Experience in Big Data System

AMS 560 Group18

Linna Hu, Hongyu Huang, Jiashuo Sun

**1. Background**

Nowadays, Big Data plays a popular role both in industrial and academic fields. Hadoop is developed as a solution to Big Data. It provides reliable, scalable, fault-tolerance and efficient service for large scale data processing based on HDFS and MapReduce.

HDFS stands for the Hadoop distributed file system and provides the distributed storage for the system. And MapReduce provides the distributed processing for Hadoop. However, MapReduce is not suitable for all classes of applications. An alternative to overcome the limitation of Hadoop is new in-memory runtime systems such as Spark, that is designed to support applications reuse a working set of data across multiple parallel operations . The weakness of Spark is that the performance is restricted by the memory.

The purpose of this assignment is to have more experiences on cloud computing systems. We try to use just 2 workloads to run the Hibench on Hadoop and Spark. Then we are supposed to get reports from Hibench, thus we can do the performance comparison between Hadoopbench and Sparkbench.

**2. The methodology**

To understand the big data system, we selected to set up an environment which includes three nodes in a cluster, Hadoop, Spark, Tez and Hibench. The main work during this research includes:

**2.1**  We registered a new account and got our SSH key address using linux system. Then we started our new cluster on cloudlab.us and we set up 2 other nodes in the cluster. We have to make sure each node could switch to another one without any problem and any password when we doing the work.

**2.2** We set up Hadoop. Before setting up it, we should download Yarn 2.7.2 and set up HDFS,Yarn to start the Hadoop.

**2.3**  Once we start HDFS and Yarn, we should have a test on them respectively. Thus we can make sure if our environment has been set up successfully.

**2.4** Then we set up MapReduce and Spark on Yarn, which is the most important step to complete the comparison. Once MapReduce was set up, we run the workcount job on big text file to test our installation process. Then we uploaded the file on hdfs.

**2.5**  We set up Apache Tez on Yarn. We built Tez and uploaded it on hdfs then tested it.

**2.6** Once Hadoop Yarn has been already set up. We are supposed to set up Spark on it. We tested it by submitting test job when we finished the installation.

**2.7**  We set up Hibench. We should configured Hibench then to run Hadoopbench. Then we run the workcount job on it. Thus we can tell if it has been set up successfully.

**3. Setup a new cluster on Cloudlab**

**3.1 Create an SSH key and a cluster**

Once we have an account on cloudlab.us, we are supposed to get ssh key to upload on the website to start our new experiment to set up a cluster.

*ssh-keygen -t rsa -b 4096 -C "*[*your\_email@example.com*](mailto:your_email@example.com)*”*

**3.2 SSH to a node in a cluster**

In the cloudlab.us, we can check our view list of our cluster to see our ssh key for each node.

*ssh -p 22 @ms0826.utah.cloudlab.us*

*ssh -p 22 @ms0824.utah.cloudlab.us*

*ssh -p 22 @ms0830.utah.cloudlab.us*

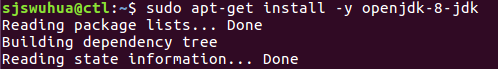
We can just use this command on our local host node, we can switch to the cluster.

**4. Setup Apache Hadoop Yarn**

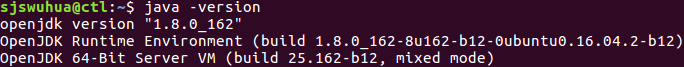
**4.1 Deploy java environment**

As prerequisites of setup Apache Hadoop Yarn, we need to need install JDK first.

*sudo apt-get install -y openjdk-8-jdk*

**

*java -version*



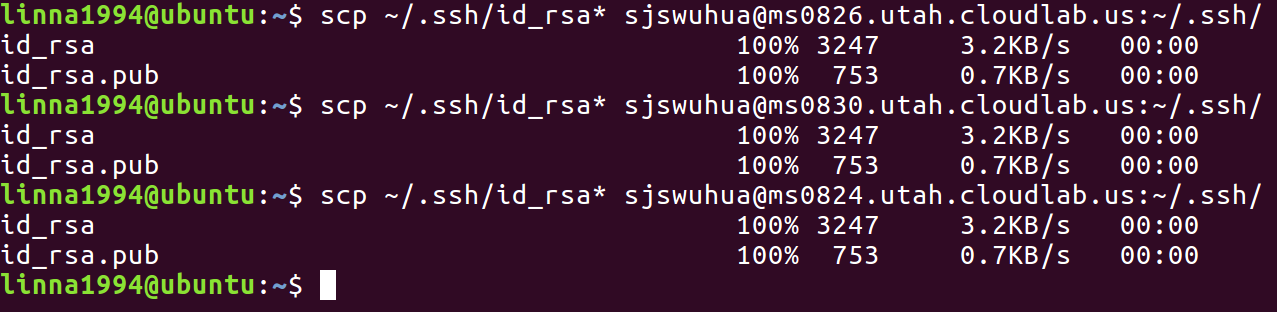
**4.2 Setup passwordless SSH**

We have already created a passwordless SSH in local host, now we need to upload the key to three nodes.

*scp ~/.ssh/id\_rsa\* sjswuhua@ms0826.utah.cloudlab.us:~/.ssh/*

*scp ~/.ssh/id\_rsa\* sjswuhua@ms0830.utah.cloudlab.us:~/.ssh/*

*scp ~/.ssh/id\_rsa\* sjswuhua@ms0824.utah.cloudlab.us:~/.ssh/*

**

Than we set the modify the permission and add related paths

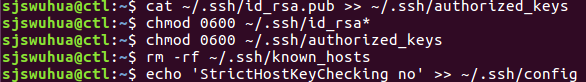
*cat ~/.ssh/id\_rsa.pub >> ~/.ssh/authorized\_keys*

*chmod 0600 ~/.ssh/id\_rsa\**

*chmod 0600 ~/.ssh/authorized\_keys*

*rm -rf ~/.ssh/known\_hosts*

*echo 'StrictHostKeyChecking no' >> ~/.ssh/config*

**

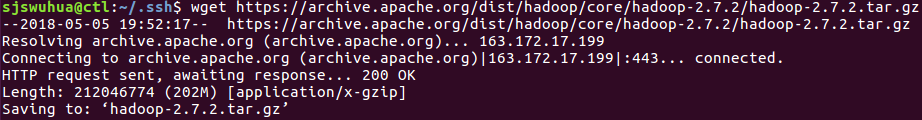
Further, we check the “config” file to ensure the setup.



**4.3 Download and Extract Apache Hadoop**

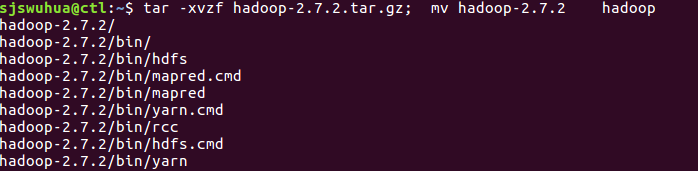
We download Apache Hadoop 2.7.2 from website for each node.

*wget https://archive.apache.org/dist/hadoop/core/hadoop-2.7.2/hadoop-2.7.2.tar.gz*



Then extract the .gz file and rename the directory in each node.

*tar -xvzf hadoop-2.7.2.tar.gz; mv hadoop-2.7.2 hadoop*



**4.4 Setup HDFS**

In each node, setup Hadoop paths.

*export JAVA\_HOME=/usr/lib/jvm/java-8-openjdk-amd64/ >> .bashrc*

*echo export HADOOP\_PREFIX=~/hadoop >> .bashrc;*

*echo export HADOOP\_YARN\_HOME=~/hadoop >> .bashrc;*

*echo export HADOOP\_HOME=~/hadoop >> .bashrc;*

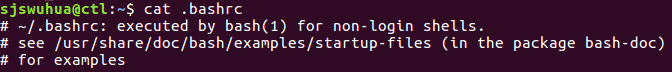
*echo export HADOOP\_CONF\_DIR=~/hadoop/etc/hadoop >> .bashrc;*

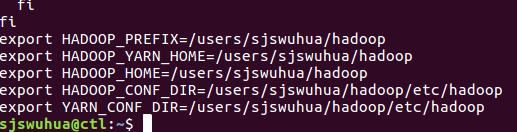
*echo export YARN\_CONF\_DIR=~/hadoop/etc/hadoop >> .bashrc;*

*source .bashrc*

*sed -i '1iexport JAVA\_HOME=/usr/lib/jvm/java-8-openjdk-amd64' hadoop/etc/hadoop/hadoop-env.sh*

Still, we check each file to make our setup is right.





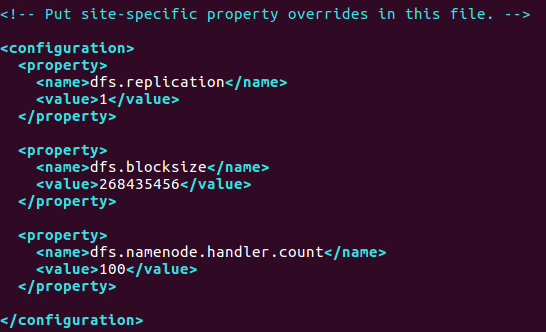
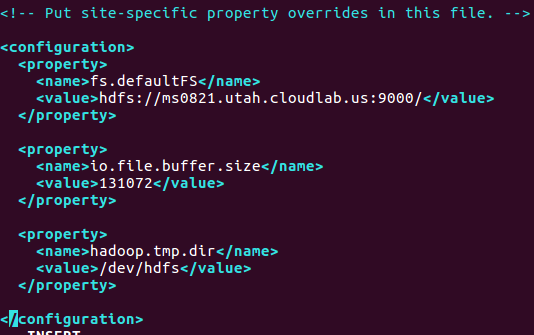


Then, we create HDFS fold and grant permission to read and write.

*sudo mkdir /dev/hdfs; sudo chmod 777 /dev/hdfs*



Then, on each node, we edit core-site.sh and hdfs-site.sh files.



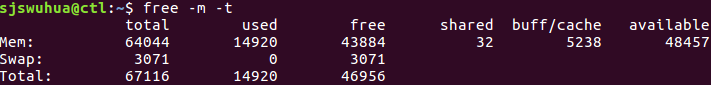
**4.5 Setup Yarn**

In order to setup yarn, first we need to know the number of CPU cores and amount of memory on each node.

*cat /proc/cpuinfo| grep "cpu cores"| uniq*

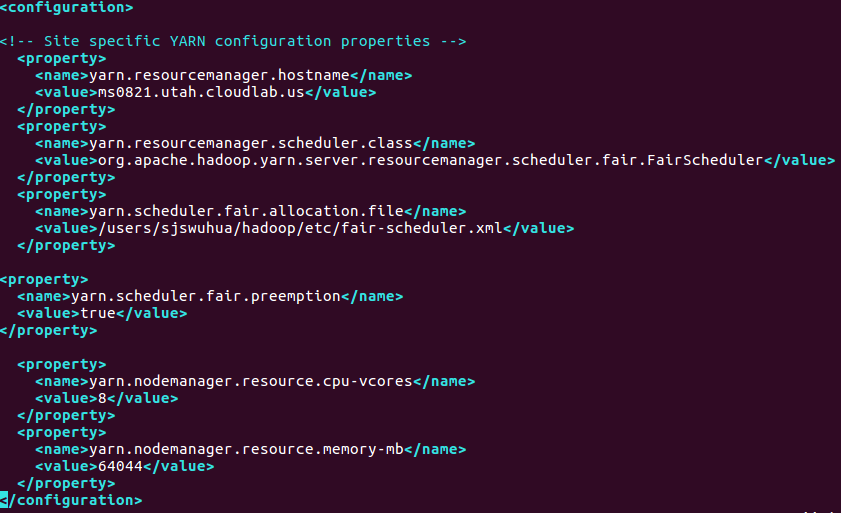


*free -m -t*

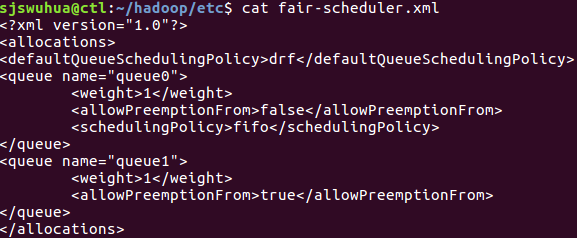


Our nodes all have 8 CPU cores and the amount of memory is 64044 MB.

Then, we edit yarn-site.sh file on each node.



And, we create fair-scheduler.xml file and edit it on each node.

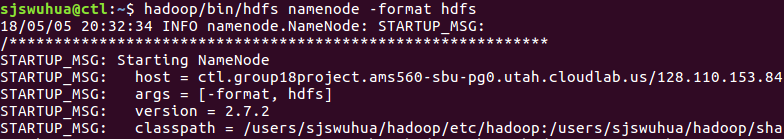


Meanwhile, we create slaves file as follow:



At last, on the master node, we format HDFS.

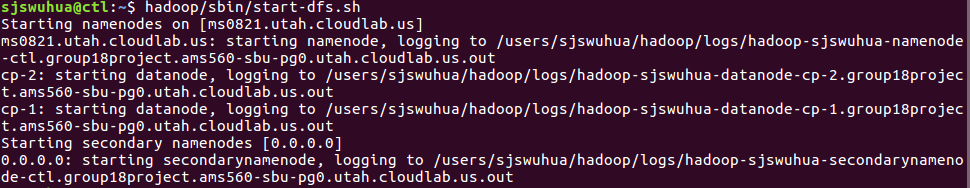
*hadoop/bin/hdfs namenode -format hdfs*



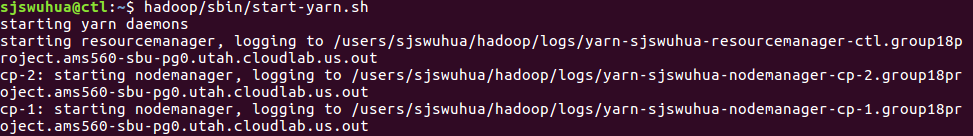
**4.6 Test Yarn**

First, we start HDFS and Yarn operating on master node.

*hadoop/sbin/start-dfs.sh*



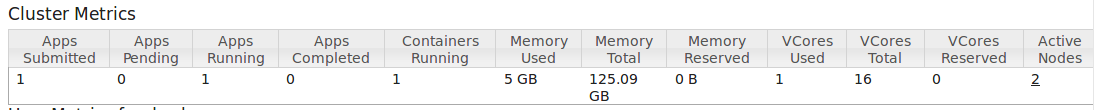
*hadoop/sbin/start-yarn.sh*



From http://ms0826.utah.cloudlab.us:50070/, we can check the HDFS status.



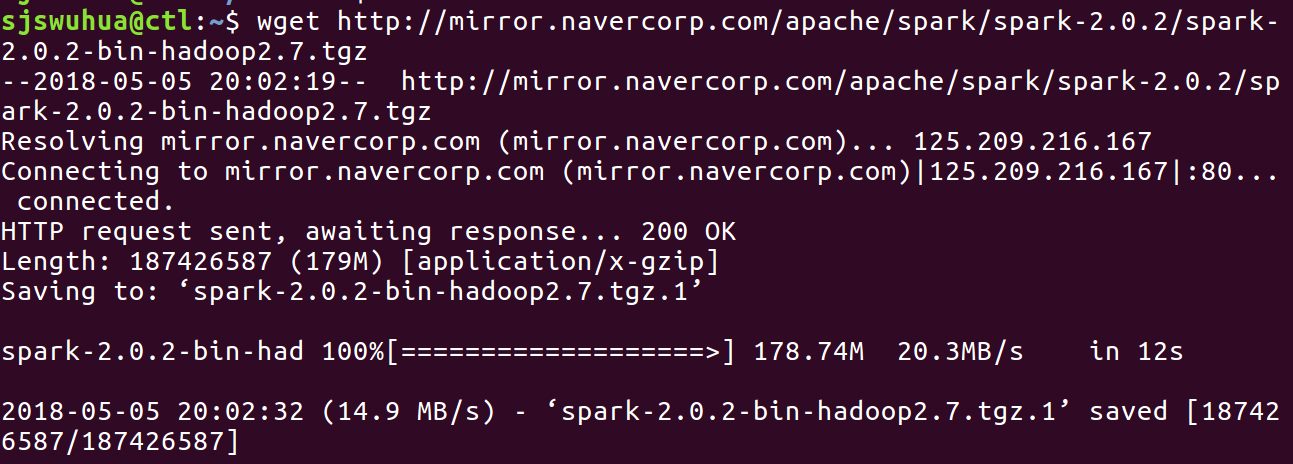
And from http://ms0826.utah.cloudlab.us:8088/, we can check yarn status.



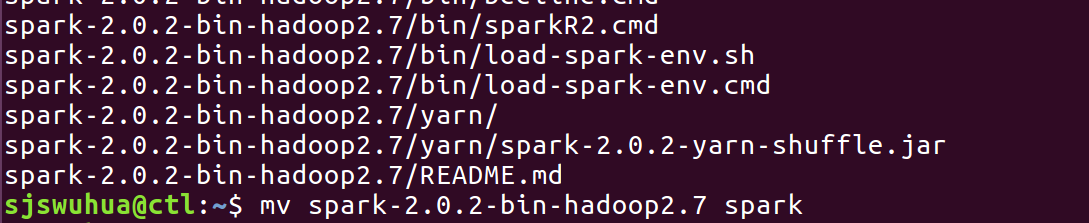
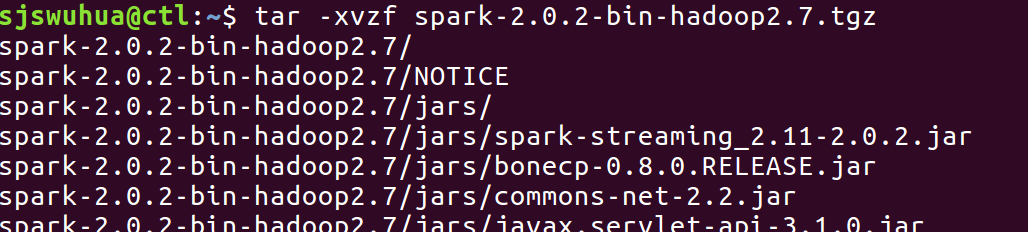
**5. Setup Spark**

**5.1 setup Spark on Yarn**

* On the each nodes, we first download Spark.



Then we decompression.

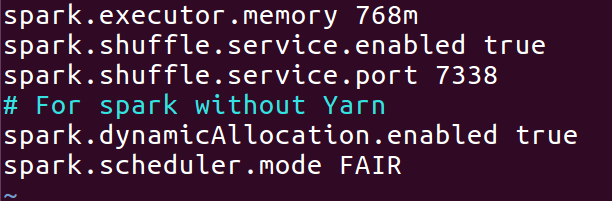


Then we create and edit “slaves” that contains slave nodes.

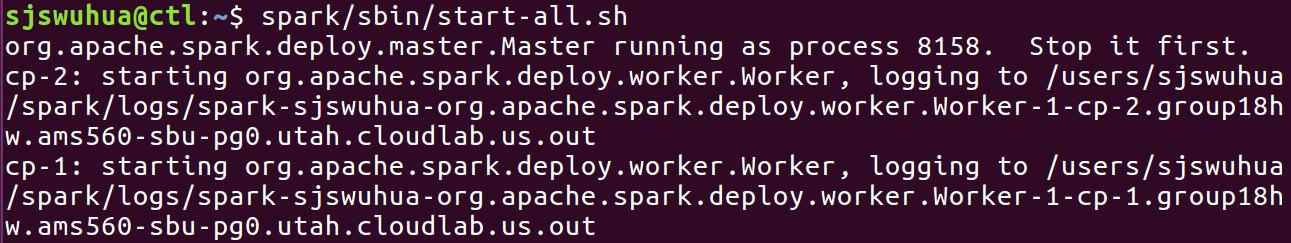




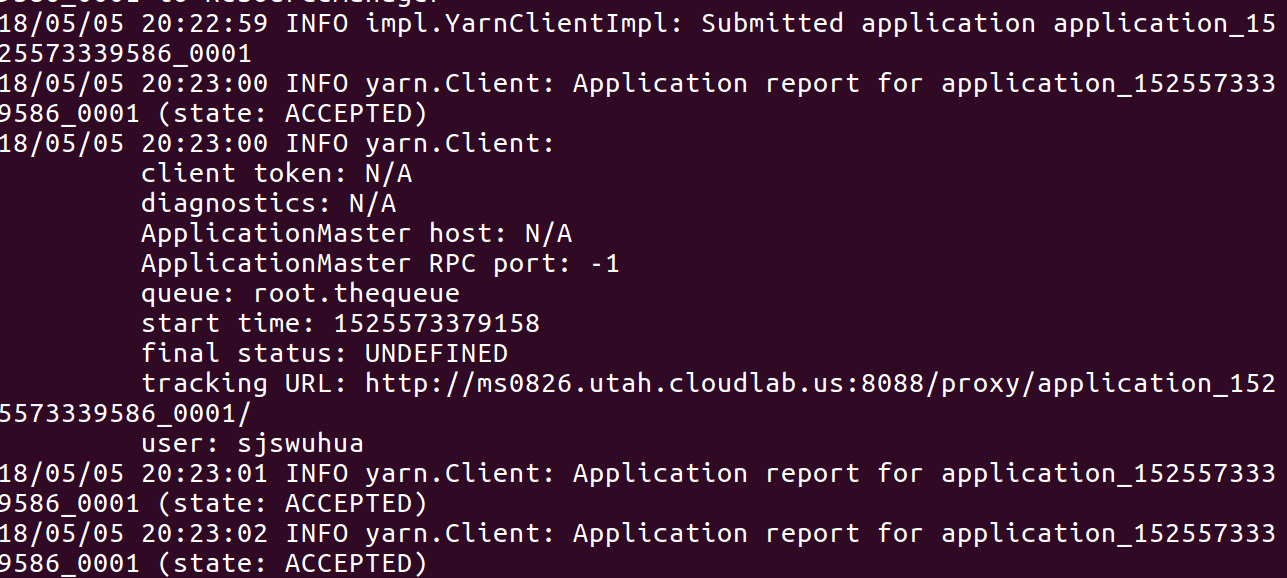
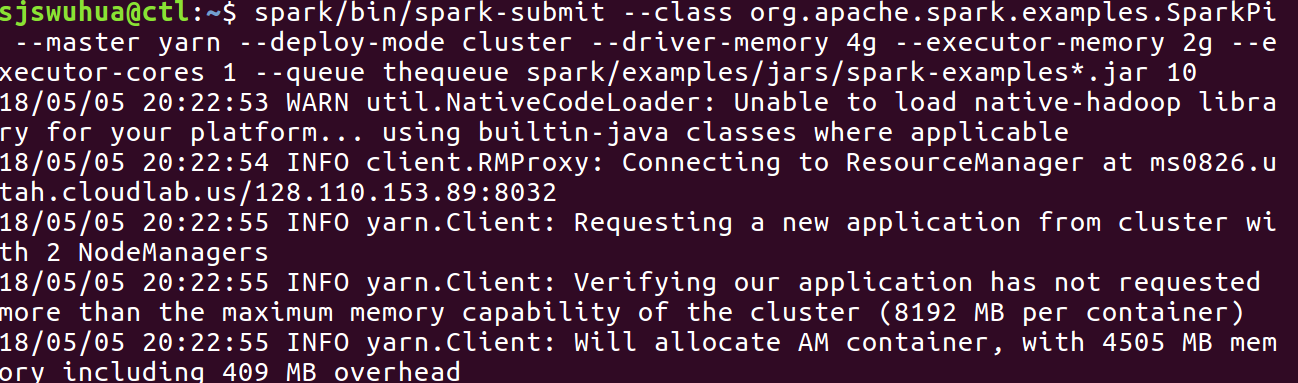
On each nodes, we create “spark-defaults.conf” at ~/spark/conf



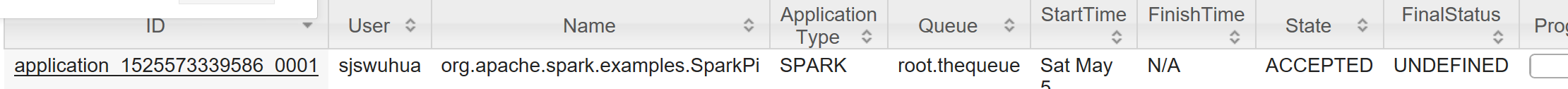
Then we start spark:



We use a script to run.



We check the process on http://ms0826.utah.cloudlab.us:8088/cluster



we can let it only accepted but not running

**5.2 Spark on a standalone cluster**

* We first stop Yarn & Spark first:

*hadoop/sbin/stop-all.sh*

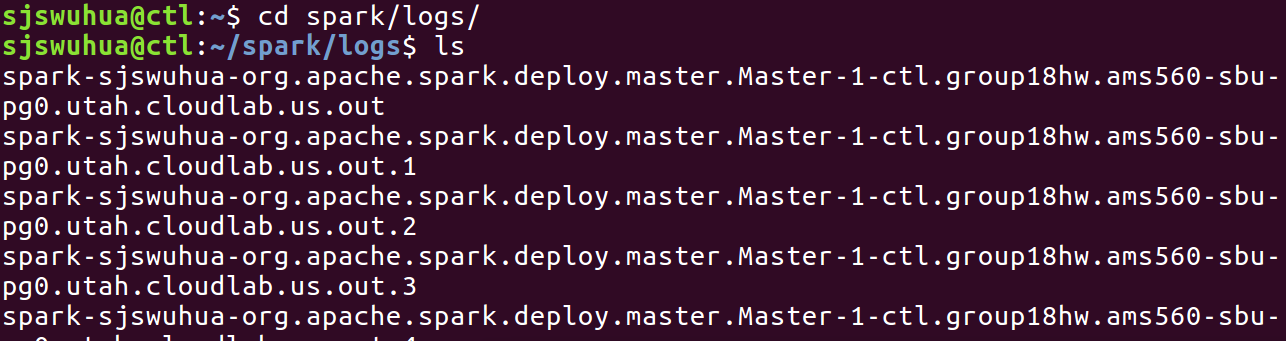
*spark/sbin/stop-all.sh*

* On the master node:

We first use code” *spark/sbin/start-master.sh”* to start master on master node.

Then, copy the spark://HOST:PORT URL from the “\*out” file.

First, we need to find the “\*out” file.



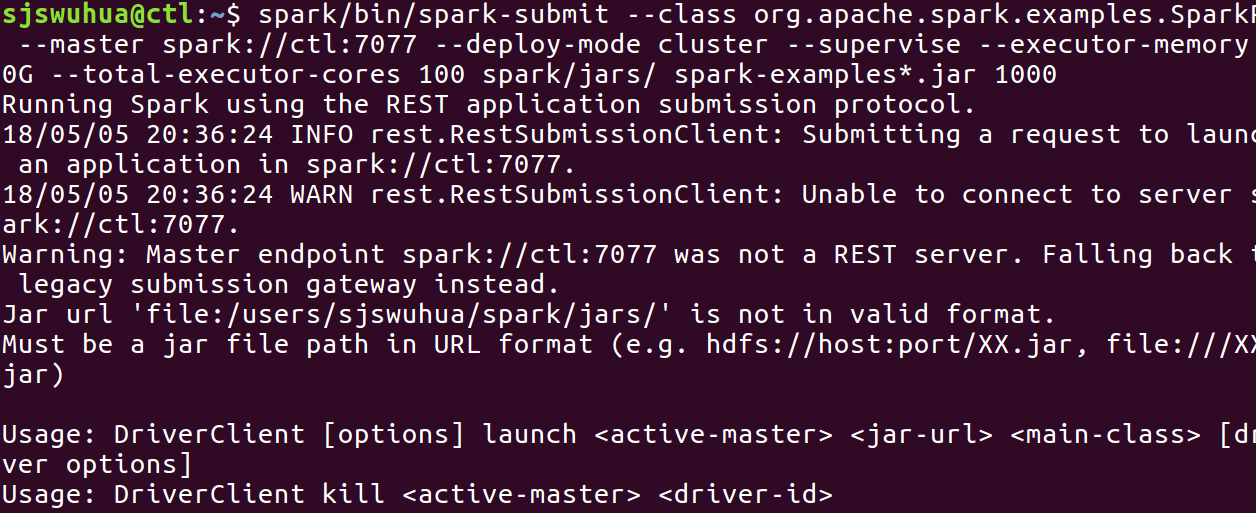
Then in the “\*out” file we find that “**spark://ctl.group18hw.ams560-sbu-pg0.utah.cloudlab.us:7077**” is our spark://HOST:PORT URL.

* On the compute nodes,

We use code ：

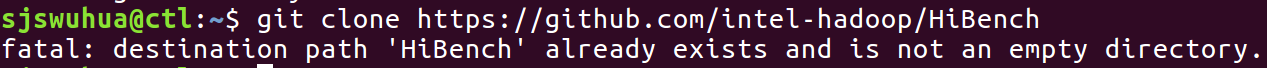
*spark/sbin/start-slave.sh spark://ctl.group18hw.ams560-sbu-pg0.utah.cloudlab.us:7077*

Then we run the spark job

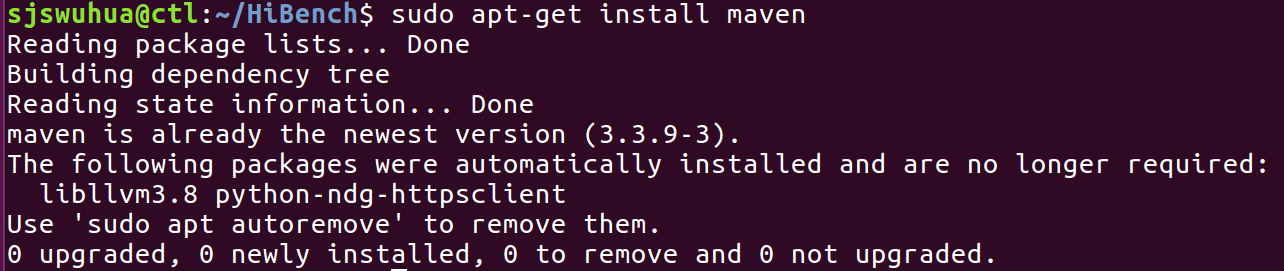


**6 . Setup HiBench**

To the master node, we first download the HiBench.

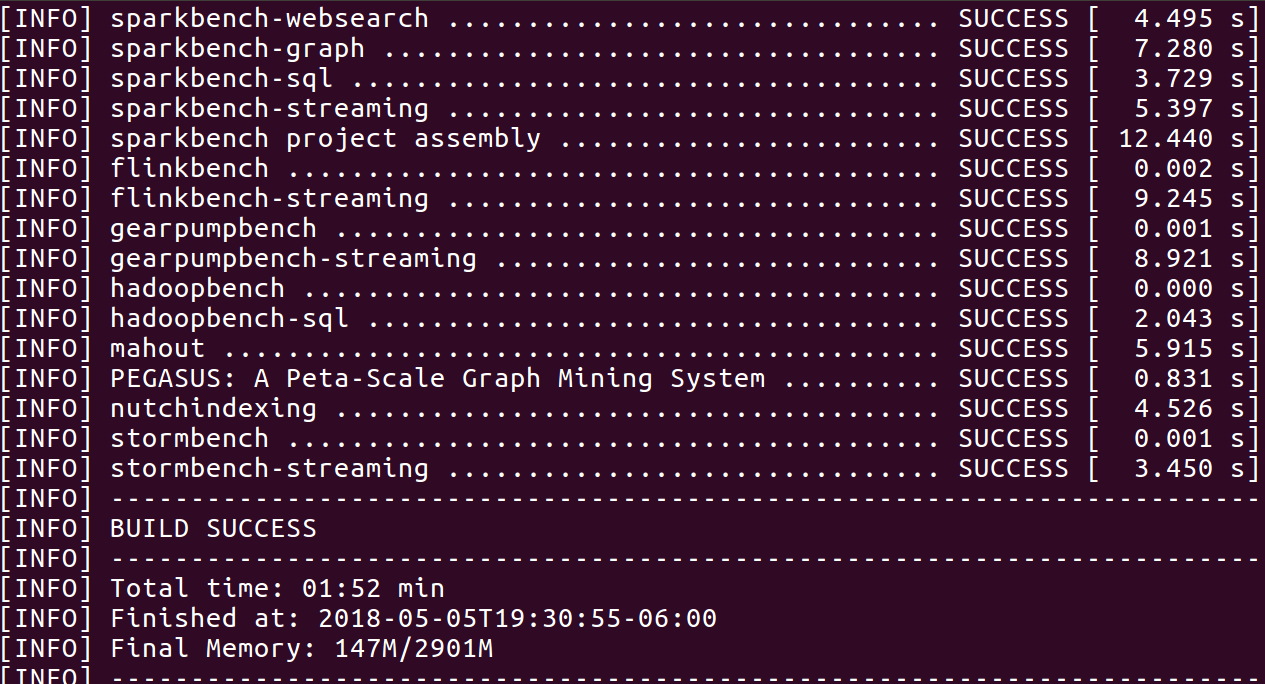


Install maven to build the source code.

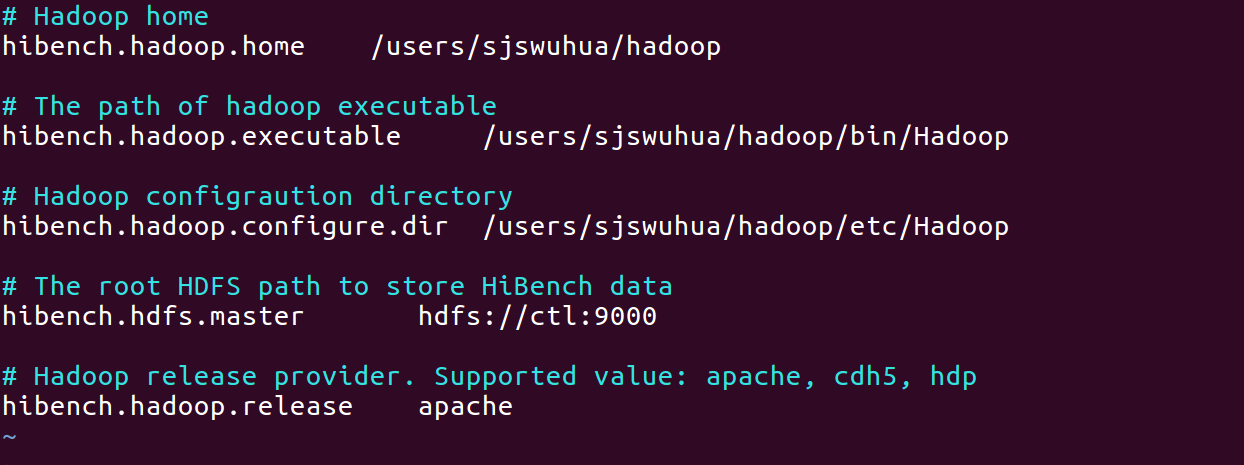
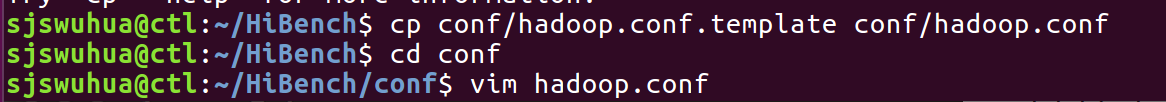


Then we continue to build HiBench, use:

*mvn -Dspark=2.0 -Dscala=2.11 clean package*



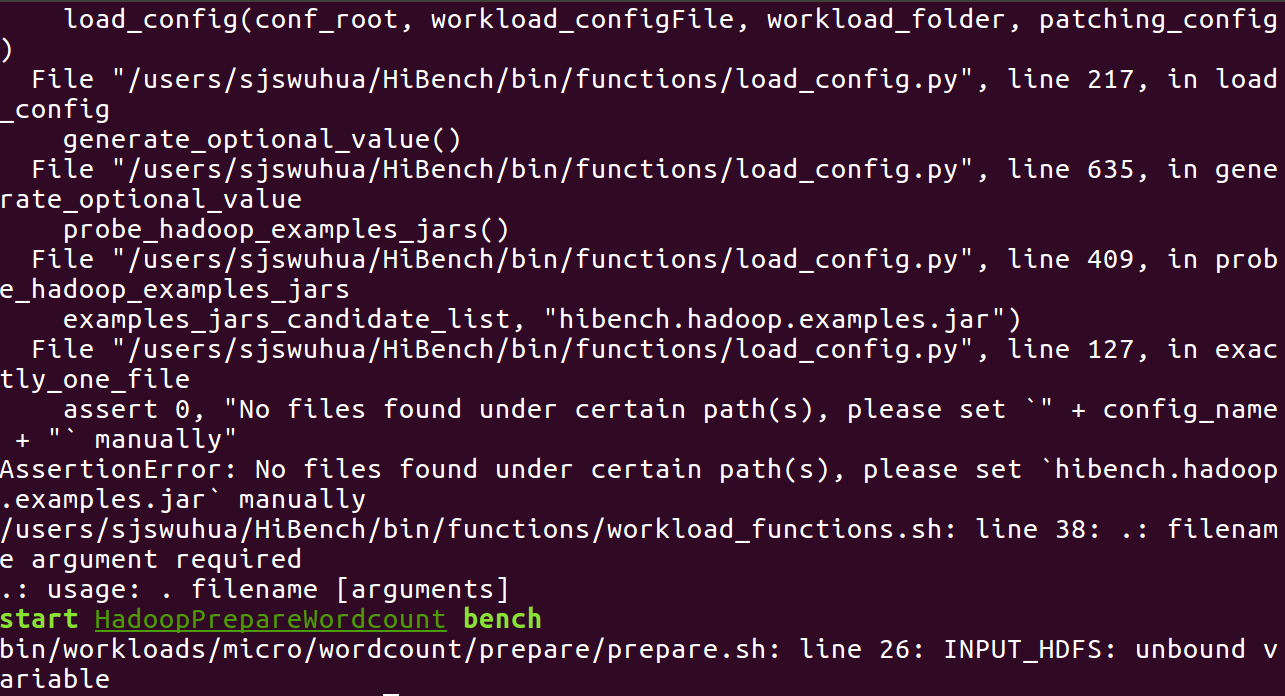
Configure Hibench to run Hadoop bench.



Prepare the workload “wordcount” with code:

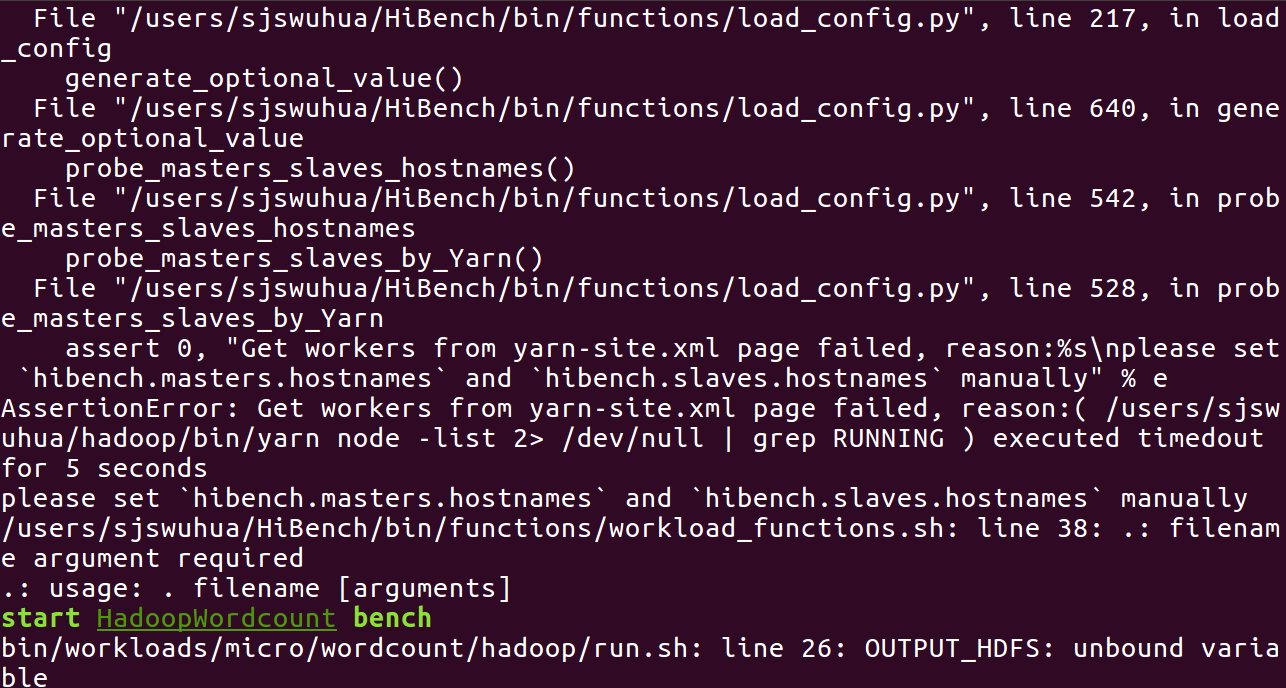
*bin/workloads/micro/wordcount/prepare/prepare.sh*

Then we get:

**

Run the workload “wordcount”:

*bin/workloads/micro/wordcount/hadoop/run.sh*

**

**7. Conclusion**

In this hand-on job, we basically follow the slides to set up HDFS, Yarn, Mapreduce, Spark and Hibench. We also set file permissions, change the configuration, set up a cluster.

However, there exist some errors during the setup process.

But the status of task on yarn is just accepted, not in running. We try to figure out and solve this problem, but no matter how we set the configuration and property in the file “yarn-site.xml” and “mapred-site.xml”, it doesn’t work. Because of the limit of time, we have to give up the new try.

So in this case, HiBench also have some problems.

But we think in this hands-on assignment, we learn a lot by setting up this stuff time after time. In this chance, we also learn deeply about the deployment and setting up the Yarn, Spark, HDFS and etc.