**SparkFaultBench**

**User Document**

**2016.9**

Contents

[1. Prepare system and environment 2](#_Toc27794)

[1.1 Prepare the cluster 2](#_Toc10294)

[1.2 Configure network 3](#_Toc24504)

[1.3 Create a user “hadoop” 4](#_Toc6100)

[1.4 Configure ssh 4](#_Toc14791)

[1.5 Install jdk 5](#_Toc656)

[2. Install and configure the Yarn cluster 5](#_Toc13310)

[2.1 Configurate hadoop 5](#_Toc30023)

[2.2 Install Scala and Spark 8](#_Toc1506)

[3. Start Cluster 10](#_Toc5919)

[3.1 Start hadoop and yarn 10](#_Toc6377)

[3.2 Bulid a path /sparklogs on hdfs to save logs: 11](#_Toc14073)

[3.3 Upload the test file to hdfs and Jar to master. 11](#_Toc26285)

[4. Generate test data 11](#_Toc7515)

[4.1 Choose Spark Application 11](#_Toc15778)

[4.2 Select algorithm 12](#_Toc3780)

[4.3 Configure parameters for application 12](#_Toc18602)

[4.4 Generate test file 13](#_Toc2440)

[4.5 Upload data to HDFS 13](#_Toc22510)

[5. Test execution 13](#_Toc25510)

[5.1 Set parameter information for running 13](#_Toc7662)

[5.2 Fill execution parameters 14](#_Toc20536)

[5.3 Run Application 14](#_Toc27046)

[5.4 Form and view test report 14](#_Toc20840)

[5.5 View details: 15](#_Toc28466)

# 1. Prepare system and environment

## 1.1 Prepare the cluster

Prepare 6 machines with system of CentOS-6.7-x86\_64-minimal. One of them has 6 cores, 6GB RAM and 60G disk, edit the host name of this machine to “Master”:

$ vi /etc/sysconfig/network

See the following drawing:



The others have 6 cores, 6GB RAM and 100G disk, edit the host name of these machine to “Slave1”, “Slave2”, “Slave3”, “Slave4”, “Slave5”.



Restart the machines so that the host name can be changed.

## 1.2 Configure network

Edit the IP of machines in the cluster to a same segment:

$ vi /etc/sysconfig/network-scripts/ifcfg-eth0

The following drawing shows network configuration of one of the machines in this manual , the IP of the others are

“133.133.134.121”,

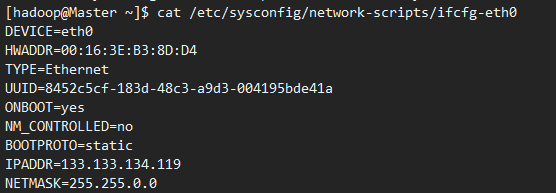
“133.133.134.149”,

“133.133.134.92”,

“133.133.134.94”,

“133.133.134.143”.

And set “BOOTPROTO” to “static” in case of IP changing.



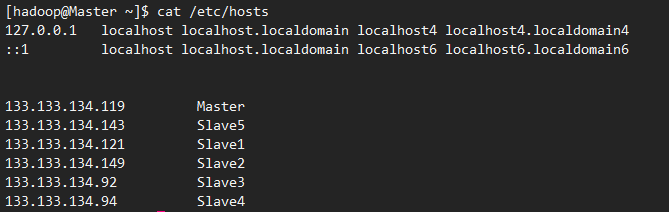
Then restart network on each machine:

$ service network restart

Modify /etc/hosts on each machine to mapping relation of host name and IP in the cluster:

$ vi /etc/hosts

As the following drawing:



Try to ping each other on each machine to see if it works, for examples, on Master:

$ ping Slave1 -c 3

$ ping Slave2 -c 3

$ ping Slave3 -c 3

$ ping Slave4 -c 3

$ ping Slave5 -c 3

## 1.3 Create a user “hadoop”

On each machine in the cluster, create a new user named “hadoop”:

$ useradd -m hadoop -s /bin/bash

$ passwd hadoop

Add administrator privileges for user “hadoop”:

$ visudo

Add a line like this:



Then change account to “hadoop”:

$ su hadoop

## 1.4 Configure ssh

SSH has been installed on CentOS as default. Run the following command on Master:

$ ssh localhost

$ cd ~/.ssh

$ rm ./id\_rsa\*

$ ssh-keygen -t rsa

$ cat ./id\_rsa.pub >> ./authorized\_keys

$ sudo chmod 644 ./authorized\_keys

Then deliver the public key on Slave1 to Slave5:

$ scp ~/.ssh/id\_rsa.pub hadoop@Slave1:/home/hadoop/

On Slave1 to Slave5, add public key to authorized\_keys:

$ ssh localhost

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

$ rm ~/id\_rsa.pub

$ sudo chmod 644 ~/.ssh/authorized\_keys

Restart sshd on Master and check if Master can ssh to each Slave without password successfully:

$ sudo service sshd restart

$ ssh slave1

## 1.5 Install jdk

Install open-JDK 1.7 on each machine with yum:

$ sudo yum install java-1.7.0-openjdk java-1.7.0-openjdk-devel

# 2. Install and configure the Yarn cluster

## 2.1 Configure Hadoop

On Master, download hadoop-2.7.1.tar.gz from

http://mirror.bit.edu.cn/apache/hadoop/common/hadoop-2.7.1/. install Hadoop in /usr/local:

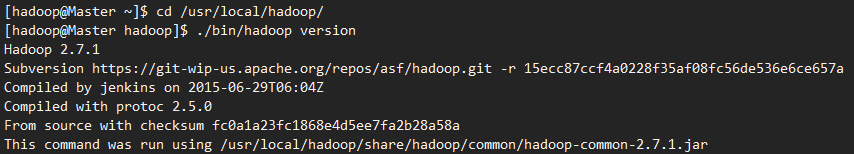
$ sudo tar -zxf hadoop-2.7.1.tar.gz -C /usr/local

$ cd /usr/local/

$ sudo mv ./hadoop-2.7.1/ ./hadoop

$ sudo chown -R hadoop:hadoop ./hadoop

Run the following command to check if Hadoop is correctly installed:

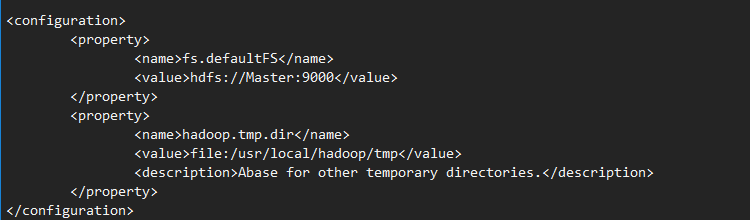


Edit following 5 files in /usr/local/hadoop/etc/hadoop on the Master:

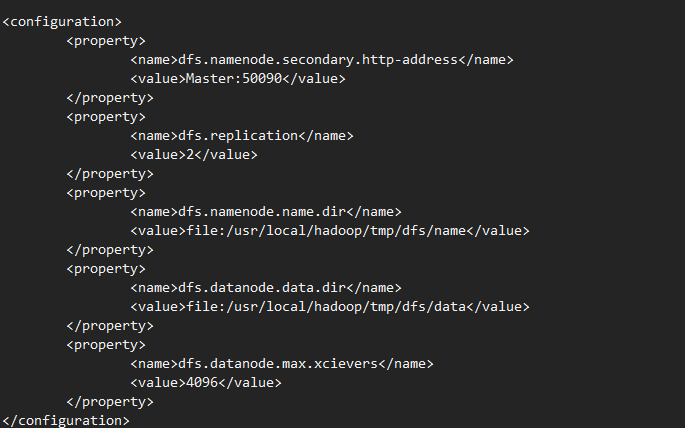
Slaves:



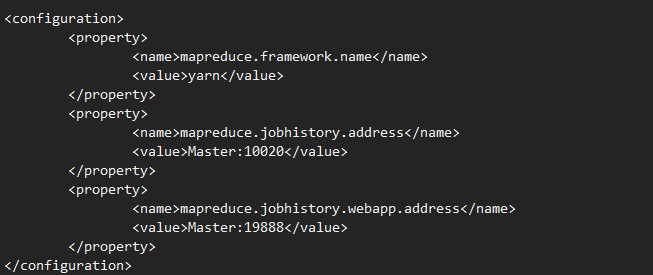
Core-site.xml:



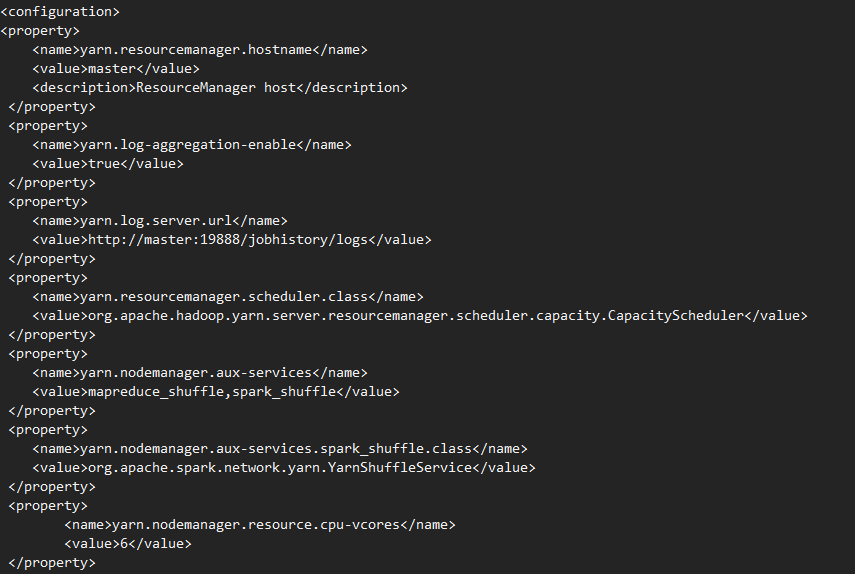
 hdfs-site.xml:



mapred-site.xml:



 yarn-site.xml:



Then copy the / usr / local / Hadoop file on to each node :

$ cd /usr/local

$ tar -zcf ~/hadoop.master.tar.gz ./hadoop

$ scp ~/hadoop.master.tar.gz Slave1:/home/hadoop

Execute the following command on each slave node:

$ sudo tar -zxf ~/hadoop.master.tar.gz -C /usr/local

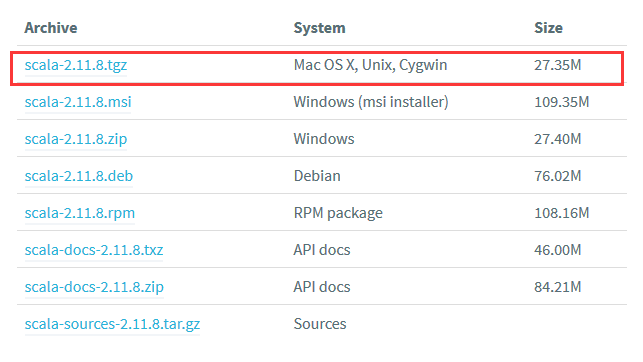
$ sudo chown -R hadoop /usr/local/hadoop

Performs formatting of NameNode on master:

$ hdfs namenode -format

## 2.2 Install Scala and Spark

Download scala 2.11.8 from <http://www.scala-lang.org/download/2.11.8.html>:



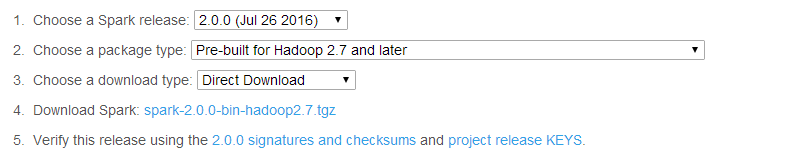
$ sudo tar -zxf scala-2.11.8.tgz -C /usr/local/

$ cd /usr/local

$ sudo mv ./scala-2.11.8.tgz/ ./scala

$ sudo chown -R hadoop:hadoop ./scala

Download spark-2.0.0-bin-hadoop2.7.tgz:



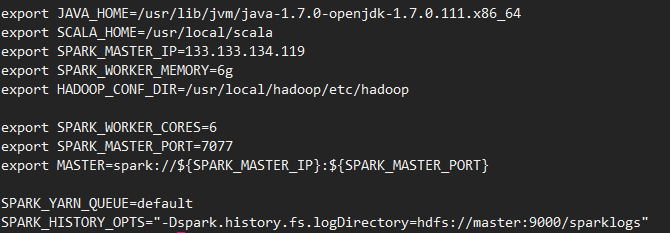
$ sudo tar -zxf spark-2.0.0-bin-hadoop2.7.tgz -C /usr/local/

$ cd /usr/local

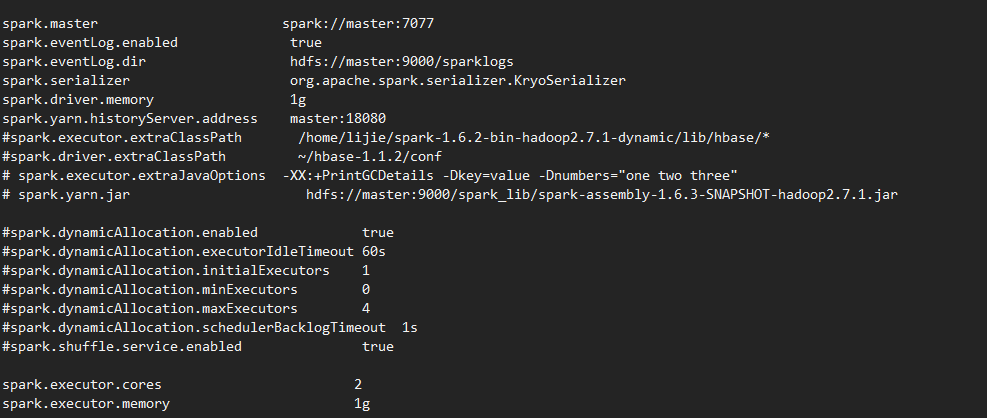
$ sudo mv ./spark-1.6.0-bin-without-hadoop/ ./spark

$ sudo chown -R hadoop:hadoop ./spark

Edit /usr/local/spark/conf/spark-env.sh:



Edit /usr/local/spark/conf/spark-defaults.conf:



Edit /usr/local/spark/conf/slaves：

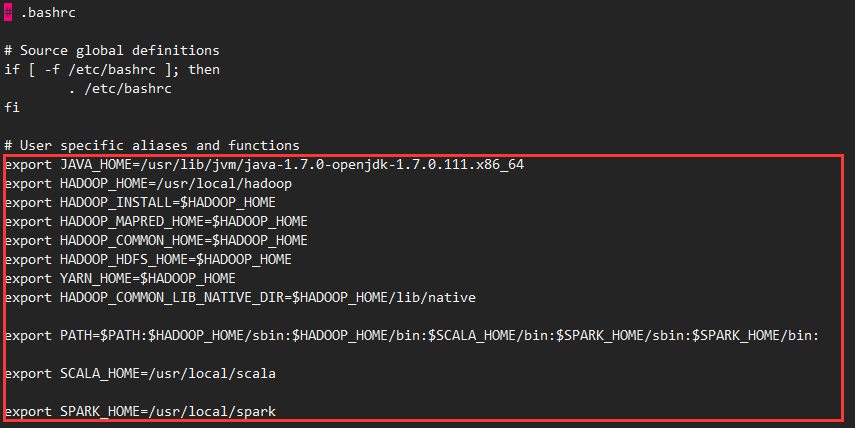


**2.3 Configure environment variable**

Run this command :

$ vi ~/.bashrc

Add lines like this:



Save and run the following command to validate the configuration:

$ source ~/.bashrc

# 3. Start Cluster

## 3.1 Start hadoop and yarn

Start hadoop and yarn:

$ start-all.sh

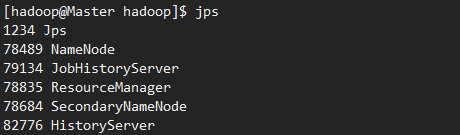
Start hadoop historyserver and spark history server:

$ mr-jobhistory-daemon.sh start historyserver

$ $SPARK\_HOME/sbin/start-hisory-server.sh

start-historyserver.sh

Run “jps” to check out if everything is opened correctly:



## 3.2 Bulid a path /sparklogs on hdfs to save logs:

$ hdfs dfs -mkdir /sparklogs

## 3.3 Upload the test file to hdfs and Jar to master.

Bulid a path to place test file on HDFS:

$ hdfs dfs -mkdir -p data

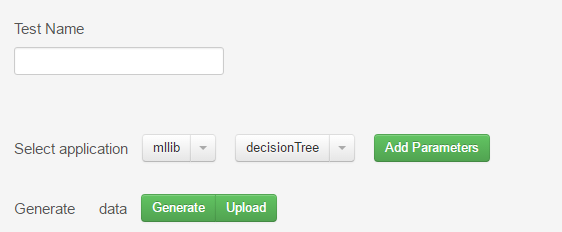
Upload test file “test.txt” to path “data”

$ hdfs dfs -put test.txt data/

Then upload Jar to Master using “scp” command.

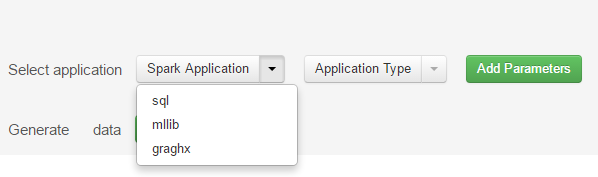
# 4. Generate test data

Open the benchmark, fill the test name, select the Spark application and different types of applications , you can configure the required parameters, and then generate the data to local , and finally uploaded to the HDFS . Details as follows:



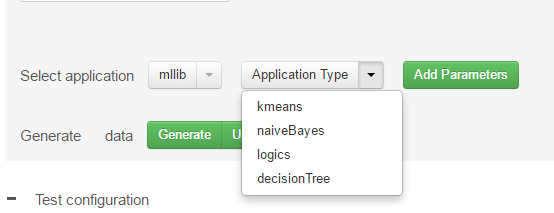
## 4.1 Choose Spark Application

For example, choose application mllib:



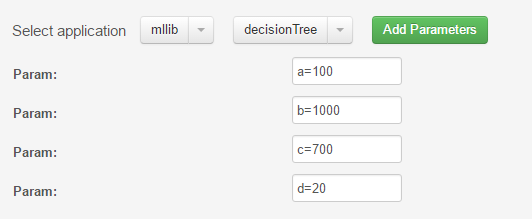
## 4.2 Select algorithm

Select the algorithm of mllib，for example, decisionTree:



## 4.3 Configure parameters for application

Click “Add Parameters”, configure parameters for the application:



## 4.4 Generate test file

Click “Generate”，and generate .txt form test file in local:



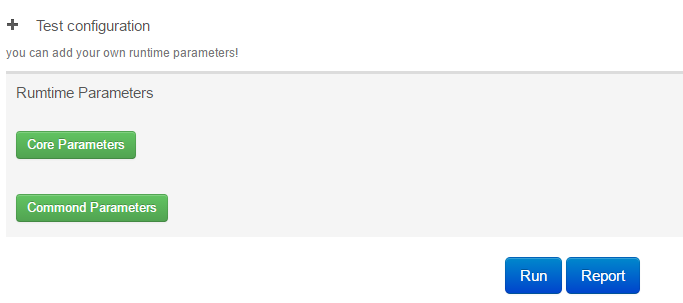
## 4.5 Upload data to HDFS

Click “Upload” button, upload data to HDFS:



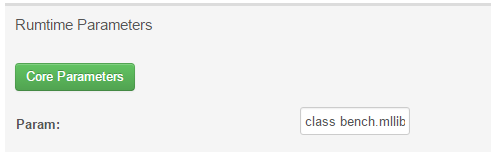
# 5. Test execution

On the benchmark, configurate parameters information during test execution , and run tests . After running , you can check the test report . Test report describes the test application type , data scale, cluster scale, application descriptions , test goal, the way of data generated. In addition , you can also link to Hadoop testing page , view the specific test conditions.



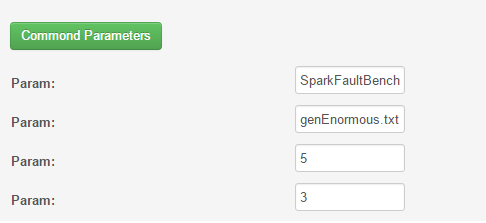
## 5.1 Set parameter information for running

Fill in the parameter information needed to run, Including driver-memory，executor-memory，executor-core, class information and so on:



## 5.2 Fill execution parameters

Fill execution parameters specific class need when running, including the required jar package , etc:



## 5.3 Run Application

Click the “Run” button, you can run the application with information configured before:



## 5.4 Form and view test report

Click the “Report” button, a test report would be formed and showed at the end of running:

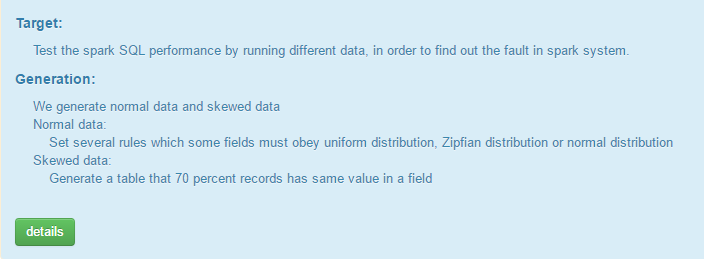
****

Test report:

****

## 5.5 View details:

Click the “details” button below , you can connect to Hadoop monitor screen and view details:



Details:

