manage remote Hadoop daemons.

You can check if the ssh daemon runs using the ps command:

```
$ps aux | grep sshd
```

If needed, you can install ssh as follows:

```
$sudo apt-get install ssh
```

To avoid typing your password each time you start the Hadoop daemons, you should create a couple of `ssh` keys thanks to the `ssh-keygen` command (if not already done so), then add the public key to the authorized keys.

```
ssh-keygen -t dsa (type ENTER for all default choices proposed)
cat $HOME/.ssh/id_dsa.pub >> $HOME/.ssh/authorized_keys
```

Check the success of this step by running:
`ssh localhost`.
When running this command, you should not be prompted to type any password.

Question 1.2

Now copy hadoop-2.7.3.tar to your VM to do so use the following command:

```
wget http://people.irisa.fr/Shadi.Ibrahim/hadoop-2.7.3.tar
```

Before starting the Yarn platform, edit its configuration files located in the `etc/hadoop/` folder. As you are using your own machine, you must configure Hadoop as follows:

- In masters and slaves files (used to automate `ssh` connections) specify the name of your machine: `localhost`
- In `etc/hadoop/hadoop-env.sh` set the JAVA_HOME variable consistently with your environment, for instance:

```
Change
# export JAVA_HOME=/usr/lib/j2sdk1.5-sun
To
export JAVA_HOME=/usr/lib/jvm/java-7-openjdk-amd64
```

- In `etc/hadoop/core-site.xml` add:

```
<configuration>
  <property>
    <name>fs.default.name</name>
      <value>hdfs://localhost:9000</value>
    </property>
</configuration>
```

This indicates the name of the machine and the associated network port on which the NameNode process will run. You should check that the 9000 port on your machine is not already in use:
`netstat -a | grep tcp | grep LISTEN | grep 9000`

- In `etc/hadoop/hdfs-site.xml` add:

```
<configuration>
  <property>
    <name>dfs.replication</name>
      <value>1</value>
    </property>
</configuration>
```

- In `etc/hadoop/mapred-site.xml` add:

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

This indicates the name of the machine and the associated network port on which the JobTracker process will run. You can check that the 9001 port on your machine is not already in use:
`netstat -a | grep tcp | grep LISTEN | grep 9001`

- In `etc/hadoop/yarn-site.xml` add:

```
<configuration>
   <property>
      <name>yarn.nodemanager.aux-services</name>
         <value>mapreduce_shuffle</value>
      </property>
</configuration>
```

Question 1.3
We will now start the platform, and start all daemons:

- Format a new distributed-filesystem:
  `$ bin/hadoop namenode -format`
- Start the HDFS daemons:
  `$ sbin/start-dfs.sh`
- The hadoop daemon log output is written to the `HADOOP DIRECTORY PATH/logs`. Check them
- A nice command for checking if the expected Hadoop processes are running is `jps`. Try it when the HDFS processes are running!
- Start the ResourceManager daemon and NodeManager daemons:
  `$ sbin/start-yarn.sh`

Question 1.4
To Browse the web interface for the NameNode and the JobTracker; by default they are available at:

- NameNode - http://localhost:50070/
- ResourceManager - http://localhost:8088/

NOTE: you may need to install a lightweight browser. You can install “elinks". *apt-get install elinks.*

Exercise 2: Running your first MapReduce program

The goal of this exercise is to execute two MapReduce examples, typically used for benchmarking, which come with the default Yarn distribution.

Question 2.1

Run the wordcount example:

- `bin/hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-examples-*jar wordcount input output`

where input is the directory containing the input files (e.g., loremIpsum-75) while the output is the directory that will be created to store the results.

Use the 75 MB data set and check the output. Then, inspect the log files generated by this job.

Question 2.2

Run the Pi estimator:

- `bin/hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-examples-*jar pi maps samples_per_map`

where maps represents the number of map tasks used and *samples_per_map* the number of points processed by each map task (i.e. the number of reducers).

Run the Pi estimator with 20 maps and 10000 *samples_per_map*. Check the log files.

Exercise 3: Checking the Job history

In order to be able to check the history of the jobs, you need to start job history server:

- `sbin/mr-jobhistory-daemon.sh -config etc/hadoop/ start historyserver`

- In `etc/hadoop/yarn-site.xml` add:

```
<configuration>
   <property>
      <name>yarn.log-aggregation-enable</name>
         <value>true</value>
      </property>
   <property>
      <name>yarn.nodemanager.remote-app-log-dir</name>
         <value>/app-logs</value>
      </property>
   <property>
      <name>yarn.nodemanager.remote-app-log-dir-suffix</name>
         <value>logs</value>
      </property>
   <property>
      <name>yarn.log.server.url</name>
         <value>http://localhost:8188/jobhistory/logs</value>
      </property>
</configuration>
```