# Hadoop+Spark HA集群搭建

1. 先决条件

1.安装环境

|  |  |  |
| --- | --- | --- |
| 软件名 | 版本 | 内核 |
| VirtualBox | Oracle 5.1.22 | R115126 |
| linux | CentOS-6.9-x86\_64-bin-DVD | 6.9 |
| java | jdk-8u131-linux-x64.tar.gz | build 1.8.0\_131-b11 |
| scala | scala-2.12.2.tgz | 2.12.2 |
| hadoop | hadoop-2.6.5.tar.gz | 2.6.5 |
| spark | spark-1.6.3-bin-hadoop2.6.tgz | 1.6.3 |
| zookeeper | zookeeper-3.4.10.tar.gz | 3.4.10 |
|  |  |  |

2. 各主机节点职责分配

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| IP | 主机名 | 安装软件 | 部署模块 | 运行进程 |
| **192.168.2.11** | **bigdata-master-01** | **hadoop**  **Java/Scala** | **NameNode**  **ResourceManager** | **NameNode**  **DFSZKFailoverController**  **ResourceManager** |
| **192.168.2.12** | **bigdata-master-02** | **hadoop**  **Java/Scala** | **NameNode**  **ResourceManager** | **NameNode**  **DFSZKFailoverController**  **ResourceManager** |
| **192.168.2.13** | **bigdata-slave-01** | **Hadoop**  **Java/Scala**  **zookeeper** | **DataNode**  **NodeManager**  **Zookeeper** | **DataNode**  **NodeManager**  **JournalNode**  **QuorumPeerMain** |
| **192.168.2.14** | **bigdata-slave-02** | **Hadoop**  **Java/Scala**  **zookeeper** | **DataNode**  **NodeManager**  **Zookeeper** | **DataNode**  **NodeManager**  **JournalNode**  **QuorumPeerMain** |
| **192.168.2.15** | **bigdata-slave-03** | **Hadoop**  **Java/Scala**  **zookeeper** | **DataNode**  **NodeManager**  **Zookeeper** | **DataNode**  **NodeManager**  **JournalNode**  **QuorumPeerMain** |

3.目录规划

bigdata---backup（安装包文件备份）

---tools（shell脚本）

---workspace（工作目录）

4.软件下载地址

1) VirtualBox

<http://101.44.1.6/files/5093000006A0E32A/download.virtualbox.org/virtualbox/5.1.22/VirtualBox-5.1.22-115126-Win.exe>

2) Centos

<http://iso.mirrors.ustc.edu.cn/centos/6.9/isos/x86_64/CentOS-6.9-x86_64-bin-DVD1.iso>

<http://iso.mirrors.ustc.edu.cn/centos/6.9/isos/x86_64/CentOS-6.9-x86_64-bin-DVD2.iso>

3)jdk

<http://download.oracle.com/otn-pub/java/jdk/8u131-b11/d54c1d3a095b4ff2b6607d096fa80163/jdk-8u131-linux-x64.tar.gz>

4)hadoop

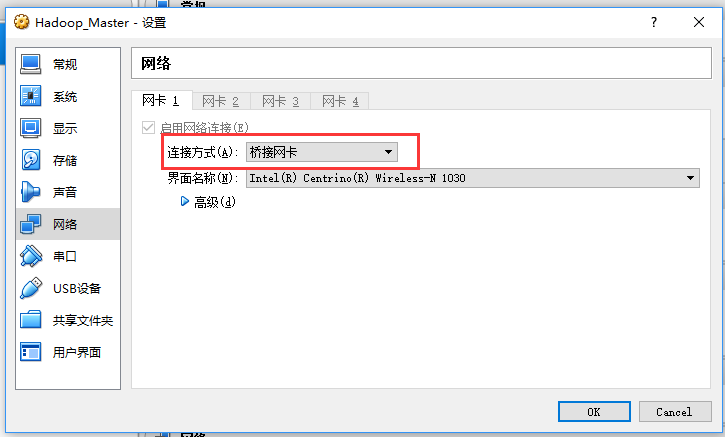
<http://www.apache.org/dyn/closer.cgi/hadoop/common/hadoop-2.6.5/hadoop-2.6.5.tar.gz>

5)spark

https://d3kbcqa49mib13.cloudfront.net/spark-1.6.3-bin-hadoop2.6.tgz

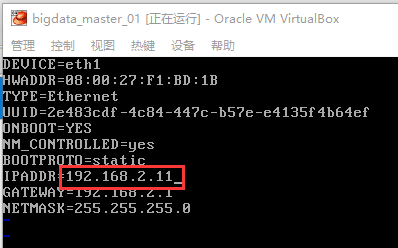
二．配置vm主机网络

1.配置网络连接模式



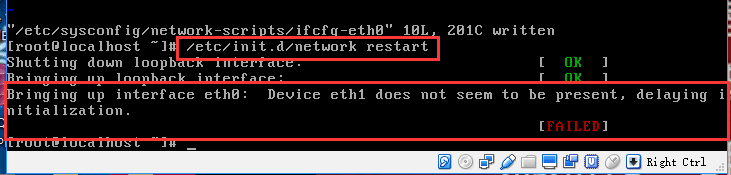
2.配置静态IP

vi /etc/sysconfig/network-scripts/incfg-eth0



刷新网络

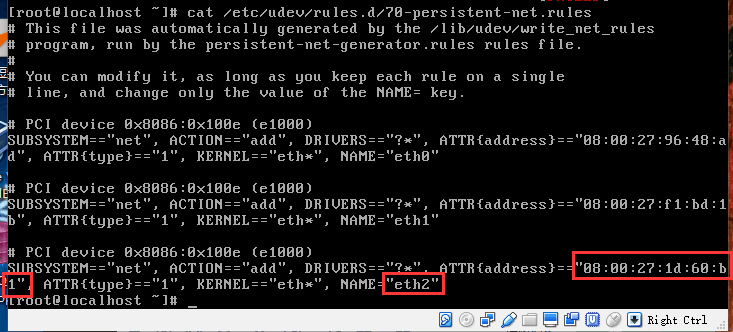
/etc/init.d/network restart



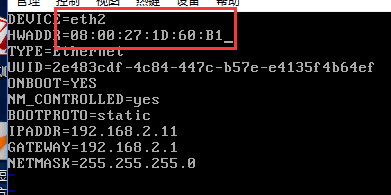
如果出现启动失败请参考以下方法解决

vi /etc/udev/rules.d/70-persistent-net.rules

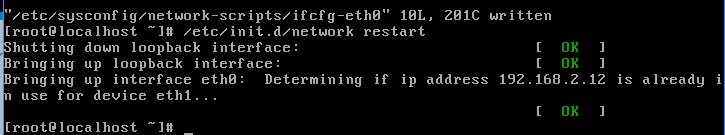
（记下name=xxx, mac=xxx）



将mac地址与name更新到网络配置文件中。

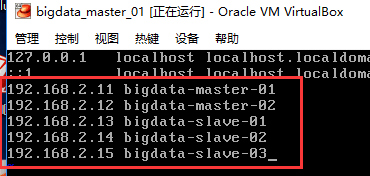


再次重启网络



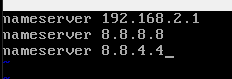
3.配置hosts

vi /etc/hosts



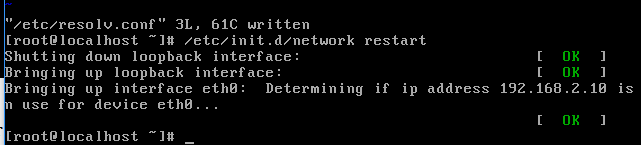
4.配置DNS

vi /etc/resolv.conf



5.重启并验证网络服务

/etc/init.d/network restart

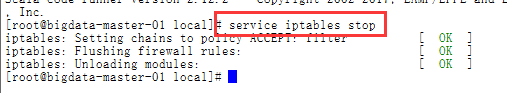


验证网络



6.关闭并禁止防火墙

关闭命令：service iptables stop



永久关闭命令：chkconfig iptables off



查看状态：service iptables status

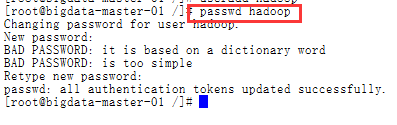


7.在各主机节点创建专门hadoop帐户

1）useradd hadoop (创建用户)



2）passwd hadoop （设置密码）



3）usermod -g root hadoop:为了方便，建议将hadoop加入root用户组，执行完后hadoop即归属于root组了



4）id hadoop 查看输出验证一下，如果看到类似下面的输出：

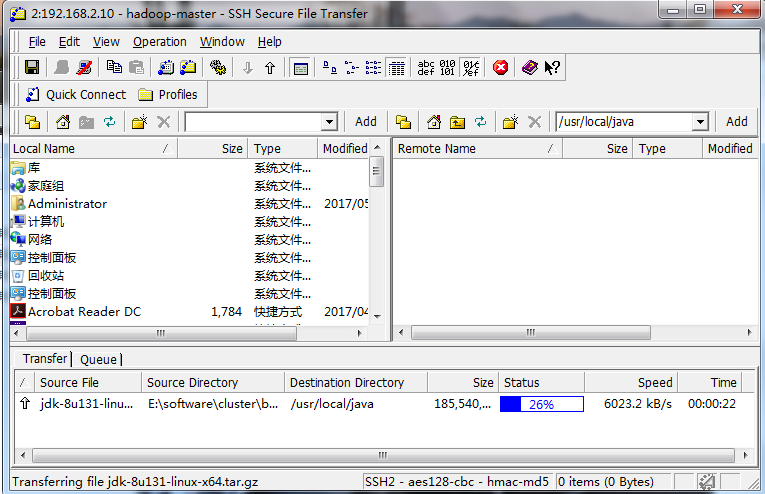
uid=500(hadoop) gid=0(root) 组=0(root)就表示OK了



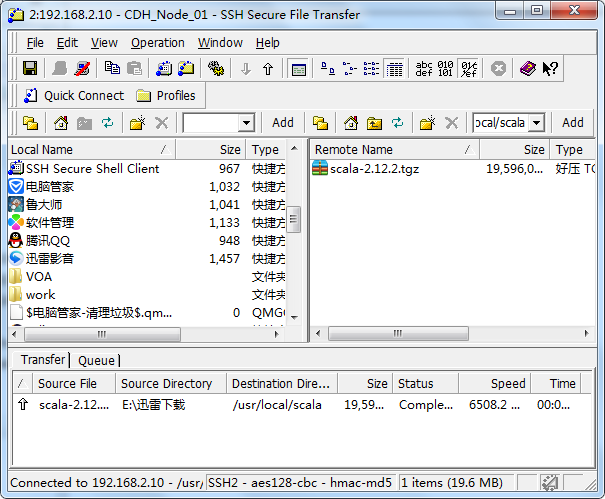
三．安装jdk与scala

1.上传安装包

jdk

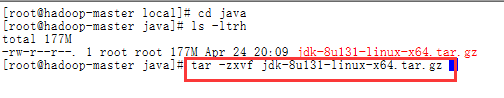


Scala

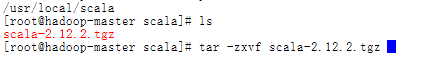


2．解压安装jdk与scala

jdk



scala



3．配置jdk和scala环境变量

jdk

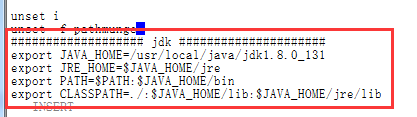
vi /etc/profile

export JAVA\_HOME=/usr/local/java/jdk1.8.0\_131

export JRE\_HOME=$JAVA\_HOME/jre

export PATH=$PATH:$JAVA\_HOME/bin

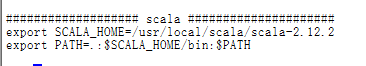
export CLASSPATH=./:$JAVA\_HOME/lib:$JAVA\_HOME/jre/lib



Scala

export SCALA\_HOME=/usr/local/scala/scala-2.12.2

export PATH=.:$SCALA\_HOME/bin:$PATH



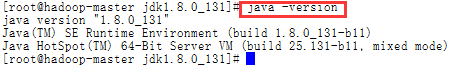
4．使环境变量生效

source /etc/profile



5．验证jdk与scala

Java -version



Scala -version



四．Clone虚拟机并配置

1.关闭当前主机

2.clone 4台主机分别命名为：

bigdata-master-02

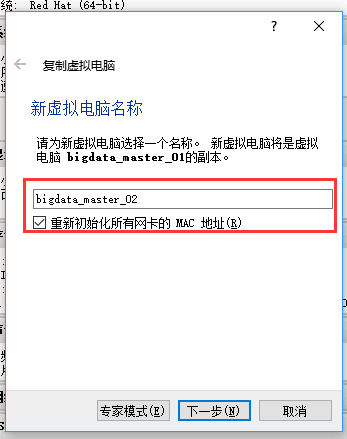
bigdata-slave-01

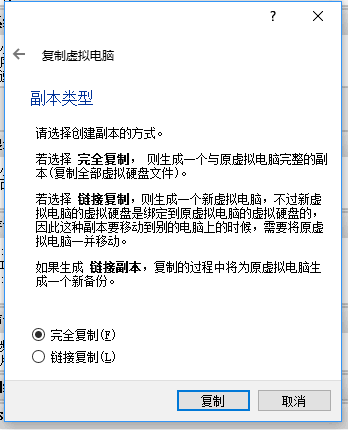
bigdata-slave-02

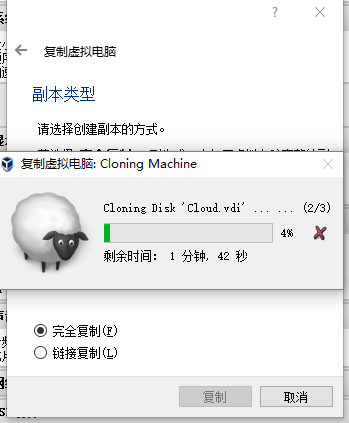
bigdata-slave-03

控制->复制

节点命名为：bigdata-xxx-xx



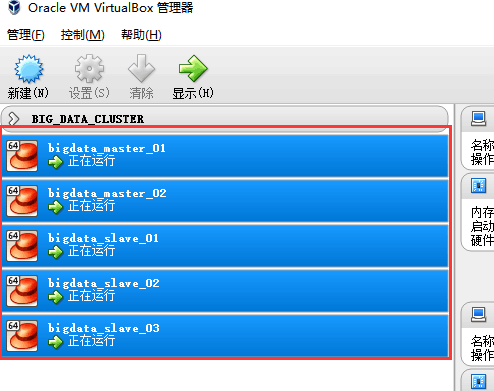




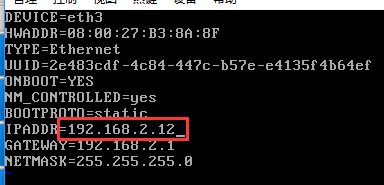
3.依照以上步骤clone其它各节点

4.启动各主机修改IP

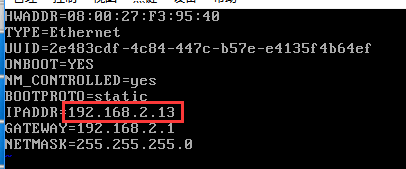
查看各节点



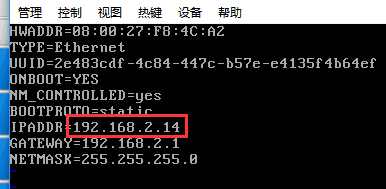
bigdata-master-02



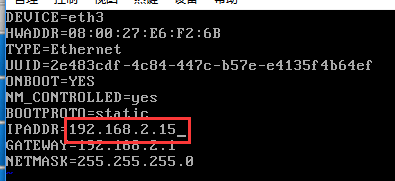
bigdata-slave-01



bigdata-slave-02



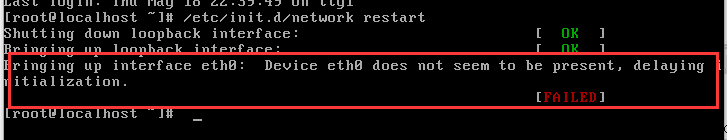
bigdata-slave-03



6.分别重启网络服务

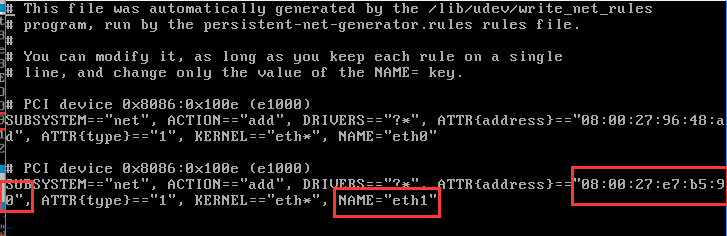
/etc/init.d/network restart

7.如果遇到以下问题，请参考以下解决方案

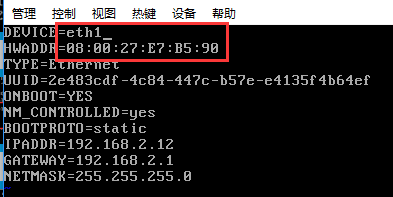


vi /etc/udev/rules.d/70-persistent-net.rules

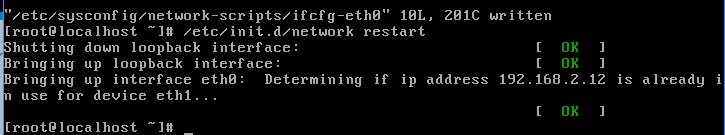
（记下name=thN,mac=08.00.27.e7.b5.90）



将mac地址与name更新到网络配置文件中。



再次重启网络



五．设置各主机节点免密码登录

Master（NameNode | JobTracker）作为客户端，要实现无密码公钥认证，连接到服务器Salve（DataNode | Tasktracker）上时，需要在Master上生成一个密钥对，包括一个公钥和一个私钥，而后将公钥复制到所有的Slave上。当Master通过SSH连接Salve时，Salve就会生成一个随机数并用Master的公钥对随机数进行加密，并发送给Master。Master收到加密数之后再用私钥解密，并将解密数回传给Slave，Slave确认解密数无误之后就允许Master进行连接了。这就是一个公钥认证过程，其间不需要用户手工输入密码。

1.在各节点输入 scp 查找当前系统是否已经安装SSH  


2.如果没有此命令执行下面命令安装：

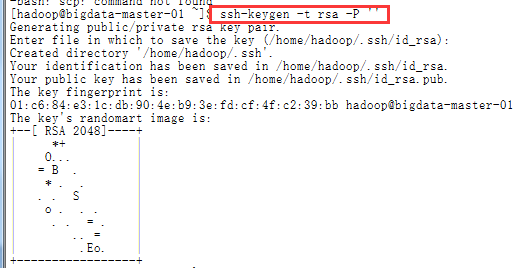
yum install openssh-clients



3.生成秘钥对配置

a) 以hadoop身份登录各节点机器（master-01,master-02,slave-01,slave-02,slave-03）

执行 ssh-keygen -t rsa -P '' 生成公钥、密钥



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

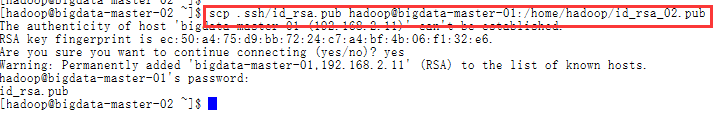
chmod 600 .ssh/authorized\_keys



b)然后用scp命令，把公钥文件发放给master-01节点

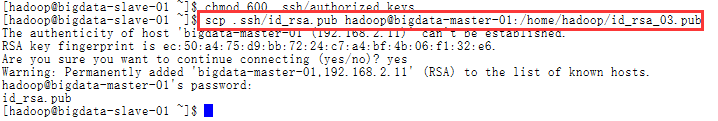
master-02上：

scp .ssh/id\_rsa.pub [hadoop@bigdata-master-01:/home/hadoop/id\_rsa\_02.pub](mailto:hadoop@bigdata-master-01:/home/hadoop/id_rsa_02.pub)



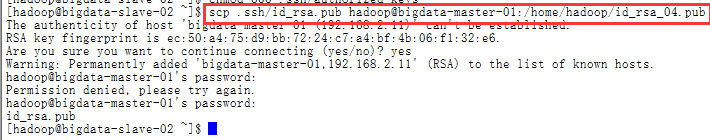
slave-01上：

scp .ssh/id\_rsa.pub [hadoop@bigdata-master-01:/home/hadoop/id\_rsa\_03.pub](mailto:hadoop@bigdata-master-01:/home/hadoop/id_rsa_03.pub)



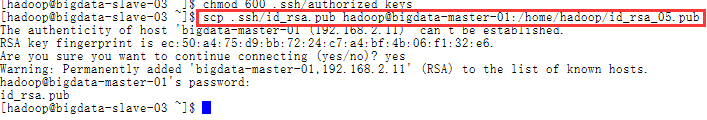
slave-02上：

scp .ssh/id\_rsa.pub [hadoop@bigdata-master-01:/home/hadoop/id\_rsa\_04.pub](mailto:hadoop@bigdata-master-01:/home/hadoop/id_rsa_04.pub)



slave-03上：

scp .ssh/id\_rsa.pub [hadoop@bigdata-master-01:/home/hadoop/id\_rsa\_05.pub](mailto:hadoop@bigdata-master-01:/home/hadoop/id_rsa_05.pub)



以上执行完后，回到master-01中，查看下/home/hadoop目录，应该有4个新文件id\_rsa\_02.pub、id\_rsa\_03.pub、id\_rsa\_04.pub、id\_rsa\_05.pub

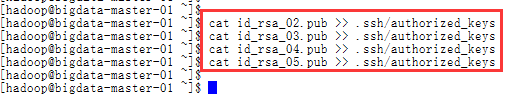
然后在master-01上，导入这4个公钥

cat id\_rsa\_02.pub >> .ssh/authorized\_keys

cat id\_rsa\_03.pub >> .ssh/authorized\_keys

cat id\_rsa\_04.pub >> .ssh/authorized\_keys

cat id\_rsa\_05.pub >> .ssh/authorized\_keys



这样，master-01这台机器上，就有所有5台机器的公钥了。

4.将master-01上的“最全”公钥，复制到其它机器

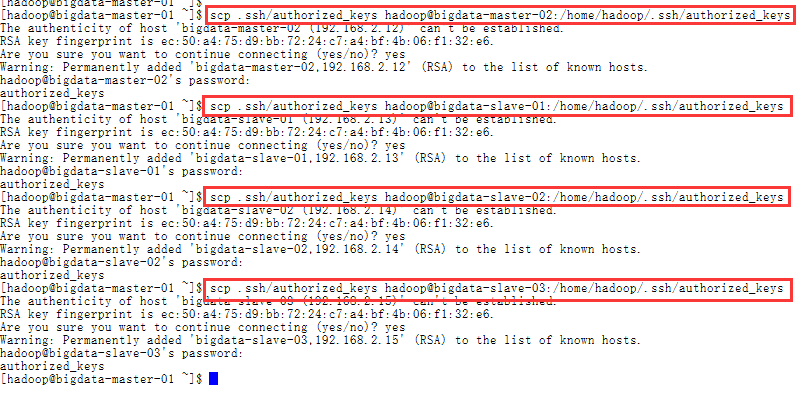
a) 继续保持在master-01上，执行以下命令：

scp .ssh/authorized\_keys [hadoop@bigdata-master-02:/home/hadoop/.ssh/authorized\_keys](mailto:hadoop@bigdata-master-02:/home/hadoop/.ssh/authorized_keys)

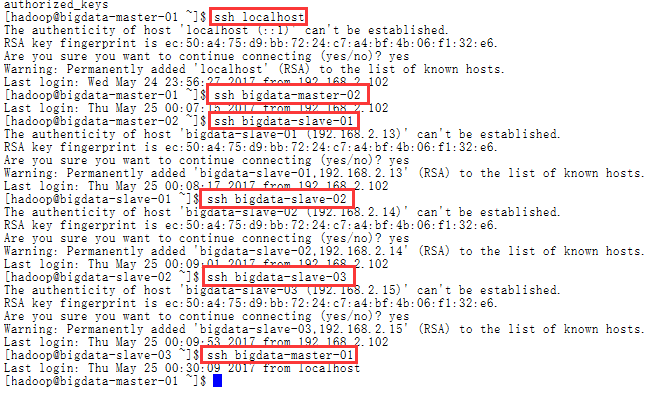
scp .ssh/authorized\_keys hadoop@bigdata-slave-01:/home/hadoop/.ssh/authorized\_keys

scp .ssh/authorized\_keys [hadoop@bigdata-slave-02:/home/hadoop/.ssh/authorized\_keys](mailto:hadoop@bigdata-slave-02:/home/hadoop/.ssh/authorized_keys)

scp .ssh/authorized\_keys hadoop@bigdata-slave-03:/home/hadoop/.ssh/authorized\_keys



b)验证免密SSH



c)创建大数据集群目录结构

在master-01创建以下目录

mkdir /home/hadoop/bigdata（主路径）



mkdir /home/hadoop/bigdata/backup（安装包备份）



mkdir /home/hadoop/bigdata/workspace（集群工作路径）



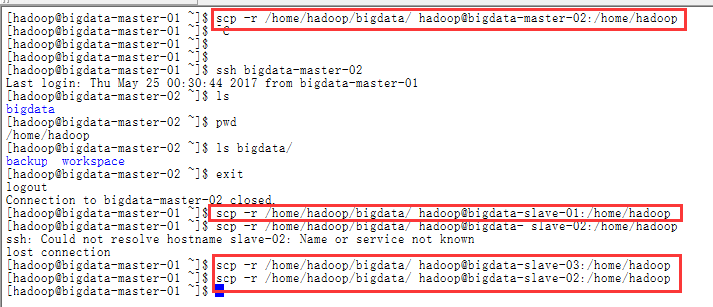
将目录结构复制到其它节点中

scp -r /home/hadoop/bigdata/ hadoop@bigdata-master-02:/home/hadoop

scp -r /home/hadoop/bigdata/ hadoop@bigdata-slave-01:/home/hadoop

scp -r /home/hadoop/bigdata/ hadoop@bigdata-slave-02:/home/hadoop

scp -r /home/hadoop/bigdata/ hadoop@bigdata-slave-03:/home/hadoop



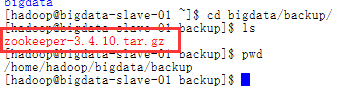
六．zookeeper安装与配置

1.安装机器

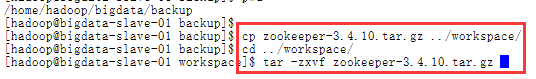
（**bigdata-slave-01，bigdata-slave-02，bigdata-slave-03**）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **192.168.2.13** | **bigdata-slave-01** | **Hadoop**  **Java/Scala**  **zookeeper** | **DataNode**  **NodeManager**  **Zookeeper** | **DataNode**  **NodeManager**  **JournalNode**  **QuorumPeerMain** |
| **192.168.2.14** | **bigdata-slave-02** | **Hadoop**  **Java/Scala**  **zookeeper** | **DataNode**  **NodeManager**  **Zookeeper** | **DataNode**  **NodeManager**  **JournalNode**  **QuorumPeerMain** |
| **192.168.2.15** | **bigdata-slave-03** | **Hadoop**  **Java/Scala**  **zookeeper** | **DataNode**  **NodeManager**  **Zookeeper** | **DataNode**  **NodeManager**  **JournalNode**  **QuorumPeerMain** |

2.上传安装包



3.在workspace目录中解压



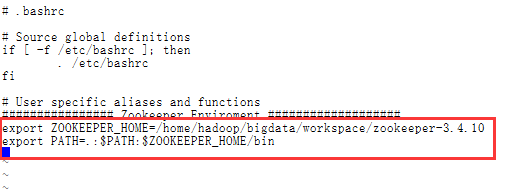
4.为zookeeper配置环境变量

vi ~/.bashrc

################ Zookeeper Enviroment ###################

export ZOOKEEPER\_HOME=/home/hadoop/bigdata/workspace/zookeeper-3.4.10

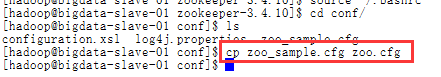
export PATH=.:$PATH:$ZOOKEEPER\_HOME/bin



5.刷新配置

source ~/.bashrc

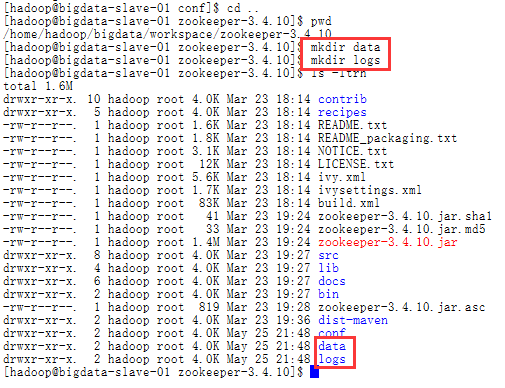
6. 复制一份zookeeper配置文件



在zookeeperk中创建工作目录

mkdir data

mkdir logs



7.修改配置文件

保存数据目录

dataDir=/home/hadoop/bigdata/workspace/zookeeper-3.4.10/data

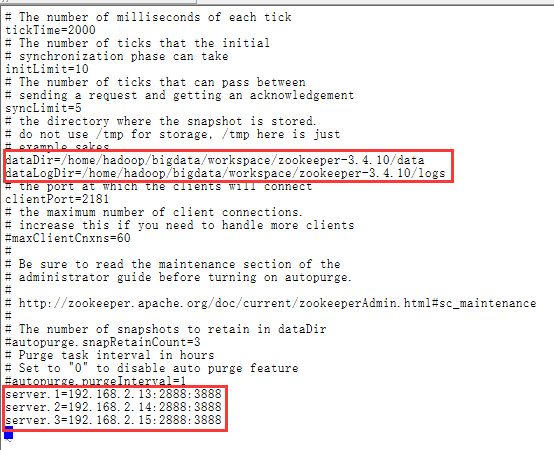
保存日志目录

dataLogDir=/home/hadoop/bigdata/workspace/zookeeper-3.4.10/logs

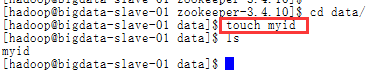
server.1=192.168.2.13:2888:3888

server.2=192.168.2.14:2888:3888

server.3=192.168.2.15:2888:3888

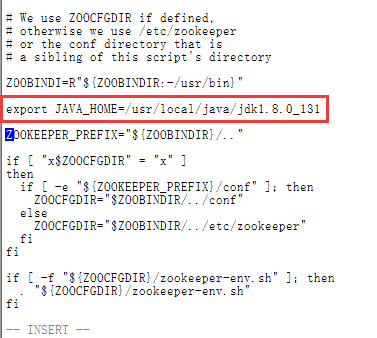


8.在zookeeper-3.4.10/data目录下创建一个文件：myid



9．为了方便集群管理脚本启动设置zkEnv.sh

vi /home/hadoop/bigdata/workspace/zookeeper-3.4.10/bin/zkEnv.sh



10.将zookeeper复制到slave-02,slave-03机器中

scp -r /home/hadoop/bigdata/workspace/zookeeper-3.4.10 hadoop@bigdata-slave-02:/home/hadoop/bigdata/workspace

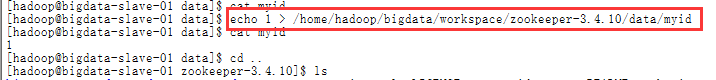


scp -r /home/hadoop/bigdata/workspace/zookeeper-3.4.10 hadoop@bigdata-slave-03:/home/hadoop/bigdata/workspace

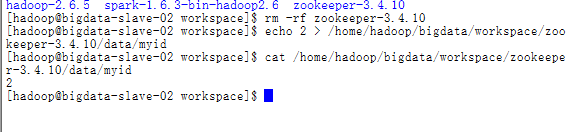


12．分别设置各节点zookeeper下data目录中的myid

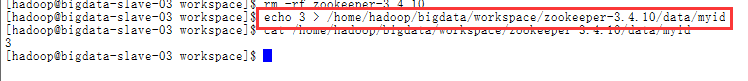
在slave-01 节点中 执行：echo 1 > /home/hadoop/bigdata/workspace/zookeeper-3.4.10/data/myid



在slave-02 节点中 执行：echo 2 > /home/hadoop/bigdata/workspace/zookeeper-3.4.10/data/myid



在slave-03 节点中 执行：echo 3 > /home/hadoop/bigdata/workspace/zookeeper-3.4.10/data/myid



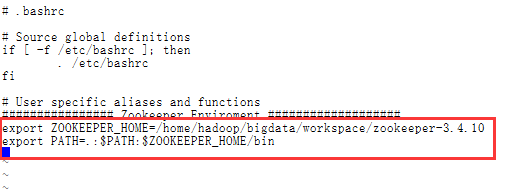
13.分别修改slave-02与slave-03中的环境变量文件并刷新保存

vi ~/.bashrc

################ Zookeeper Enviroment ###################

export ZOOKEEPER\_HOME=/home/hadoop/bigdata/workspace/zookeeper-3.4.10

export PATH=.:$PATH:$ZOOKEEPER\_HOME/bin

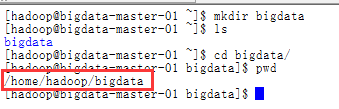


刷新配置

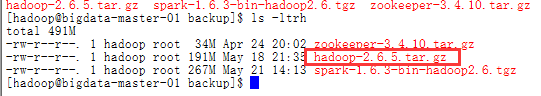
source ~/.bashrc

七．hadoop安装与配置

1. 创建集群根目录

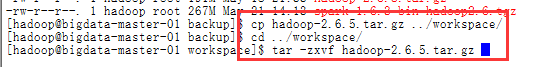


1.上传hadoop安装包



2.解压hadoop安装包到workspace中

tar -zxvf hadoop-2.6.5.tar.gz



3.配置hadoop环境变量（hadoop用户级别）

vi ~/.bashrc

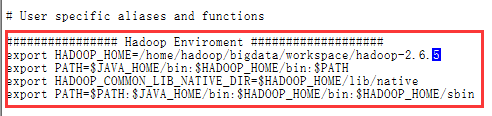
################ Hadoop Enviroment ###################

export HADOOP\_HOME=/home/hadoop/bigdata/workspace/hadoop-2.6.5

export PATH=$JAVA\_HOME/bin:$HADOOP\_HOME/bin:$PATH

export HADOOP\_COMMON\_LIB\_NATIVE\_DIR=$HADOOP\_HOME/lib/native

export PATH=$PATH:$JAVA\_HOME/bin:$HADOOP\_HOME/bin:$HADOOP\_HOME/sbin



刷新配置

source ~/.bashrc

4.在hadoop目录中创建以下工作目录

mkdir tmp

mkdir dfs

mkdir logs

mkdir logs/hadoop

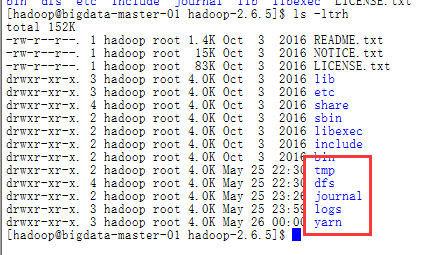
mkdir journal

mkdir dfs/data

mkdir dfs/name

mkdir yarn

mkdir yarn/local



5.修改hadoop配置文件

一共有7个文件要修改：

$HADOOP\_HOME/etc/hadoop/hadoop-env.sh

$HADOOP\_HOME/etc/hadoop/yarn-env.sh

$HADOOP\_HOME/etc/hadoop/core-site.xml

$HADOOP\_HOME/etc/hadoop/hdfs-site.xml

$HADOOP\_HOME/etc/hadoop/mapred-site.xml

$HADOOP\_HOME/etc/hadoop/yarn-site.xml

$HADOOP\_HOME/etc/hadoop/slaves

其中$HADOOP\_HOME表示hadoop根目录，本文中默认为/home/hadoop/bigdata/workspace/hadoop-2.6.5

a) 修改hadoop-env.sh 、yarn-env.sh

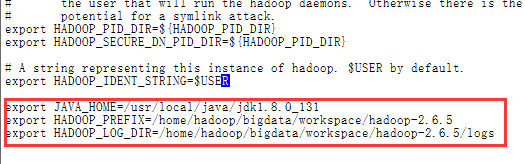
这二个文件主要是修改JAVA\_HOME后的目录，改成实际本机jdk所在目录位置vi etc/hadoop/hadoop-env.sh （及 vi etc/hadoop/yarn-env.sh）找到下面这行的位置，改成（jdk目录位置，大家根据实际情况修改）

export JAVA\_HOME=/usr/local/java/jdk1.8.0\_131

export HADOOP\_PREFIX=/home/hadoop/bigdata/workspace/hadoop-2.6.5

export HADOOP\_LOG\_DIR=/home/hadoop/bigdata/workspace/hadoop-2.6.5/logs

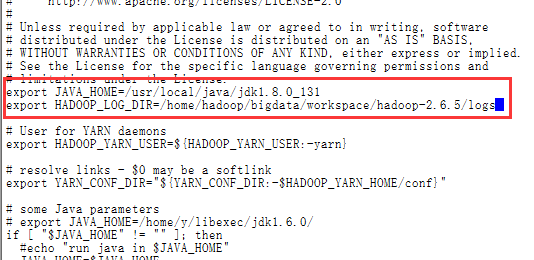
vi /home/hadoop/bigdata/workspace/hadoop-2.6.5/etc/hadoop/hadoop-env.sh



export JAVA\_HOME=/usr/local/java/jdk1.8.0\_131

export HADOOP\_PREFIX=/home/hadoop/bigdata/workspace/hadoop-2.6.5

vi /home/hadoop/bigdata/workspace/hadoop-2.6.5/etc/hadoop/yarn-env.sh



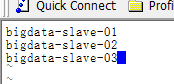
b) **配置文件:slaves**

bigdata-slave-01

bigdata-slave-02

bigdata-slave-03

vi slave



c) core-site.xml 参考下面的内容修改：

<configuration>

　　<!--这里的值指的是默认的HDFS路径。该值来自于hdfs-site.xml中的配置-->

<property>

<name>fs.defaultFS</name>

<value>hdfs://bigdata</value>

</property>

<!-- 指定hadoop临时目录 -->

<property>

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

<value>/home/hadoop/bigdata/workspace/hadoop-2.6.5/tmp</value>

<description>Abase for other temporary directories.</description>

</property>

<!--指定可以在任何IP访问-->

<property>

<name>hadoop.proxyuser.hduser.hosts</name>

<value>\*</value>

</property>

<!--指定所有用户可以访问-->

<property>

<name>hadoop.proxyuser.hduser.groups</name>

<value>\*</value>

</property>

<!--这里是ZooKeeper集群的地址和端口。注意，数量一定是奇数，且不少于三个节点-->

<property>

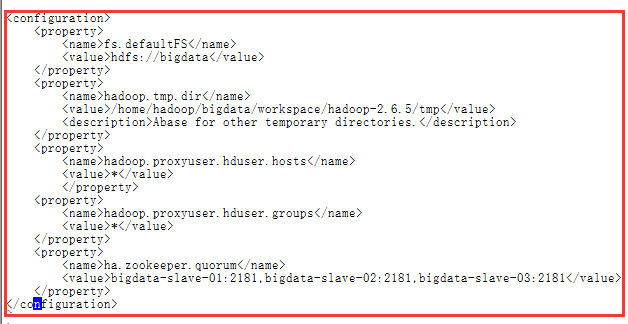
<name>ha.zookeeper.quorum</name>

<value>bigdata-slave-01:2181,bigdata-slave-02:2181,bigdata-slave-03:2181</value>

</property>

</configuration>

vi /home/hadoop/bigdata/workspace/hadoop-2.6.5/etc/hadoop/core-site.xml



d) hdfs-site.xml参考下面的内容修改：

<configuration>

<!--指定hdfs的集群名为bigdata，需要和core-site.xml中的保持一致 -->

<property>

<name>dfs.nameservices</name>

<value>bigdata</value>

</property>

<!-- ns1下面有两个NameNode，分别是nameNode1，nameNode2 -->

<property>

<name>dfs.ha.namenodes.bigdata</name>

<value>nameNode1, nameNode2</value>

</property>

<!-- nameNode1的RPC通信地址 -->

<property>

<name>dfs.namenode.rpc-address.bigdata.nameNode1</name>

<value>bigdata-master-01:9000</value>

</property>

<!-- nameNode2的RPC通信地址 -->

<property>

<name>dfs.namenode.rpc-address.bigdata.nameNode2</name>

<value>bigdata-master-02:9000</value>

</property>

<!-- nameNode1的http通信地址 -->

<property>

<name>dfs.namenode.http-address.bigdata.nameNode1</name>

<value>bigdata-master-01:50070</value>

</property>

<!-- nameNode2的http通信地址 -->

<property>

<name>dfs.namenode.http-address.bigdata.nameNode2</name>

<value>bigdata-master-02:50070</value>

</property>

<!--指定JournalNode集群在对NameNode的目录进行共享时，自己存储数据的磁盘路径-->

<property>

<name>dfs.journalnode.edits.dir</name>

<value>/home/hadoop/bigdata/workspace/hadoop-2.6.5/journal</value>

</property>

<!-- 指定cluster1的两个NameNode共享edits文件目录时，使用的JournalNode集群信息-->

<property>

<name>dfs.namenode.shared.edits.dir</name>

<value>qjournal://bigdata-slave-01:8485;bigdata-slave-02:8485;bigdata-slave-03:8485/bigdata</value>

</property>

<!-- qj方式共享edits。使用此方式-->

<property>

<name>dfs.namenode.edits.journal-plugin.qjournal</name>

<value>org.apache.hadoop.hdfs.qjournal.client.QuorumJournalManager</value>

</property>

<!--开启NameNode失败自动切换-->

<property>

<name>dfs.ha.automatic-failover.enabled</name>

<value>true</value>

</property>

<!--设置为true，允许NN尝试恢复之前失败的dfs.namenode.name.dir目录。在创建checkpoint是做此尝试。如果设置多个磁盘，建议允许-->

<property>

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

<value>true</value>

</property>

<!--指定出故障时，哪个实现类负责执行故障切换-->

<property>

<name>dfs.client.failover.proxy.provider.bigdata</name>

<value>org.apache.hadoop.hdfs.server.namenode.ha.ConfiguredFailoverProxyProvider</value>

</property>

<!--一旦需要NameNode切换，使用ssh方式进行操作-->

<property>

<name>dfs.ha.fencing.methods</name>

<value>sshfence</value>

</property>

<!--如果使用ssh进行故障切换，使用ssh通信时用的密钥存储的位置-->

<property>

<name>dfs.ha.fencing.ssh.private-key-files</name>

<value>/home/hadoop/.ssh/id\_rsa</value>

</property>

<!--指定namenode名称空间的存储地址, 可以是按逗号分隔的目录列表-->

<property>

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

<value>file:/home/hadoop/bigdata/workspace/hadoop-2.6.5/dfs/name</value>

</property>

<!--指定datanode数据存储地址, 可以是按逗号分隔的目录列表-->

<property>

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

<value>file:/home/hadoop/bigdata/workspace/hadoop-2.6.5/dfs/data</value>

</property>

<!--指定数据冗余份数,不超过机器数即可-->

<property>

<name>dfs.replication</name>

<value>3</value>

</property>

<!--指定可以通过web访问hdfs目录-->

<property>

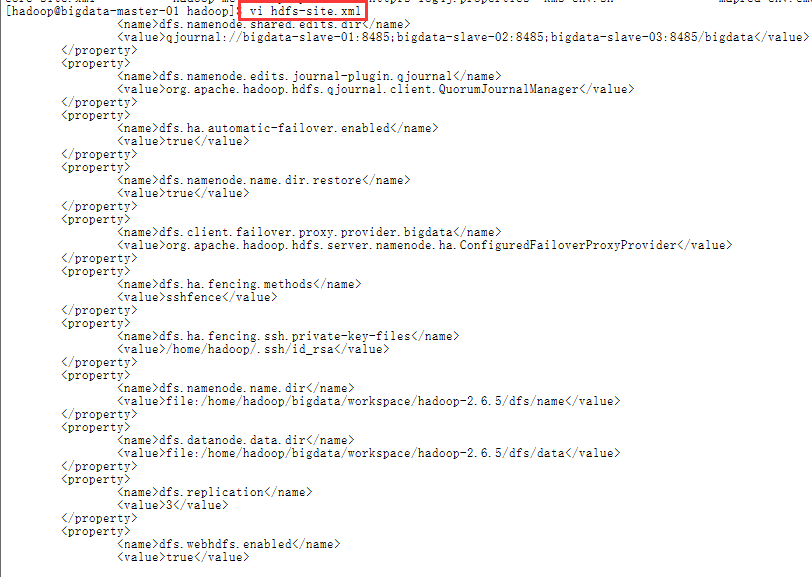
<name>dfs.webhdfs.enabled</name>

<value>true</value>

</property>

</configuration>

vi /home/hadoop/bigdata/workspace/hadoop-2.6.5/etc/hadoop/hdfs-site.xml



d) mapred-site.xml参考下面的内容修改：

<configuration>

<!-- 配置 MapReduce Applications -->

<property>

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

<value>yarn</value>

</property>

<!-- 配置 MapReduce JobHistory Server 地址 ，默认: 0.0.0.0:10020 -->

<property>

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

<value>bigdata-master-01:10020</value>

</property>

<!-- 配置 MapReduce JobHistory Server web ui 地址， 默认: 0.0.0.0:19888 -->

<property>

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

<value>bigdata-master-01:19888</value>

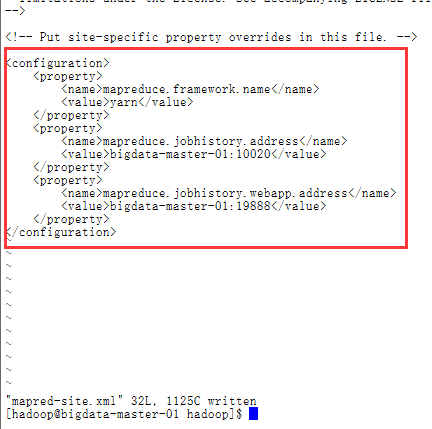
</property>

</configuration>

cd /home/hadoop/bigdata/workspace/hadoop-2.6.5/etc/hadoop/

cp mapred-site.xml.template mapred-site.xml

vi mapred-site.xml



e)yarn-site.xml参考下面的内容修改：

**注意：配置yarn.resourcemanager.ha.id时，在master-01上配置rm1,在master-02上配置rm2,注意：一般都喜欢把配置好的文件远程复制到其它机器上，但这个在YARN的另一个机器上一定要修改，其他机器上不配置此项**

<configuration>

<!--rm失联后重新链接的时间-->

<property>

<name>yarn.resourcemanager.connect.retry-interval.ms</name>

<value>2000</value>

</property>

<!--开启resource manager HA,默认为false-->

<property>

<name>yarn.resourcemanager.ha.enabled</name>

<value>true</value>

</property>

<!--配置resource manager 命名-->

<property>

<name>yarn.resourcemanager.ha.rm-ids</name>

<value>rm1,rm2</value>

</property>

<!--开启resourcemanager故障自动切换，指定机器-->

<property>

<name>yarn.resourcemanager.ha.automatic-failover.enabled</name>

<value>true</value>

</property>

<property>

<name>yarn.resourcemanager.hostname.rm1</name>

<value>bigdata-master-01</value>

</property>

<property>

<name>yarn.resourcemanager.hostname.rm2</name>

<value>bigdata-master-02</value>

</property>

<!--在master-01上配置rm1,在master-02上配置rm2,注意：一般都喜欢把配置好的文件远程复制到其它机器上，但这个在YARN的另一个机器上一定要修改，其他机器上不配置此项-->

<property>

<name>yarn.resourcemanager.ha.id</name>

<value>rm1</value>

<description>If we want to launch more than one RM in single node, we need this configuration</description>

</property>

<!--开启resourcemanager自动恢复功能-->

<property>

<name>yarn.resourcemanager.recovery.enabled</name>

<value>true</value>

</property>

<!--用于持久存储的类。尝试开启-->

<property>

<name>yarn.resourcemanager.store.class</name>

<value>org.apache.hadoop.yarn.server.resourcemanager.recovery.ZKRMStateStore</value>

</property>

<property>

<name>yarn.resourcemanager.zk-address</name>

<value>bigdata-slave-01:2181,bigdata-slave-02:2181,bigdata-slave-03:2181</value>

</property>

<!--失联等待连接时间-->

<property>

<name>yarn.app.mapreduce.am.scheduler.connection.wait.interval-ms</name>

<value>5000</value>

</property>

<!-- 集群的Id，使用该值确保RM不会做为其它集群的active -->

<property>

<name>yarn.resourcemanager.cluster-id</name>

<value>bigdata</value>

</property>

<!--配置rm1-->

<property>

<name>yarn.resourcemanager.address.rm1</name>

<value>bigdata-master-01:8132</value>

</property>

<property>

<name>yarn.resourcemanager.scheduler.address.rm1</name>

<value>bigdata-master-01:8130</value>

</property>

<!-- RM的网页接口地址：端口-->

<property>

<name>yarn.resourcemanager.webapp.address.rm1</name>

<value>bigdata-master-01:8188</value>

</property>

<property>

<name>yarn.resourcemanager.resource-tracker.address.rm1</name>

<value>bigdata-master-01:8131</value>

</property>

<!-- RM管理接口地址：端口-->

<property>

<name>yarn.resourcemanager.admin.address.rm1</name>

<value>bigdata-master-01:8033</value>

</property>

<property>

<name>yarn.resourcemanager.ha.admin.address.rm1</name>

<value>bigdata-master-01:23142</value>

</property>

<!--配置rm2-->

<property>

<name>yarn.resourcemanager.address.rm2</name>

<value>bigdata-master-02:8132</value>

</property>

<property>

<name>yarn.resourcemanager.scheduler.address.rm2</name>

<value>bigdata-master-02:8130</value>

</property>

<!-- RM的网页接口地址：端口-->

<property>

<name>yarn.resourcemanager.webapp.address.rm2</name>

<value>bigdata-master-02:8188</value>

</property>

<property>

<name>yarn.resourcemanager.resource-tracker.address.rm2</name>

<value>bigdata-master-02:8131</value>

</property>

<!-- RM管理接口地址：端口-->

<property>

<name>yarn.resourcemanager.admin.address.rm2</name>

<value>bigdata-master-02:8033</value>

</property>

<property>

<name>yarn.resourcemanager.ha.admin.address.rm2</name>

<value>bigdata-master-02:23142</value>

</property>

<!--请配置为：mapreduce\_shuffle，在Yarn上开启MR的必须项-->

<property>

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

<value>mapreduce\_shuffle</value>

</property>

<property>

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

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

</property>

<!-- nodemanager本地文件存储目录-->

<property>

<name>yarn.nodemanager.local-dirs</name>

<value>/home/hadoop/bigdata/workspace/hadoop-2.6.5/yarn/local</value>

</property>

<!--存储container日志的地方-->

<property>

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

<value>/home/hadoop/bigdata/workspace/hadoop-2.6.5/log/hadoop</value>

</property>

<property>

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

<value>2048</value>

<discription>每个节点可用内存,单位MB</discription>

</property>

<property>

<name>yarn.scheduler.minimum-allocation-mb</name>

<value>258</value>

<discription>单个任务可申请最少内存，默认1024MB</discription>

</property>

<property>

<name>yarn.scheduler.maximum-allocation-mb</name>

<value>2048</value>

<discription>单个任务可申请最大内存，默认8192MB</discription>

</property>

<property>

<name>yarn.log-aggregation-enable</name>

<value>true</value>

<discription>是否启用日志聚集功能</discription>

</property>

</configuration>

**vi yarn-site.xml**



f)复制hadoop工程到其它各节点中

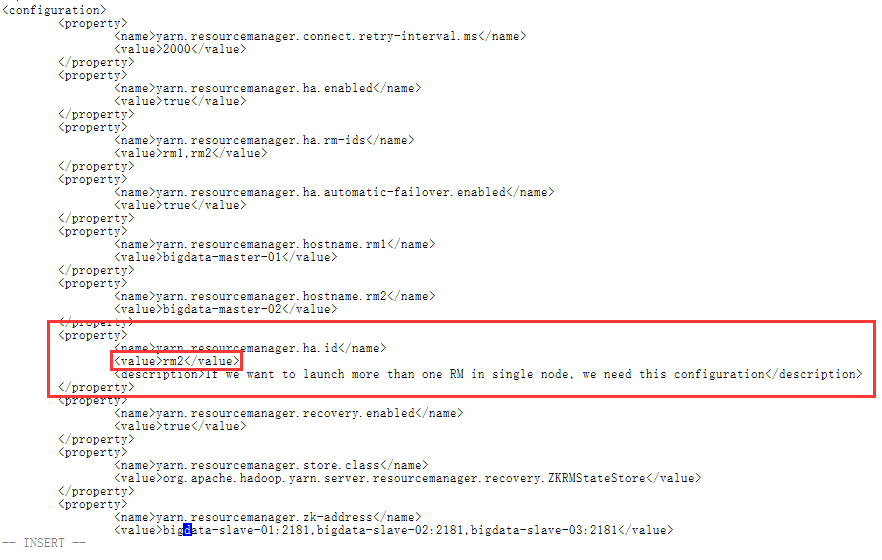
scp -r /home/hadoop/bigdata/workspace/hadoop-2.6.5 hadoop@bigdata-master-02:/home/hadoop/bigdata/workspace

scp -r /home/hadoop/bigdata/workspace/hadoop-2.6.5 hadoop@bigdata-slave-01:/home/hadoop/bigdata/workspace

scp -r /home/hadoop/bigdata/workspace/hadoop-2.6.5 hadoop@bigdata-slave-02:/home/hadoop/bigdata/workspace

scp -r /home/hadoop/bigdata/workspace/hadoop-2.6.5 hadoop@bigdata-slave-03:/home/hadoop/bigdata/workspace

g)修改master-02节点yarn-site.xml文件中arn.resourcemanager.ha.id为rm2



h)配置各节点hadoop的环境变量

################ Hadoop Enviroment ###################

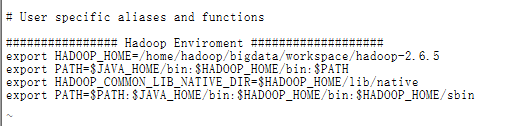
export HADOOP\_HOME=/home/hadoop/bigdata/workspace/hadoop-2.6.5

export PATH=$JAVA\_HOME/bin:$HADOOP\_HOME/bin:$PATH

export HADOOP\_COMMON\_LIB\_NATIVE\_DIR=$HADOOP\_HOME/lib/native

export PATH=$PATH:$JAVA\_HOME/bin:$HADOOP\_HOME/bin:$HADOOP\_HOME/sbin

vi ~/.bashrc



source ~/.bashrc

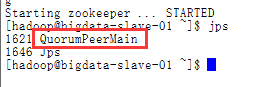
## 启动服务

### 1.启动Zookeeper集群

分别在slave-01,slave-02, slave-03上启动Zookeeper·

zkServer.sh start

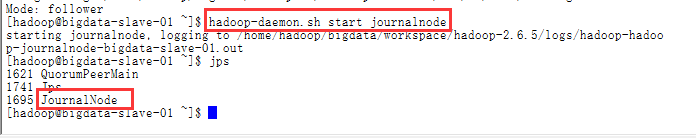
启动成功后slave-01,slave-02, slave-03会有**QuorumPeerMain**进程



### 2.创建命名空间（格式化bigdata）

格式化前,先在各journalnode 节点机器上先启动JournalNode进程

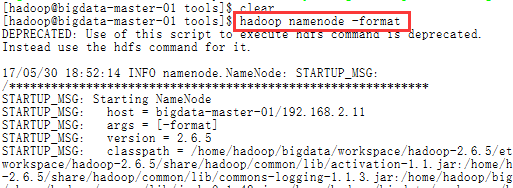
hadoop-daemon.sh start journalnode



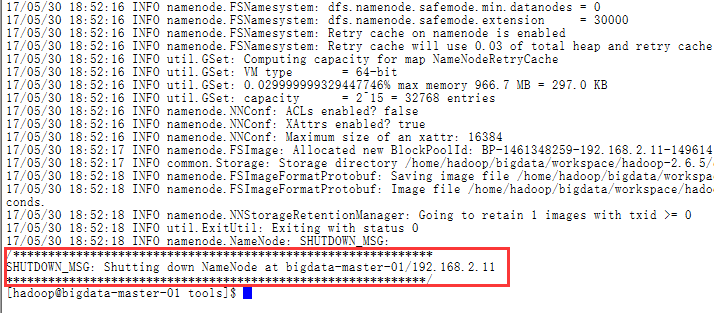
在master-01上执行：

NameNode格式化

hadoop namenode -format

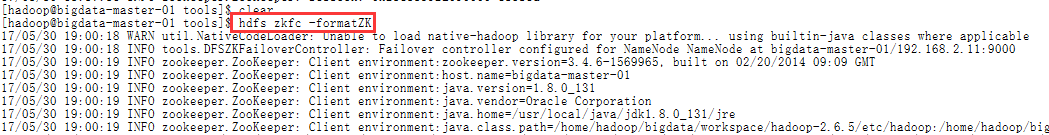


格式化完成

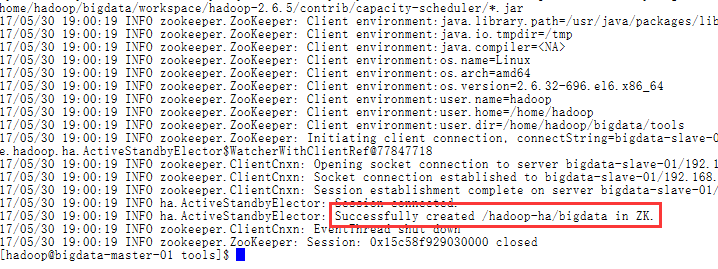


hdfs格式化:

hdfs zkfc -formatZK



格式化完成



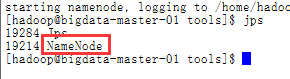
### 3.启动主NameNode节点

在master-01上执行：

hadoop-daemon.sh start namenode



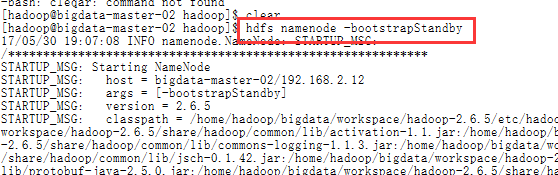
启动成功后master-01会有NameNode进程



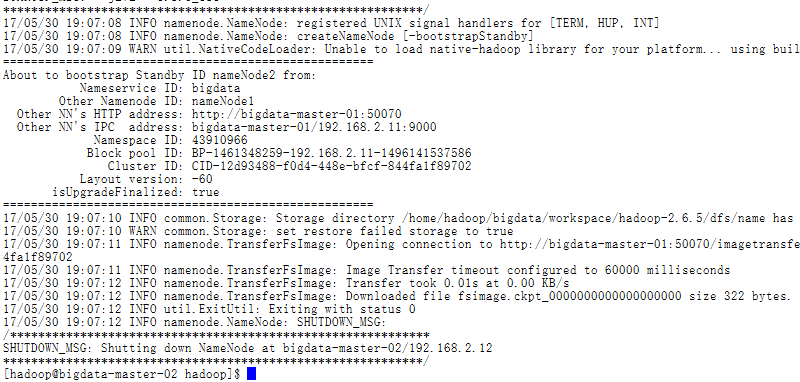
### 4.格式备NameNode节点（namenode2）

在master-02上执行命令：

hdfs namenode –bootstrapStandby



格式化完成



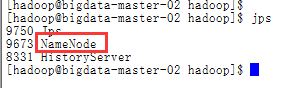
### 5.启动备NameNode节点（namenode2）

在master-02上执行：

hadoop-daemon.sh start namenode

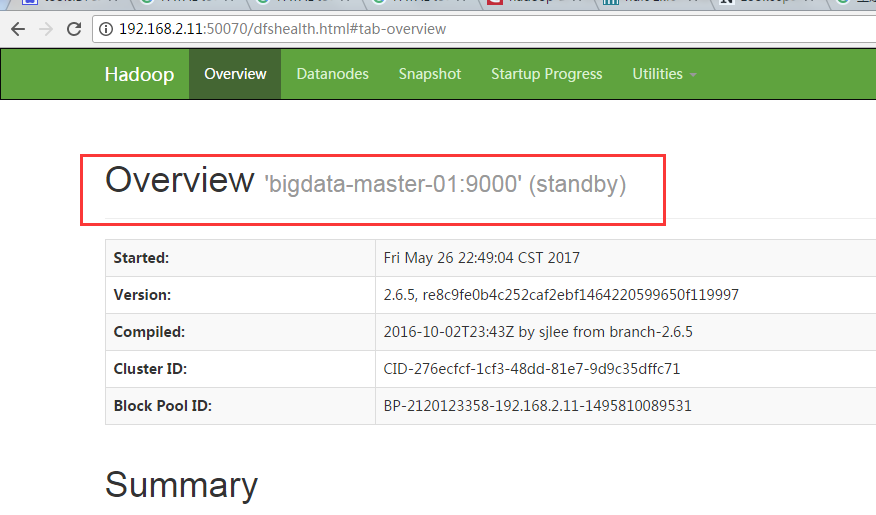


启动成功后master-02会有**NameNode**进程

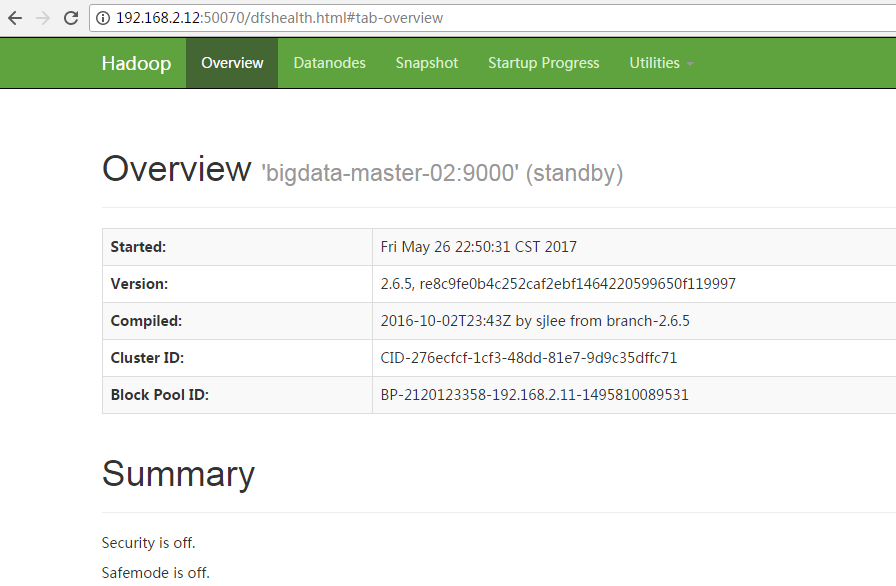


检查UI：

http://192.168.2.11:50070 standby



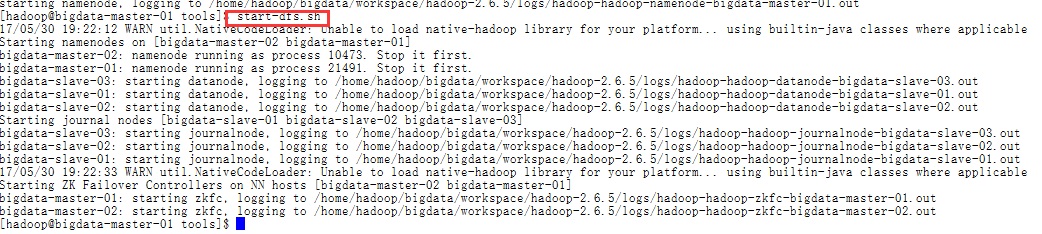
192.168.3.12:50070 standby



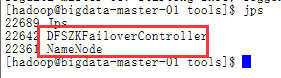
### 6.启动集群

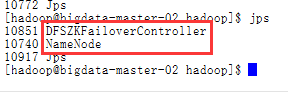
在master-01上执行

start-dfs.sh



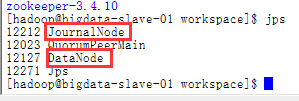
启动成功后master-01，master-02上会有**DFSZKFailoverController**进程

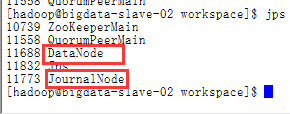


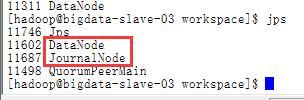


slave-01，slave-02，slave-03会有**JournalNode**进程

slave-01，slave-02，slave-03会有**DataNode**进程

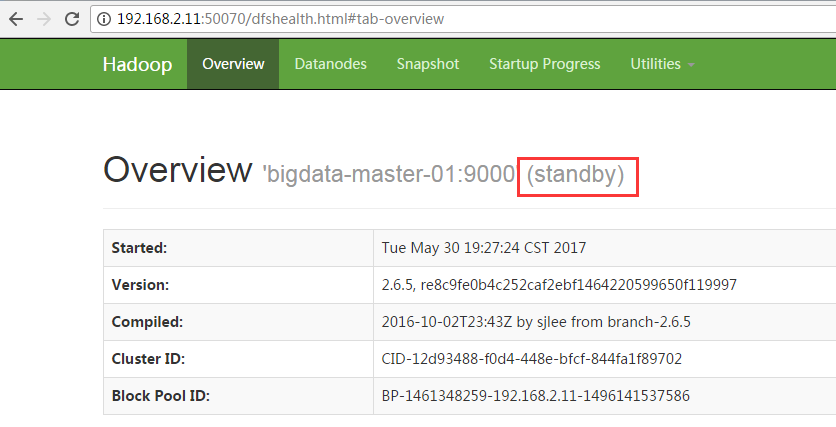




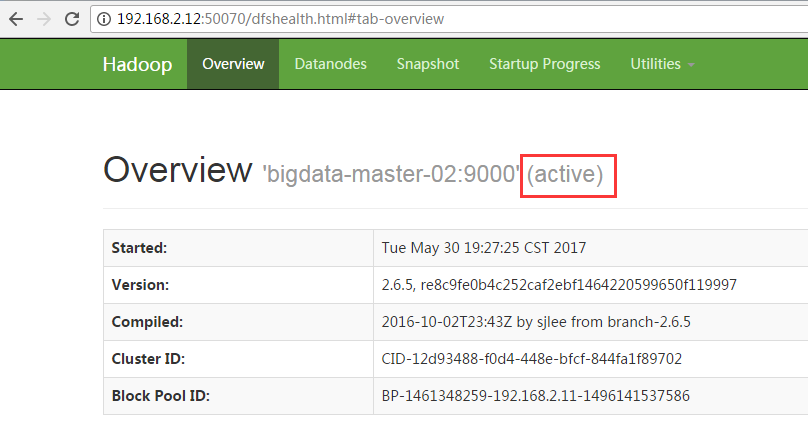


namenode检查：

192.168.3.11:50070 standby



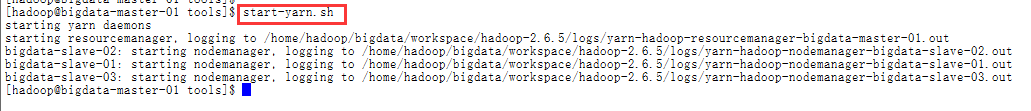
192.168.3.12:50070 active

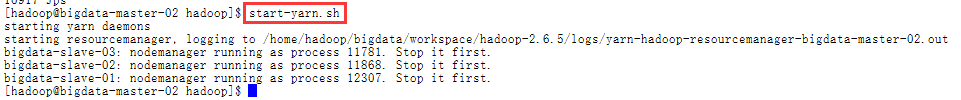


### 7启动YARN（namenode1）

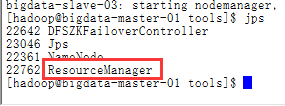
在master-01, master-02上执行：

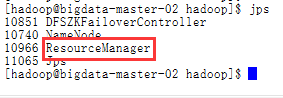
start-yarn.sh



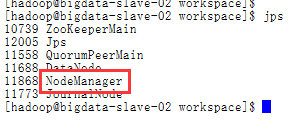


启动成功后master-01, master-02上会有resourcemanager进程



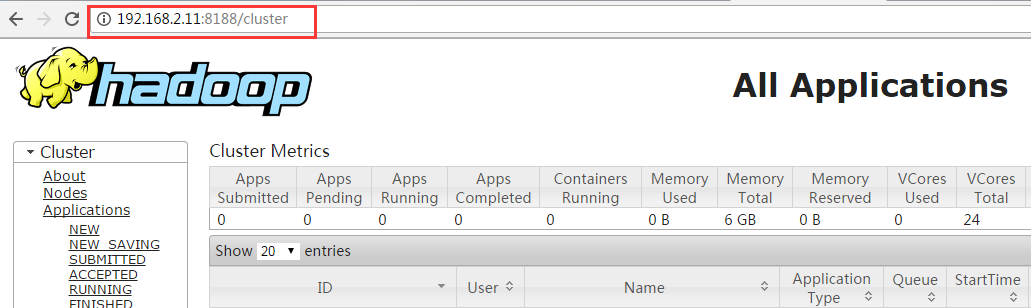


slave-01，slave-02，slave-03上会有nodemanager进程

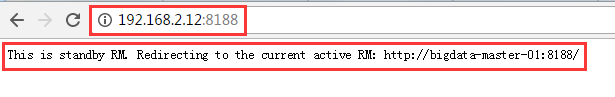


检查：

192.168.2.11:8188 active

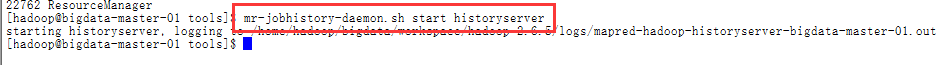


192.168.2.12:8188 standby



**10.启动hostory server**

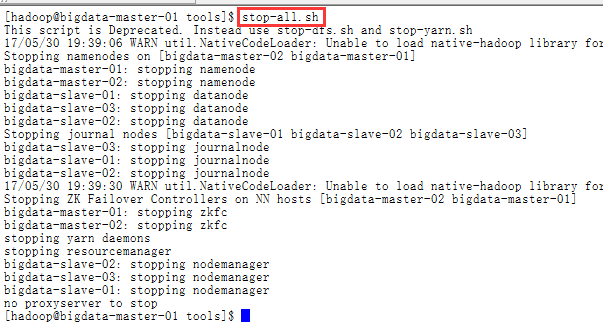
mr-jobhistory-daemon.sh start historyserver



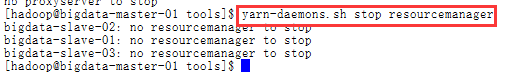
### 11.重启集群

在master-01上执行：

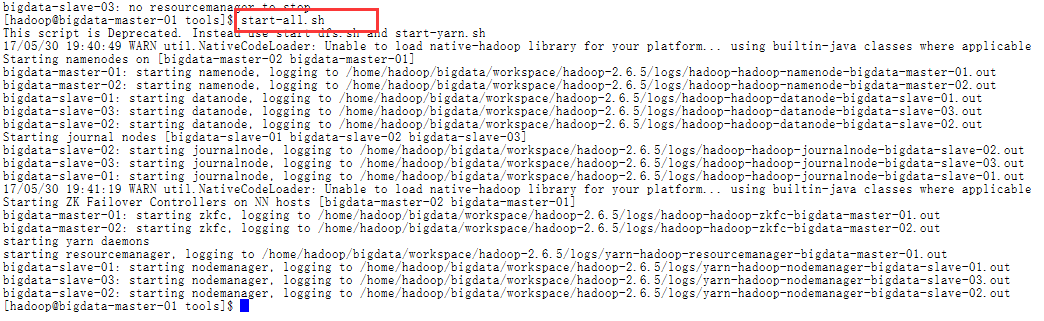
stop-all.sh



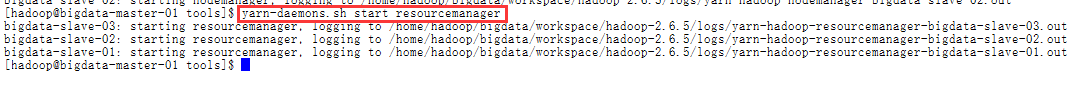
yarn-daemons.sh stop resourcemanager



start-all.sh

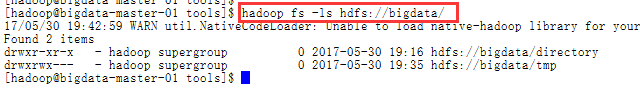


yarn-daemons.sh start resourcemanager

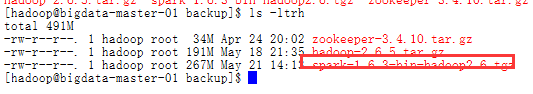


### 10.查看hdfs

hadoop fs -ls hdfs://bigdata/

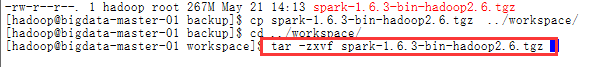


1. spark安装与配置
2. 上传spark安装包



1. 解压spark包

$SPARK\_HOME=/home/hadoop/bigdata/workspace/spark-1.6.3-bin-hadoop2.6



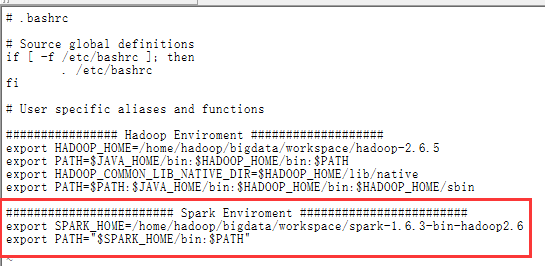
3.修改环境变量

######################## Spark Enviroment ########################

export SPARK\_HOME=/home/hadoop/bigdata/workspace/spark-1.6.3-bin-hadoop2.6

export PATH="$SPARK\_HOME/bin:$PATH"

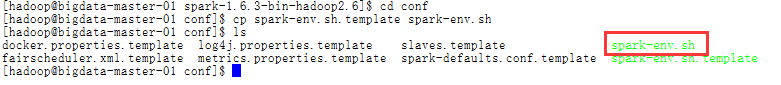
vi ~/.bashrc



4.刷新环境变量

source ~/.bashrc

5.配置$SPARK\_HOME/conf/spark-env.sh文件

cp spark-env.sh.template spark-env.sh

#配置内容如下：

export SCALA\_HOME=/usr/local/scala/scala-2.12.2

export JAVA\_HOME=/usr/local/java/jdk1.8.0\_131

export HADOOP\_HOME=/home/hadoop/bigdata/workspace/hadoop-2.6.5

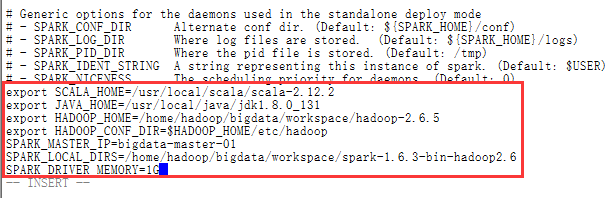
export HADOOP\_CONF\_DIR=$HADOOP\_HOME/etc/hadoop

SPARK\_MASTER\_IP=bigdata-master-01

SPARK\_LOCAL\_DIRS=/home/hadoop/bigdata/workspace/spark-1.6.3-bin-hadoop2.6

SPARK\_DRIVER\_MEMORY=1G

vi /home/hadoop/bigdata/workspace/spark-1.6.3-bin-hadoop2.6/conf/spark-env.sh



## 6．配置Spark History Server

cp spark-defaults.conf.template spark-defaults.conf

vi spark-default.conf

spark.master spark://bigdata-master-01:7077

spark.eventLog.enabled true

spark.eventLog.dir hdfs://bigdata/directory

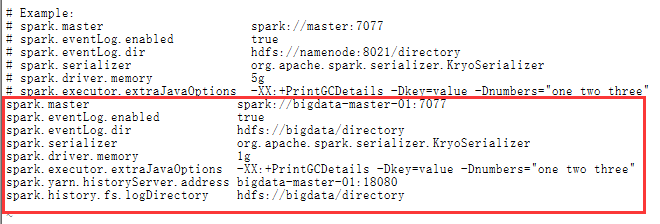
spark.serializer org.apache.spark.serializer.KryoSerializer

spark.driver.memory 1g

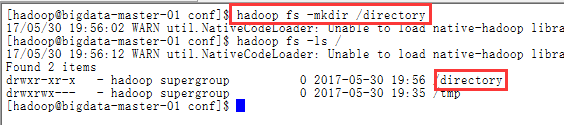
spark.executor.extraJavaOptions -XX:+PrintGCDetails -Dkey=value -Dnumbers="one two three"

spark.yarn.historyServer.address bigdata-master-01:18080

spark.history.fs.logDirectory hdfs://bigdata/directory

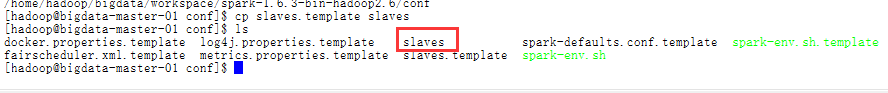


创建directory目录



7.配置$SPARK\_HOME/conf/slaves文件

cp slaves.template slaves

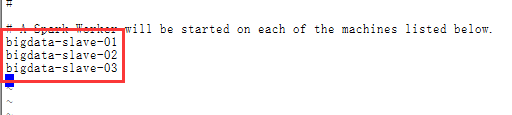


配置内容如下

bigdata-slave-01

bigdata-slave-02

bigdata-slave-03



8. 复制spark到各节点：

将配置好的spark文件复制到bigdata-slave-N节点

scp -r /home/hadoop/bigdata/workspace/spark-1.6.3-bin-hadoop2.6 hadoop@bigdata-master-02:/home/hadoop/bigdata/workspace

scp -r /home/hadoop/bigdata/workspace/spark-1.6.3-bin-hadoop2.6 hadoop@bigdata-slave-01:/home/hadoop/bigdata/workspace

scp -r /home/hadoop/bigdata/workspace/spark-1.6.3-bin-hadoop2.6 hadoop@bigdata-slave-02:/home/hadoop/bigdata/workspace

scp -r /home/hadoop/bigdata/workspace/spark-1.6.3-bin-hadoop2.6 hadoop@bigdata-slave-03:/home/hadoop/bigdata/workspace

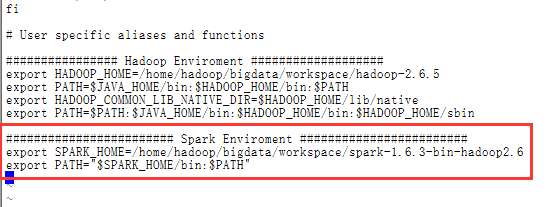
9.修改各节点的hadoop用户的spark环境变量

######################## Spark Enviroment ########################

export SPARK\_HOME=/home/hadoop/bigdata/workspace/spark-1.6.3-bin-hadoop2.6

export PATH="$SPARK\_HOME/bin:$PATH"

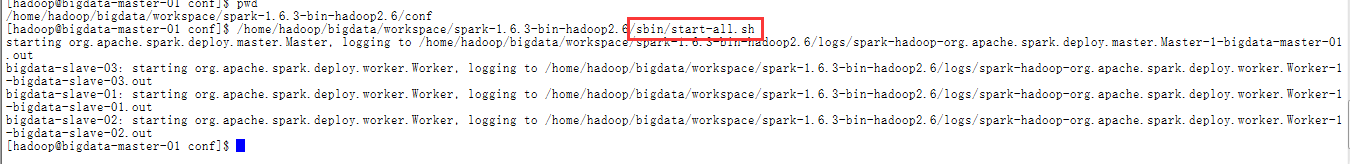
进入环境变量配置文件：vi ~/.bashrc



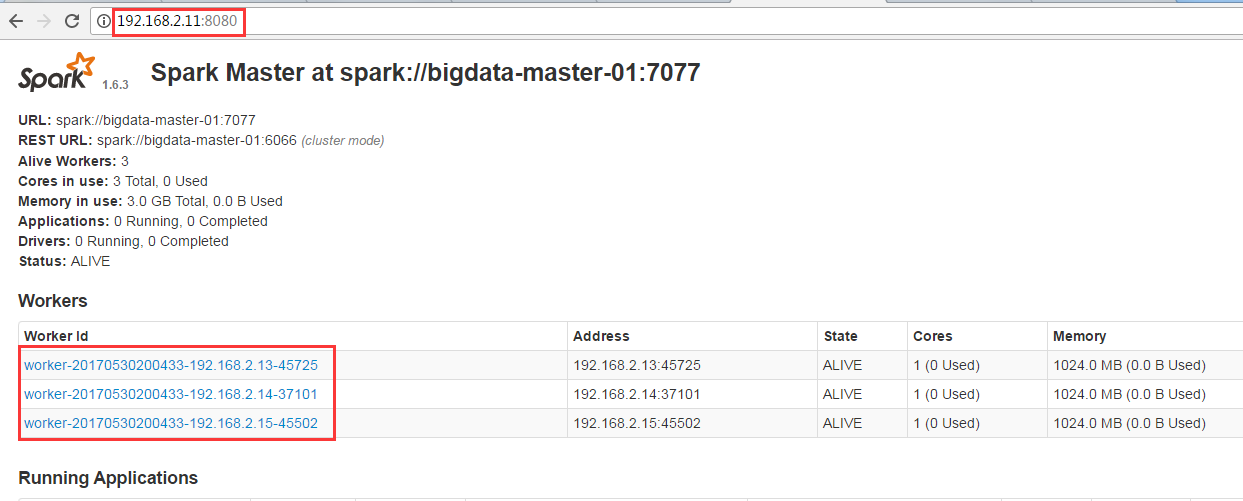
刷新配置：source ~/.bashrc

10.启动Spark

/home/hadoop/bigdata/workspace/spark-1.6.3-bin-hadoop2.6/sbin/start-all.sh

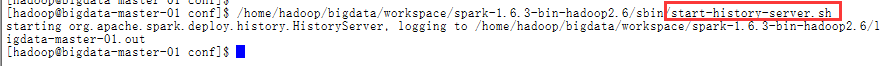


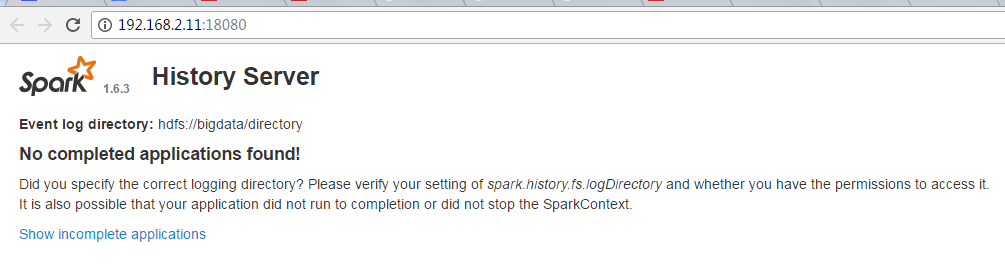
查看



11.启动Spark History Server

/home/hadoop/bigdata/workspace/spark-1.6.3-bin-hadoop2.6/sbin/start-history-server.sh





十．测试集群

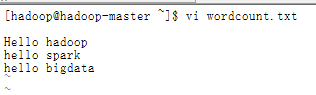
1.测试hadoop

测试的源文件的内容为:

Hello hadoop

hello spark

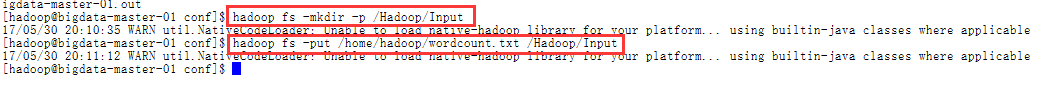
hello bigdata



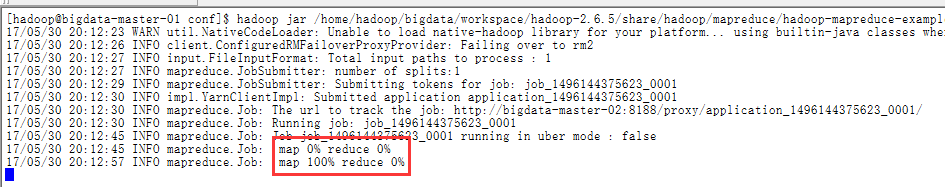
然后执行下列命令：

hadoop fs -mkdir -p /Hadoop/Input

hadoop fs -put /home/hadoop/wordcount.txt /Hadoop/Input

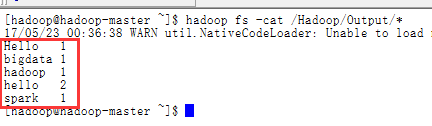


hadoop jar /home/hadoop/bigdata/workspace/hadoop-2.6.5/share/hadoop/mapreduce/hadoop-mapreduce-examples-2.6.5.jar wordcount /Hadoop/Input /Hadoop/Output

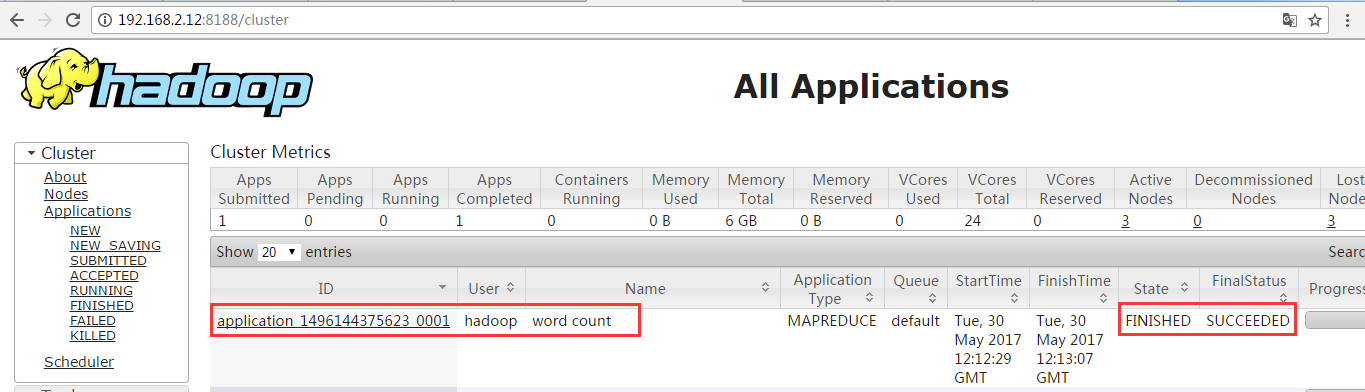


查看hadoop计算结果：

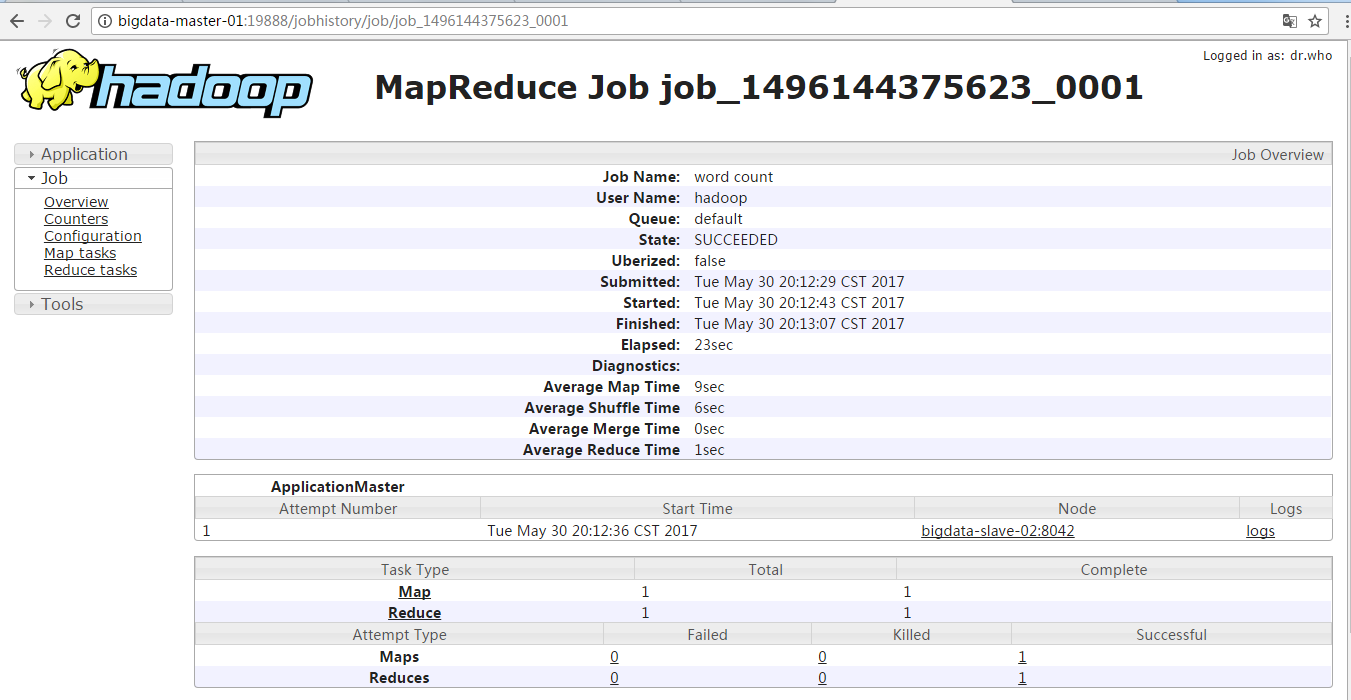
hadoop fs -cat /Hadoop/Output/\*

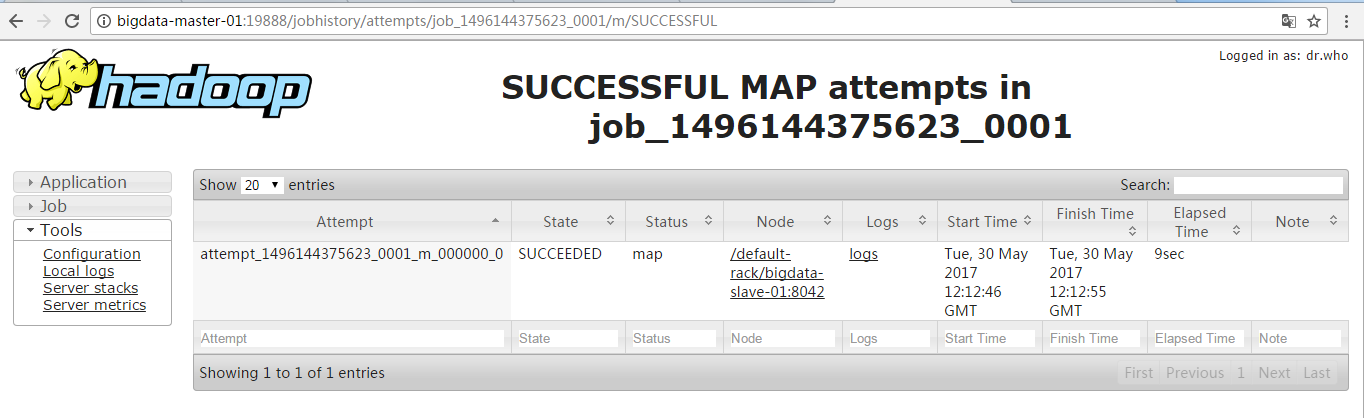


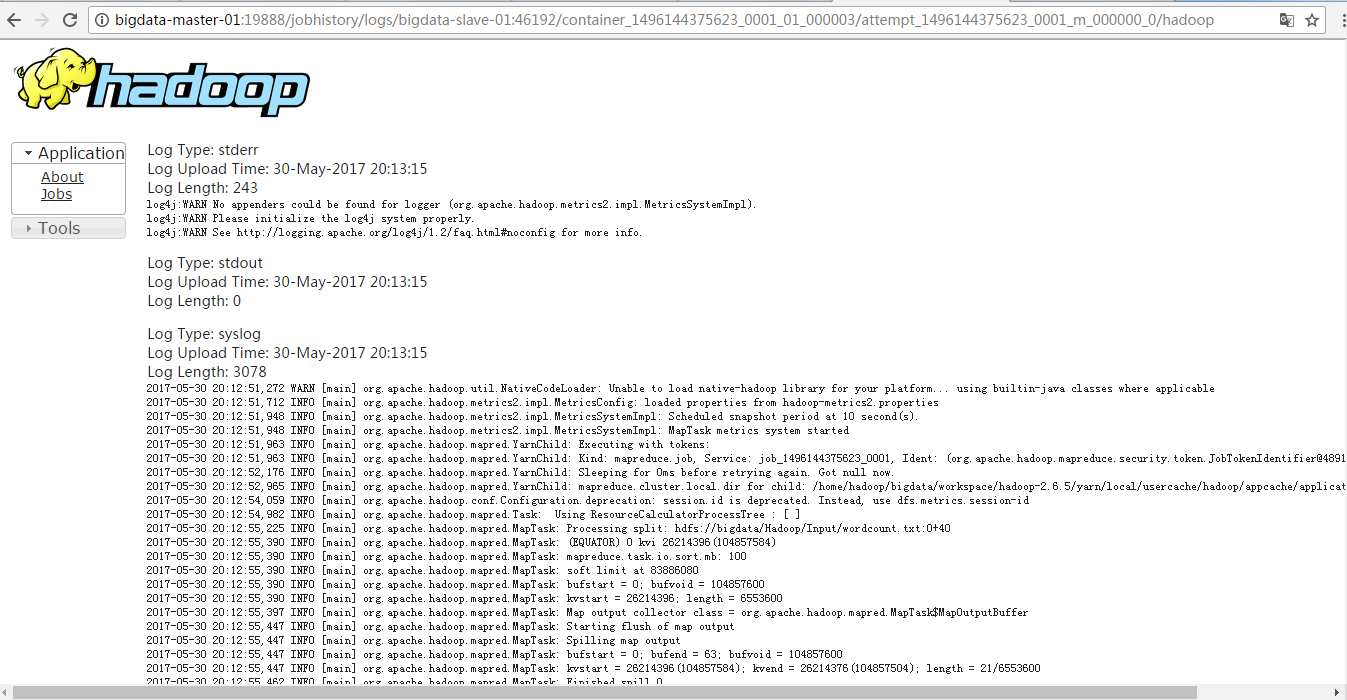
查看UI



查看log记录







hadoop集群搭建成功！

2.测试spark

为了避免麻烦这里我们使用spark-shell，做一个简单的worcount的测试

用于在测试hadoop的时候我们已经在hdfs上存储了测试的源文件，下面就是直接拿来用就好了！

spark-shell

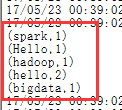
val file=sc.textFile("hdfs://bigdata/Hadoop/Input/wordcount.txt")

val rdd = file.flatMap(line => line.split(" ")).map(word => (word,1)).reduceByKey(\_+\_)

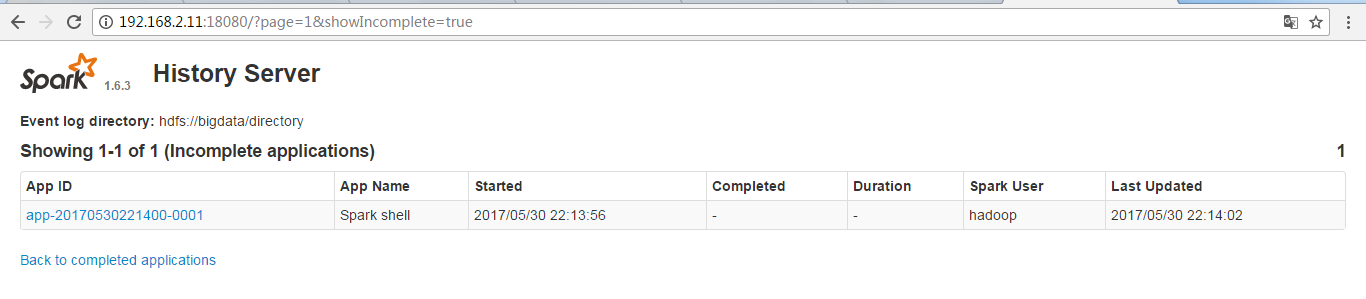
rdd.collect()

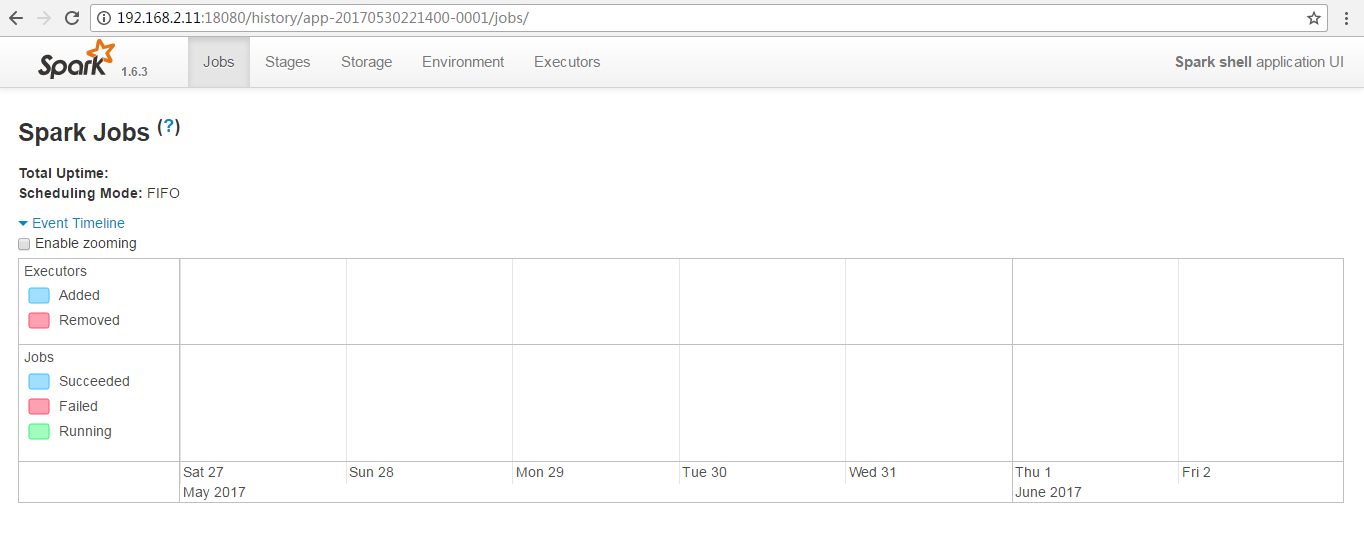
rdd.foreach(println)





查看spark job日志





十一．集群管理

## 1．正确的启动顺序

**1) ZooKeeper -> [Hadoop](http://lib.csdn.net/base/hadoop" \o "Hadoop知识库" \t "_blank) -> Spark**

**2) ZooKeeper -> JournalNode (Hadoop) -> NameNode (Hadoop) -> DataNode (Hadoop) -> 主 ResourceManager/NodeManager (Hadoop) -> 备份 ResourceManager (Hadoop) -> ZKFC (Hadoop) -> MapReduce JobHistory (Hadoop) -> Spark**

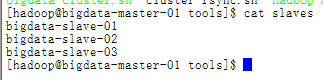
**关闭集群的顺序则相反**

2．脚本部署在下面的路径下

/home/hadoop/bigdata/tools

## 创建Slaves 配置文件

**文件名**：slaves   
**功能**：记录运行 datanode 、nodemanager、HRegionServer、QuorumPeerMain、JournalNode 的节点 IP 地址或主机名   
**内容**



1. 管理脚本

1) ZooKeeper 管理脚本

文件名：zk-manager.sh

功能：启动、关闭与重启 ZooKeeper 集群，并可查看运行 ZK 服务的的模式（leader or follower？）

内容:

#!/bin/bash

SLAVES=$(cat slaves)

start\_time=`date +%s`

for slave in $SLAVES

do

case $1 in

start) ssh -t $slave "source ~/.bash\_profile;zkServer.sh start" 1>/dev/null;;

stop) ssh -t $slave "source ~/.bash\_profile;zkServer.sh stop" 1>/dev/null ;;

status) ssh -t $slave "source ~/.bash\_profile;zkServer.sh status" ;;

restart)ssh -t $slave "source ~/.bash\_profile;zkServer.sh restart" 1>/dev/null;;

\*) echo -e "Usage: sh zk-manager.sh {start|stop|restart} ^\_^\n" && exit ;;

esac

done

end\_time=`date +%s`

elapse\_time=$((${end\_time}-${start\_time}))

echo -e "\n$1 ZooKeeper Server takes ${elapse\_time} seconds\n"

2）JournalNode 管理脚本

文件名：journal-manager.sh

功能：启动、关闭运行在各个 slaves 上的 JournalNode 进程

内容

#!/bin/bash

start\_time=`date +%s`

SLAVES=$(cat slaves)

for slave in $SLAVES

do

case $1 in

start) ssh -t $slave "hadoop-daemon.sh start journalnode" ;;

stop) ssh -t $slave "hadoop-daemon.sh stop journalnode" ;;

\*) echo -e "Usage: sh journal-manager.sh {start|stop} ^\_^\n" && exit ;;

esac

done

end\_time=`date +%s`

elapse\_time=$((${end\_time}-${start\_time}))

echo -e "\n$1 JournalNode Server takes ${elapse\_time} seconds\n"

3） Hadoop 管理脚本

文件名：hadoop-manager.sh

功能：管理 hadoop 的启动与关闭

内容

#!/bin/bash

start\_time=`date +%s`

SLAVES=$(cat slaves)

for slave in $SLAVES

do

case $1 in

start) ssh -t $slave "hadoop-daemon.sh start journalnode" ;;

stop) ssh -t $slave "hadoop-daemon.sh stop journalnode" ;;

\*) echo -e "Usage: sh journal-manager.sh {start|stop} ^\_^\n" && exit ;;

esac

done

end\_time=`date +%s`

elapse\_time=$((${end\_time}-${start\_time}))

echo -e "\n$1 JournalNode Server takes ${elapse\_time} seconds\n"

[hadoop@bigdata-master-01 tools]$ clear

[hadoop@bigdata-master-01 tools]$ cat hadoop-manager.sh

# which machine to be active NameNode

NameNode\_1=bigdata-master-01

# which machine to be standy NameNode

NameNode\_2=bigdata-master-02

# which machine to be active ResourceManager

ResourceManager\_1=bigdata-master-01

# which machine to be standby ResourceManager

ResourceManager\_2=bigdata-master-02

# which machine to be JobHistoryServer

HistoryServer=bigdata-master-01

start\_time=`date +%s`

# make sure which namenode is active and which resourcemanager is active

function getServiceState () {

hdfs haadmin -getServiceState nameNode1 | grep 'active' >> /dev/null && NameNode\_Active=${NameNode1} && NameNode\_Standby=${NameNode2}

hdfs haadmin -getServiceState nameNode2 | grep 'active' >> /dev/null && NameNode\_Active=${NameNode2} && NameNode\_Standby=${NameNode1}

yarn rmadmin -getServiceState rm1 | grep 'active' >> /dev/null && ResourceManager\_Active=${ResourceManager\_1} && ResourceManager\_Standby=${ResourceManager\_2}

yarn rmadmin -getServiceState rm2 | grep 'active' >> /dev/null && ResourceManager\_Active=${ResourceManager\_2} && ResourceManager\_Standby=${ResourceManager\_1}

}

case $1 in

start) ssh -t ${NameNode\_1} "hadoop-daemon.sh start namenode";

ssh -t ${NameNode\_1} "hadoop-daemon.sh start zkfc";

ssh -t ${NameNode\_2} "hadoop-daemon.sh start namenode";

ssh -t ${NameNode\_2} "hadoop-daemon.sh start zkfc";

ssh -t ${NameNode\_1} "hadoop-daemons.sh start datanode";

ssh -t ${ResourceManager\_1} "start-yarn.sh";

ssh -t ${ResourceManager\_2} "yarn-daemon.sh start resourcemanager";

ssh -t ${HistoryServer} "mr-jobhistory-daemon.sh start historyserver";

;;

stop) getServiceState

ssh -t ${HistoryServer} "mr-jobhistory-daemon.sh stop historyserver";

ssh -t ${ResourceManager\_2} "yarn-daemon.sh stop resourcemanager";

ssh -t ${ResourceManager\_1} "stop-yarn.sh";

ssh -t ${NameNode\_1} "hadoop-daemons.sh stop datanode";

ssh -t ${NameNode\_2} "hadoop-daemon.sh stop namenode";

ssh -t ${NameNode\_1} "hadoop-daemon.sh stop namenode";

ssh -t ${NameNode\_2} "hadoop-daemon.sh stop zkfc";

ssh -t ${NameNode\_1} "hadoop-daemon.sh stop zkfc";

;;

\* ) echo -e "Usage: hadoop-manager.sh {start|stop} ^\_^\n" && exit;

;;

esac

end\_time=`date +%s`

elapse\_time=$((${end\_time}-${start\_time}))

echo -e "$1 Hadoop Server takes ${elapse\_time} seconds\n"

4）Spark 管理脚本

文件名：spark-manager.sh

功能：启动、关闭 Spark 集群

内容:

#!/bin/bash

spark\_01=bigdata-master-01

spark\_02=bigdata-master-02

spark\_03=bigdata-slave-01

spark\_04=bigdata-slave-02

spark\_05=bigdata-slave-03

start\_time=`date +%s`

case $1 in

start) ssh -t $spark\_01 "source ~/.bash\_profile;/home/hadoop/bigdata/workspace/spark-1.6.3-bin-hadoop2.6/sbin/start-all.sh" 1>/dev/null;

ssh -t $spark\_02 "source ~/.bash\_profile;/home/hadoop/bigdata/workspace/spark-1.6.3-bin-hadoop2.6/sbin/start-all.sh" 1>/dev/null;

ssh -t $spark\_03 "source ~/.bash\_profile;/home/hadoop/bigdata/workspace/spark-1.6.3-bin-hadoop2.6/sbin/start-all.sh" 1>/dev/null;

ssh -t $spark\_04 "source ~/.bash\_profile;/home/hadoop/bigdata/workspace/spark-1.6.3-bin-hadoop2.6/sbin/start-all.sh" 1>/dev/null;

ssh -t $spark\_05 "source ~/.bash\_profile;/home/hadoop/bigdata/workspace/spark-1.6.3-bin-hadoop2.6/sbin/start-all.sh" 1>/dev/null;

ssh -t $spark\_01 "source ~/.bash\_profile;/home/hadoop/bigdata/workspace/spark-1.6.3-bin-hadoop2.6/sbin/start-history-server.sh" 1>/dev/null;

;;

stop) ssh -t $spark\_01 "source ~/.bash\_profile;/home/hadoop/bigdata/workspace/spark-1.6.3-bin-hadoop2.6/sbin/stop-history-server.sh" 1>/dev/null;

ssh -t $spark\_05 "source ~/.bash\_profile;/home/hadoop/bigdata/workspace/spark-1.6.3-bin-hadoop2.6/sbin/stop-all.sh" 1>/dev/null;

ssh -t $spark\_04 "source ~/.bash\_profile;/home/hadoop/bigdata/workspace/spark-1.6.3-bin-hadoop2.6/sbin/stop-all.sh" 1>/dev/null;

ssh -t $spark\_03 "source ~/.bash\_profile;/home/hadoop/bigdata/workspace/spark-1.6.3-bin-hadoop2.6/sbin/stop-all.sh" 1>/dev/null;

ssh -t $spark\_03 "source ~/.bash\_profile;/home/hadoop/bigdata/workspace/spark-1.6.3-bin-hadoop2.6/sbin/stop-all.sh" 1>/dev/null;

ssh -t $spark\_01 "source ~/.bash\_profile;/home/hadoop/bigdata/workspace/spark-1.6.3-bin-hadoop2.6/sbin/stop-all.sh" 1>/dev/null;

;;

\*) echo -e "Usage: hadoop-manager.sh {start|stop} ^\_^\n" && exit;

;;

esac

end\_time=`date +%s`

elapse\_time=$((${end\_time}-${start\_time}))

echo -e "$1 Spark Server takes ${elapse\_time} seconds\n"

5)整个集群管理脚本

文件名：Bigdata-Cluster.sh

功能：统一启动、关闭及查看 ZooKeeper+Hadoop+Spark 大集群

内容:

#!/bin/bash

CLUSTER\_CONF\_PATH=$(cd "$(dirname "$0")"; pwd)

NameNode\_1=bigdata-master-01

NameNode\_2=bigdata-master-02

SLAVE\_1=bigdata-slave-01

SLAVE\_2=bigdata-slave-02

SLAVE\_3=bigdata-slave-03

start\_time=`date +%s`

function showJps() {

echo -e "\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"

echo -e "current the process on ${NameNode\_1} is:" && ssh -t ${NameNode\_1} source ~/.bash\_profile; jps

echo -e "\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"

echo -e "current the process on ${NameNode\_2} is:" && ssh -t ${NameNode\_2} source ~/.bash\_profile; jps

echo -e "\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"

echo -e "current the process on ${SLAVE\_1} is:" && ssh -t ${SLAVE\_1} source ~/.bash\_profile; jps

echo -e "\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"

echo -e "current the process on ${SLAVE\_2} is:" && ssh -t ${SLAVE\_2} source ~/.bash\_profile; jps

echo -e "\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"

echo -e "current the process on ${SLAVE\_3} is:" && ssh -t ${SLAVE\_3} source ~/.bash\_profile; jps

echo -e "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"

}

case $1 in

start) sh $CLUSTER\_CONF\_PATH/zk-manager.sh start ;

sh $CLUSTER\_CONF\_PATH/journal-manager.sh start;

sh $CLUSTER\_CONF\_PATH/hadoop-manager.sh start;

#sh $CLUSTER\_CONF\_PATH/hbase-manager.sh start;

sh $CLUSTER\_CONF\_PATH/spark-manager.sh start;

showJps

;;

stop) sh $CLUSTER\_CONF\_PATH/spark-manager.sh stop;

#sh $CLUSTER\_CONF\_PATH/hbase-manager.sh stop;

sh $CLUSTER\_CONF\_PATH/spark-manager.sh stop;

sh $CLUSTER\_CONF\_PATH/hadoop-manager.sh stop;

sh $CLUSTER\_CONF\_PATH/journal-manager.sh stop;

sh $CLUSTER\_CONF\_PATH/zk-manager.sh stop;

showJps

;;

status) showJps

;;

\*) echo -e "Usage: sh bigdata-cluster.sh {start|stop|status} ^\_^\n" ;;

esac

end\_time=`date +%s`

elapse\_time=$((${end\_time}-${start\_time}))

echo -e "\n$1 Bigdata Cluster takes ${elapse\_time} seconds\n"

1. 集群访问说明：

ResoureManager:192.168.2.11:50070

ClusterManager:192.168.2.11:8188

Spark UI:192.168.2.11:8080

Spark History Server:192.168.2.11:18080