Hadoop 学习笔记

harli

2015/1/8

# 官网学习

# [Hadoop2.3+Hive0.12集群部署](http://www.cnblogs.com/Scott007/p/3614960.html)

<http://www.cnblogs.com/Scott007/p/3614960.html>

## 0 机器说明

|  | |
| --- | --- |
| **IP** | **Role** |
| **192.168.1.106** | **NameNode、DataNode、NodeManager、ResourceManager** |
| **192.168.1.107** | **SecondaryNameNode、NodeManager、DataNode** |
| **192.168.1.108** | **NodeManager、DataNode** |
| **192.168.1.106** | **HiveServer** |

## 1 打通无密钥

    配置HDFS，首先就得把机器之间的无密钥配置上。我们这里为了方便，把机器之间的双向无密钥都配置上。

**(1)**产生RSA密钥信息

ssh-keygen -t rsa

一路回车，直到产生一个图形结构，此时便产生了RSA的私钥id\_rsa和公钥id\_rsa.pub，位于/home/user/.ssh目录中。

**(2)**将所有机器节点的ssh证书公钥拷贝至/home/user/.ssh/authorized\_keys文件中，三个机器都一样。

**(3)**切换到root用户，修改/etc/ssh/sshd\_config文件，配置：

RSAAuthentication yes

PubkeyAuthentication yes

AuthorizedKeysFile .ssh/authorized\_keys

**(4)**重启ssh服务：service sshd restart

**(5)**使用ssh服务，远程登录：

http://images.cnitblog.com/i/392365/201403/202240390524436.jpg

 ssh配置成功。

## 2 安装Hadoop2.3

     将对应的hadoop2.3的tar包解压缩到本地之后，主要就是修改配置文件，文件的路径都在etc/hadoop中，下面列出几个主要的。

(1)core-site.xml

[复制代码](javascript:void(0);)

1 <configuration>

2 <property>

3 <name>hadoop.tmp.dir</name>

4 <value>file:/home/sdc/tmp/hadoop-${user.name}</value>

5 </property>

6 <property>

7 <name>fs.default.name</name>

8 <value>hdfs://192.168.1.106:9000</value>

9 </property>

10 </configuration>

[复制代码](javascript:void(0);)

(2)hdfs-site.xml

[复制代码](javascript:void(0);)

1 <configuration>

2 <property>

3 <name>dfs.replication</name>

4 <value>3</value>

5 </property>

6 <property>

7 <name>dfs.namenode.secondary.http-address</name>

8 <value>192.168.1.107:9001</value>

9 </property>

10 <property>

11 <name>dfs.namenode.name.dir</name>

12 <value>file:/home/sdc/dfs/name</value>

13 </property>

14 <property>

15 <name>dfs.datanode.data.dir</name>

16 <value>file:/home/sdc/dfs/data</value>

17 </property>

18 <property>

19 <name>dfs.replication</name>

20 <value>3</value>

21 </property>

22 <property>

23 <name>dfs.webhdfs.enabled</name>

24 <value>true</value>

25 </property>

26 </configuration>

[复制代码](javascript:void(0);)

(3)hadoop-env.sh

主要是将其中的JAVA\_HOME赋值：

export JAVA\_HOME=/usr/local/jdk1.6.0\_27

(4)mapred-site.xml

[复制代码](javascript:void(0);)

1 <configuration>

2 <property>

3 <!-- 使用yarn作为资源分配和任务管理框架 -->

4 <name>mapreduce.framework.name</name>

5 <value>yarn</value>

6 </property>

7 <property>

8 <!-- JobHistory Server地址 -->

9 <name>mapreduce.jobhistory.address</name>

10 <value>centos1:10020</value>

11 </property>

12 <property>

13 <!-- JobHistory WEB地址 -->

14 <name>mapreduce.jobhistory.webapp.address</name>

15 <value>centos1:19888</value>

16 </property>

17 <property>

18 <!-- 排序文件的时候一次同时最多可并行的个数 -->

19 <name>mapreduce.task.io.sort.factor</name>

20 <value>100</value>

21 </property>

22 <property>

23 <!-- reuduce shuffle阶段并行传输数据的数量 -->

24 <name>mapreduce.reduce.shuffle.parallelcopies</name>

25 <value>50</value>

26 </property>

27 <property>

28 <name>mapred.system.dir</name>

29 <value>file:/home/sdc/Data/mr/system</value>

30 </property>

31 <property>

32 <name>mapred.local.dir</name>

33 <value>file:/home/sdc/Data/mr/local</value>

34 </property>

35 <property>

36 <!-- 每个Map Task需要向RM申请的内存量 -->

37 <name>mapreduce.map.memory.mb</name>

38 <value>1536</value>

39 </property>

40 <property>

41 <!-- 每个Map阶段申请的Container的JVM参数 -->

42 <name>mapreduce.map.java.opts</name>

43 <value>-Xmx1024M</value>

44 </property>

45 <property>

46 <!-- 每个Reduce Task需要向RM申请的内存量 -->

47 <name>mapreduce.reduce.memory.mb</name>

48 <value>2048</value>

49 </property>

50 <property>

51 <!-- 每个Reduce阶段申请的Container的JVM参数 -->

52 <name>mapreduce.reduce.java.opts</name>

53 <value>-Xmx1536M</value>

54 </property>

55 <property>

56 <!-- 排序内存使用限制 -->

57 <name>mapreduce.task.io.sort.mb</name>

58 <value>512</value>

59 </property>

60 </configuration>

[复制代码](javascript:void(0);)

　　注意上面的几个内存大小的配置，其中Container的大小一般都要小于所能申请的最大值，否则所运行的Mapreduce任务可能无法运行。

(5)yarn-site.xml

[复制代码](javascript:void(0);)

1 <configuration>

2 <property>

3 <name>yarn.nodemanager.aux-services</name>

4 <value>mapreduce\_shuffle</value>

5 </property>

6 <property>

7 <name>yarn.nodemanager.aux-services.mapreduce.shuffle.class</name>

8 <value>org.apache.hadoop.mapred.ShuffleHandler</value>

9 </property>

10 <property>

11 <name>yarn.resourcemanager.address</name>

12 <value>centos1:8080</value>

13 </property>

14 <property>

15 <name>yarn.resourcemanager.scheduler.address</name>

16 <value>centos1:8081</value>

17 </property>

18 <property>

19 <name>yarn.resourcemanager.resource-tracker.address</name>

20 <value>centos1:8082</value>

21 </property>

22 <property>

23 <!-- 每个nodemanager可分配的内存总量 -->

24 <name>yarn.nodemanager.resource.memory-mb</name>

25 <value>2048</value>

26 </property>

27 <property>

28 <name>yarn.nodemanager.remote-app-log-dir</name>

29 <value>${hadoop.tmp.dir}/nodemanager/remote</value>

30 </property>

31 <property>

32 <name>yarn.nodemanager.log-dirs</name>

33 <value>${hadoop.tmp.dir}/nodemanager/logs</value>

34 </property>

35 <property>

36 <name>yarn.resourcemanager.admin.address</name>

37 <value>centos1:8033</value>

38 </property>

39 <property>

40 <name>yarn.resourcemanager.webapp.address</name>

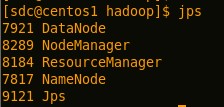
41 <value>centos1:8088</value>

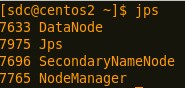
42 </property>

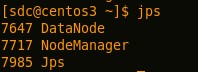
43 </configuration>

[复制代码](javascript:void(0);)

    此外，配置好对应的HADOOP\_HOME环境变量之后，将当前hadoop文件发送到所有的节点，在sbin目录中有**start-all.sh**脚本，启动可见：

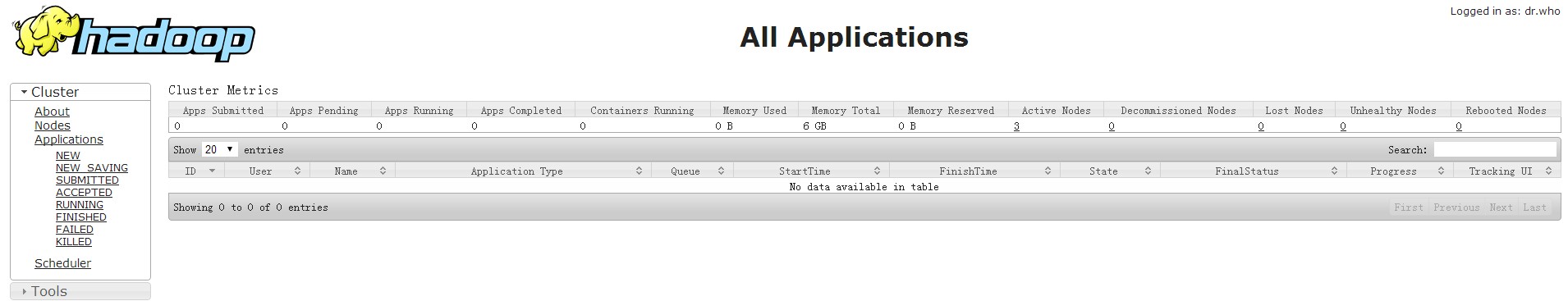




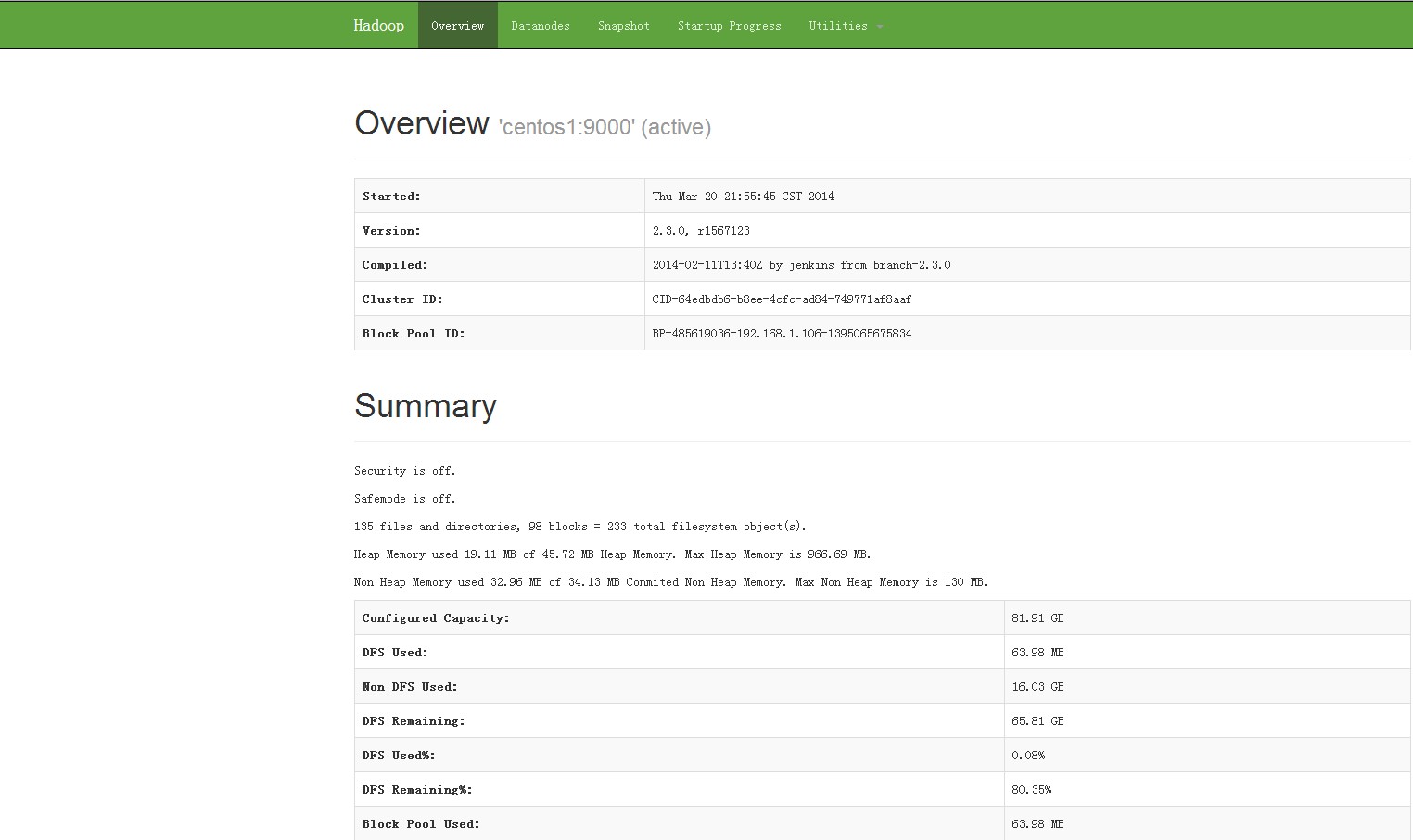


    启动完成之后，有如下两个WEB界面：

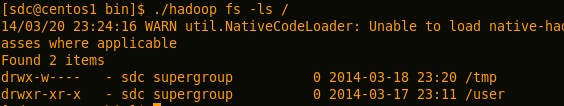
http://192.168.1.106:8088/cluster



http://192.168.1.106:50070/dfshealth.html



使用最简单的命令检查下HDFS：



## 3 安装Hive0.12

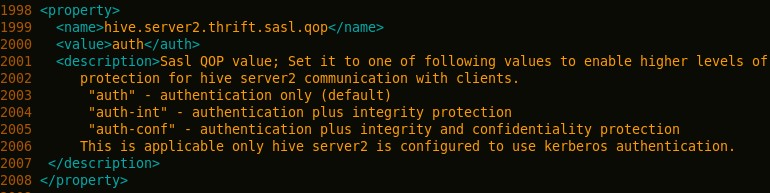
    将Hive的tar包解压缩之后，首先配置下HIVE\_HOME的环境变量。然后便是一些配置文件的修改：

(1)hive-env.sh

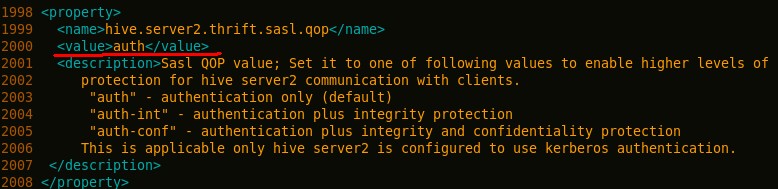
将其中的HADOOP\_HOME变量修改为当前系统变量值。

(2)hive-site.xml

* 修改hive.server2.thrift.sasl.qop属性



修改为：



* 将hive.metastore.schema.verification对应的值改为false

        强制metastore的schema一致性，开启的话会校验在metastore中存储的信息的版本和hive的jar包中的版本一致性，并且关闭自动schema迁移，用户必须手动的升级hive并且迁移schema，关闭的话只会在版本不一致时给出警告。

* 修改hive的元数据存储位置，改为mysql存储：

[复制代码](javascript:void(0);)

1 <property>

2 <name>javax.jdo.option.ConnectionURL</name>

3 <value>jdbc:mysql://localhost:3306/hive?characterEncoding=UTF-8</value>

4 <description>JDBC connect string for a JDBC metastore</description>

5 </property>

6

7 <property>

8 <name>javax.jdo.option.ConnectionDriverName</name>

9 <value>com.mysql.jdbc.Driver</value>

10 <description>Driver class name for a JDBC metastore</description>

11 </property>

12

13 <property>

14 <name>javax.jdo.PersistenceManagerFactoryClass</name>

15 <value>org.datanucleus.api.jdo.JDOPersistenceManagerFactory</value>

16 <description>class implementing the jdo persistence</description>

17 </property>

18

19 <property>

20 <name>javax.jdo.option.DetachAllOnCommit</name>

21 <value>true</value>

22 <description>detaches all objects from session so that they can be used after transaction is committed</description>

23 </property>

24

25 <property>

26 <name>javax.jdo.option.NonTransactionalRead</name>

27 <value>true</value>

28 <description>reads outside of transactions</description>

29 </property>

30

31 <property>

32 <name>javax.jdo.option.ConnectionUserName</name>

33 <value>hive</value>

34 <description>username to use against metastore database</description>

35 </property>

36

37 <property>

38 <name>javax.jdo.option.ConnectionPassword</name>

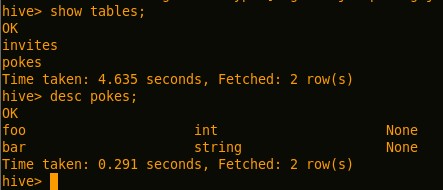
39 <value>123</value>

40 <description>password to use against metastore database</description>

41 </property>

[复制代码](javascript:void(0);)

    在bin下启动hive脚本，运行几个hive语句：



## 4 安装Mysql5.6

 见[http://www.cnblogs.com/Scott007/p/3572604.html](http://www.cnblogs.com/Scott007/p/3572604.html" \t "_blank)

## 5 Pi计算实例、Hive表的计算实例运行

     在Hadoop的安装目录bin子目录下，执行hadoop自带的示例，pi的计算，命令为：

./hadoop jar ../share/hadoop/mapreduce/hadoop-mapreduce-examples-2.3.0.jar pi 10 10

运行日志为：

[复制代码](javascript:void(0);)

1 Number of Maps = 10

2 Samples per Map = 10

3 14/03/20 23:50:04 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable

4 Wrote input for Map #0

5 Wrote input for Map #1

6 Wrote input for Map #2

7 Wrote input for Map #3

8 Wrote input for Map #4

9 Wrote input for Map #5

10 Wrote input for Map #6

11 Wrote input for Map #7

12 Wrote input for Map #8

13 Wrote input for Map #9

14 Starting Job

15 14/03/20 23:50:06 INFO client.RMProxy: Connecting to ResourceManager at centos1/192.168.1.106:8080

16 14/03/20 23:50:07 INFO input.FileInputFormat: Total input paths to process : 10

17 14/03/20 23:50:07 INFO mapreduce.JobSubmitter: number of splits:10

18 14/03/20 23:50:08 INFO mapreduce.JobSubmitter: Submitting tokens for job: job\_1395323769116\_0001

19 14/03/20 23:50:08 INFO impl.YarnClientImpl: Submitted application application\_1395323769116\_0001

20 14/03/20 23:50:08 INFO mapreduce.Job: The url to track the job: http://centos1:8088/proxy/application\_1395323769116\_0001/

21 14/03/20 23:50:08 INFO mapreduce.Job: Running job: job\_1395323769116\_0001

22 14/03/20 23:50:18 INFO mapreduce.Job: Job job\_1395323769116\_0001 running in uber mode : false

23 14/03/20 23:50:18 INFO mapreduce.Job: map 0% reduce 0%

24 14/03/20 23:52:21 INFO mapreduce.Job: map 10% reduce 0%

25 14/03/20 23:52:27 INFO mapreduce.Job: map 20% reduce 0%

26 14/03/20 23:52:32 INFO mapreduce.Job: map 30% reduce 0%

27 14/03/20 23:52:34 INFO mapreduce.Job: map 40% reduce 0%

28 14/03/20 23:52:37 INFO mapreduce.Job: map 50% reduce 0%

29 14/03/20 23:52:41 INFO mapreduce.Job: map 60% reduce 0%

30 14/03/20 23:52:43 INFO mapreduce.Job: map 70% reduce 0%

31 14/03/20 23:52:46 INFO mapreduce.Job: map 80% reduce 0%

32 14/03/20 23:52:48 INFO mapreduce.Job: map 90% reduce 0%

33 14/03/20 23:52:51 INFO mapreduce.Job: map 100% reduce 0%

34 14/03/20 23:52:59 INFO mapreduce.Job: map 100% reduce 100%

35 14/03/20 23:53:02 INFO mapreduce.Job: Job job\_1395323769116\_0001 completed successfully

36 14/03/20 23:53:02 INFO mapreduce.Job: Counters: 49

37 File System Counters

38 FILE: Number of bytes read=226

39 FILE: Number of bytes written=948145

40 FILE: Number of read operations=0

41 FILE: Number of large read operations=0

42 FILE: Number of write operations=0

43 HDFS: Number of bytes read=2670

44 HDFS: Number of bytes written=215

45 HDFS: Number of read operations=43

46 HDFS: Number of large read operations=0

47 HDFS: Number of write operations=3

48 Job Counters

49 Launched map tasks=10

50 Launched reduce tasks=1

51 Data-local map tasks=10

52 Total time spent by all maps in occupied slots (ms)=573584

53 Total time spent by all reduces in occupied slots (ms)=20436

54 Total time spent by all map tasks (ms)=286792

55 Total time spent by all reduce tasks (ms)=10218

56 Total vcore-seconds taken by all map tasks=286792

57 Total vcore-seconds taken by all reduce tasks=10218

58 Total megabyte-seconds taken by all map tasks=440512512

59 Total megabyte-seconds taken by all reduce tasks=20926464

60 Map-Reduce Framework

61 Map input records=10

62 Map output records=20

63 Map output bytes=180

64 Map output materialized bytes=280

65 Input split bytes=1490

66 Combine input records=0

67 Combine output records=0

68 Reduce input groups=2

69 Reduce shuffle bytes=280

70 Reduce input records=20

71 Reduce output records=0

72 Spilled Records=40

73 Shuffled Maps =10

74 Failed Shuffles=0

75 Merged Map outputs=10

76 GC time elapsed (ms)=710

77 CPU time spent (ms)=71800

78 Physical memory (bytes) snapshot=6531928064

79 Virtual memory (bytes) snapshot=19145916416

80 Total committed heap usage (bytes)=5696757760

81 Shuffle Errors

82 BAD\_ID=0

83 CONNECTION=0

84 IO\_ERROR=0

85 WRONG\_LENGTH=0

86 WRONG\_MAP=0

87 WRONG\_REDUCE=0

88 File Input Format Counters

89 Bytes Read=1180

90 File Output Format Counters

91 Bytes Written=97

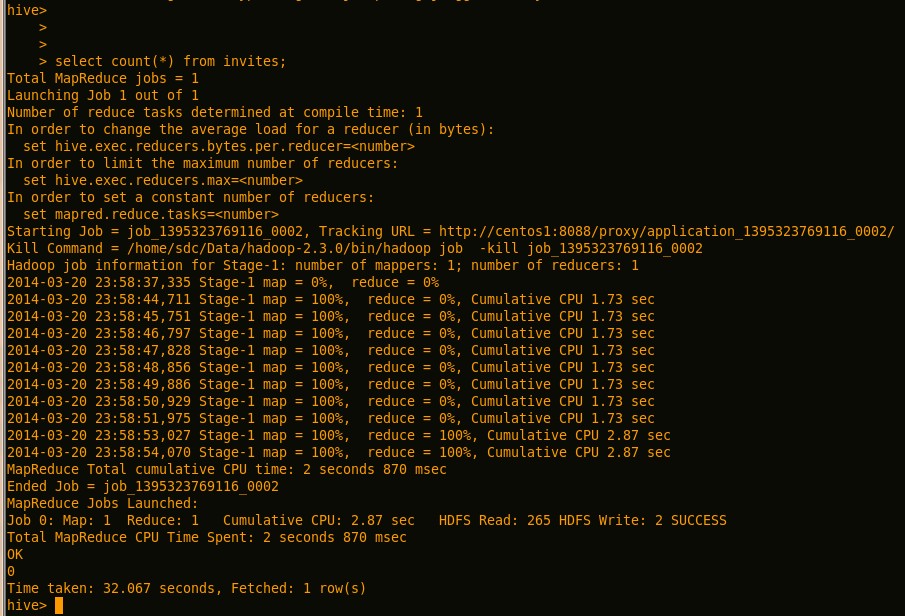
92 Job Finished in 175.556 seconds

93 Estimated value of Pi is 3.20000000000000000000

[复制代码](javascript:void(0);)

    如果运行不起来，那说明HDFS的配置有问题啊！

    Hive中执行count等语句，可以触发mapduce任务：



    如果运行的时候出现类似于如下的错误：

Error in metadata: java.lang.RuntimeException: Unable to instantiate org.apache.hadoop.hive.metastore.HiveMetaStoreClient

说明元数据存储有问题，可能是以下两方面的原因：

(1)HDFS的元数据存储有问题：

$HADOOP\_HOME/bin/hadoop fs -mkdir /tmp

$HADOOP\_HOME/bin/hadoop fs -mkdir /user/hive/warehouse

$HADOOP\_HOME/bin/hadoop fs -chmod g+w /tmp

$HADOOP\_HOME/bin/hadoop fs -chmod g+w /user/hive/warehouse

(2)Mysql的授权有问题：

在mysql中执行如下命令，其实就是给Mysql中的Hive数据库赋权

grant all on db.\* to hive@'%' identified by '密码';(使用户可以远程连接Mysql)

grant all on db.\* to hive@'localhost' identified by '密码';(使用户可以本地连接Mysql)

flush privileges;

    具体哪方面的原因，可以查看hive的日志。

s

# Hadoop 调试

Java选项设置:

\_OPTS -Xdebug -Xrunjdwp:transport=dt\_socket,address=6666,server=y,suspend=y"

<http://www.lifeba.org/arch/hadoop_debug.html>

**一、HDFS调试**

1、bin/hdfs 脚本中  
namenode下加入

 HADOOP\_OPTS="$HADOOP\_OPTS -Xdebug -Xrunjdwp:transport=dt\_socket,address=7777,server=y,suspend=y"

secondarynamenode下加入

HADOOP\_OPTS="$HADOOP\_OPTS -Xdebug -Xrunjdwp:transport=dt\_socket,address=7778,server=y,suspend=y"

**[](http://www.lifeba.org/wp-content/uploads/2012/05/lifeba.org_hadoop_debug_1.jpg)**

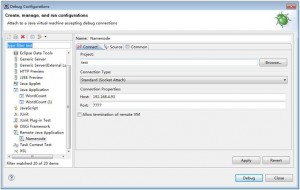
2、执行相应的脚本，然后通过Eclipse IDE连接来下断调试。

启动服务./start-dfs.sh，停止服务./stop-dfs.sh

启动后会停留在 Listening for transport dt\_socket at address: 7777 ，IDE连接后才会继续。

**二、eclipse hadoop 调试**

打开Eclipse的 Debug Configurations ，新建一个Remote Java Application。指定下project工程项目，以及需要调试的hadoop节点的ip和端口。

**[](http://www.lifeba.org/wp-content/uploads/2012/05/lifeba.org_hadoop_debug_3.jpg)**

在hadoop节点启动后，点击debug来连接远程的hadoop程序进行调试。hadoop节点在调试启动方式下，必须IDE连接上去后，才会继续，否则会停留在  
上面的Listening for transport dt\_socket at address: 7777，直到有IDE连接成功。

说明：  
1、如果需要调试datanode，在datanode节点下的bin/hdfs 中的datanode节点同样加入

HADOOP\_OPTS="$HADOOP\_OPTS -Xdebug -Xrunjdwp:transport=dt\_socket,address=7777,server=y,suspend=y"

然后通过./hdfs datanode datanode启动，这样就可以连调。

2、不能使用外层hadoop启动或者start-all.sh启动，调试的话必须直接使用实际调用的sh启动。如  
如 ./hadoop namenode -format 要换为 ./hdfs namenode -format

**三、MapReduce调试**

1、调试MapReduce先启动HDFS服务，然后在bin/mapred 脚本中  
jobtracker下加入

HADOOP\_OPTS="$HADOOP\_OPTS -Xdebug -Xrunjdwp:transport=dt\_socket,address=6666,server=y,suspend=y"

tasktracker下加入

HADOOP\_OPTS="$HADOOP\_OPTS -Xdebug -Xrunjdwp:transport=dt\_socket,address=6667,server=y,suspend=y"

[](http://www.lifeba.org/wp-content/uploads/2012/05/lifeba.org_hadoop_debug_2.jpg)

2、执行相应的脚本，同样通过Eclipse IDE连接来下断调试。

启动服务start-mapred.sh，停止服务stop-mapred.sh

如果需要驱动某个节点，可以执行单个启动命令。./mapred jobtracker 后面指定需要开启的服务的名称。

# 参考

<http://www.lifeba.org/arch/hadoop_debug.html>