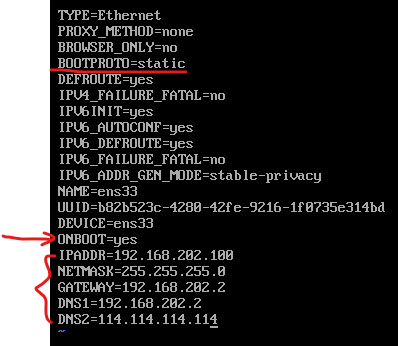
# 安装配置centos7虚拟机

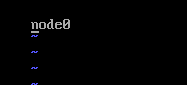
## 网络设置

vi /etc/sysconfig/network-scripts/ifcfg-ens33



## 配置计算机名称

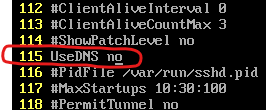
vi /etc/hostname



reboot重启虚拟机

## 配置ssh，禁用DNS

vi /etc/ssh/sshd\_config



重启sshd服务：service sshd restart

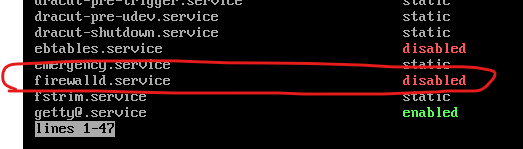
或者：systemctl restart sshd

## 禁用防火墙

查看防火墙状态：systemctl list-unit-files



禁用防火墙：systemctl disable firewalld



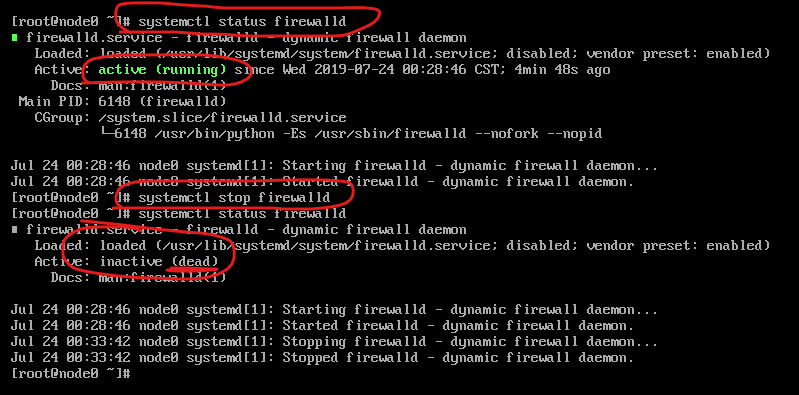
查看firewalld服务的运行状态并关闭防火墙服务：

查看防火墙服务状态：systemctl status firewalld

下图中防火墙开启

关闭防火墙服务：systemctl stop firewalld

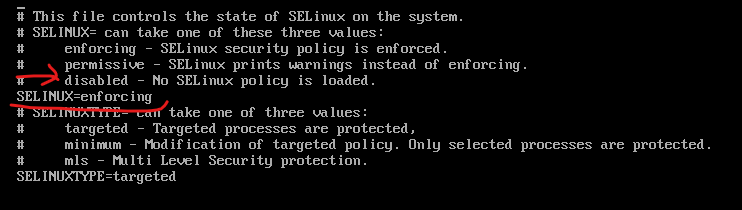
再次查看发现已经关闭



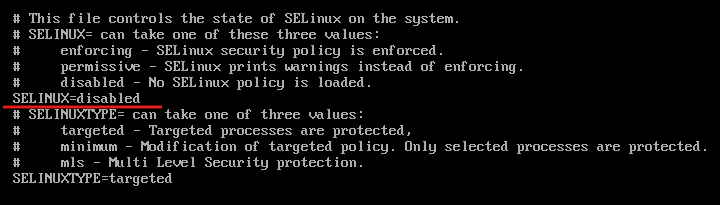
## 禁用selinux

getenforce命令查看selinux状态

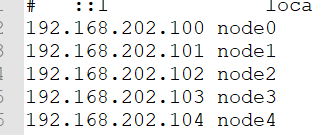
vi /etc/selinux/config



关闭selinux服务：需要重启虚拟机（reboot重启）

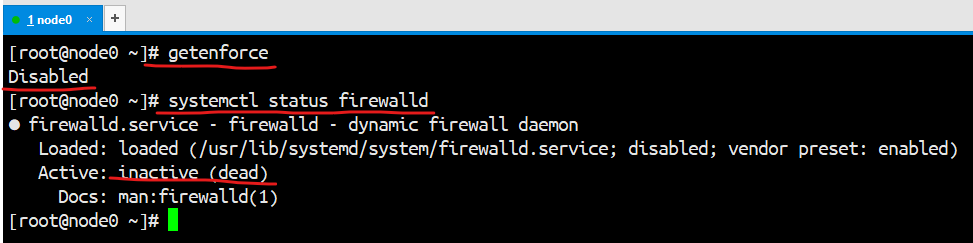


在windows的C:\Windows\System32\drivers\etc\hosts文件中追加：



用于解析node0|1|2|3|4的访问。

重新启动用xshell连接之后查看selinux和防火墙状态：

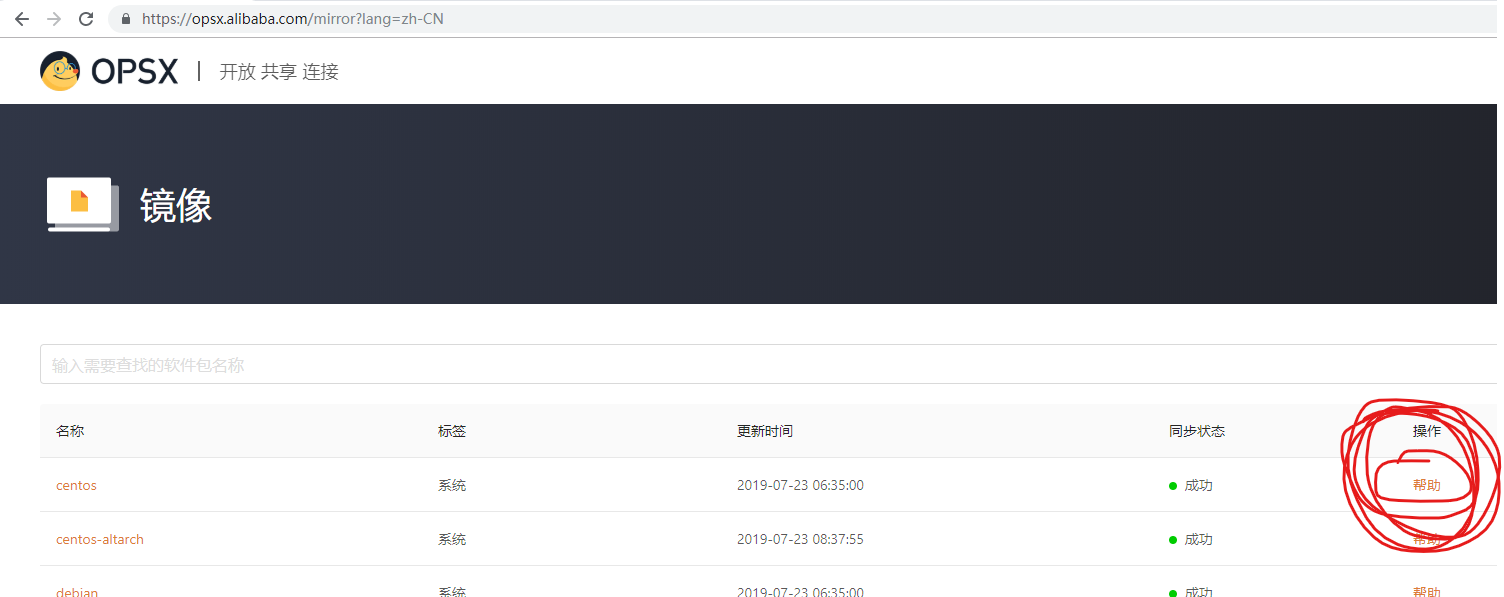


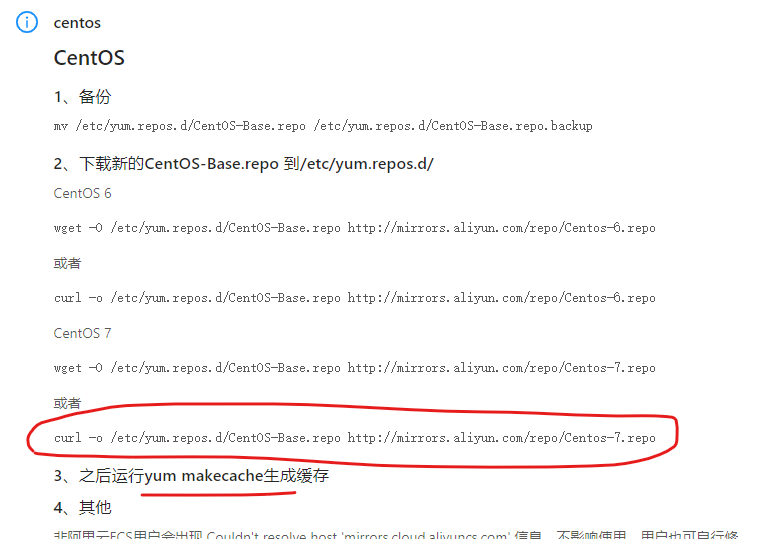
## 设置yum源

将yum源设置为alibaba的镜像：

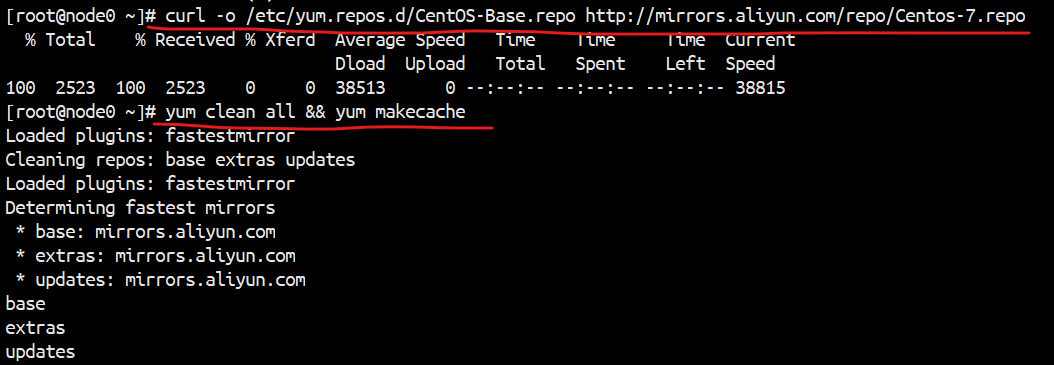
访问：<https://opsx.alibaba.com/mirror?lang=zh-CN>

点击centos的帮助

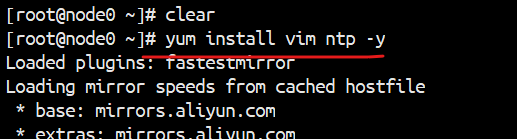




在xshell中执行操作：



## 安装vim和ntp



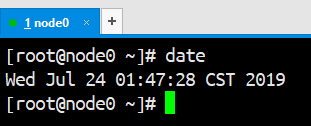
## 设置时间同步

查看服务状态：systemctl list-unit-files

发现ntpd.service没有开启自启

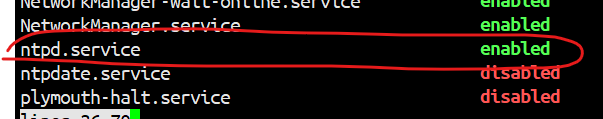


查看时间：



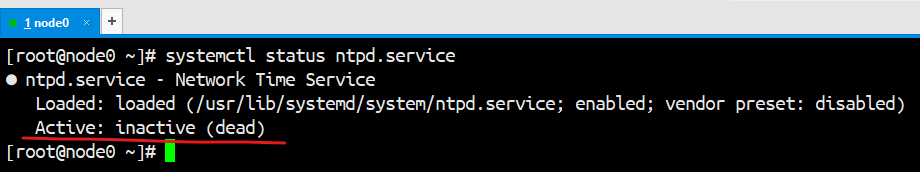
设置开机自启：

systemctl enable ntpd.service

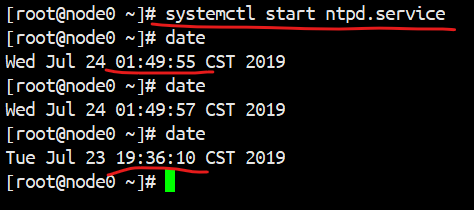


查看ntpd.service是否启动了：

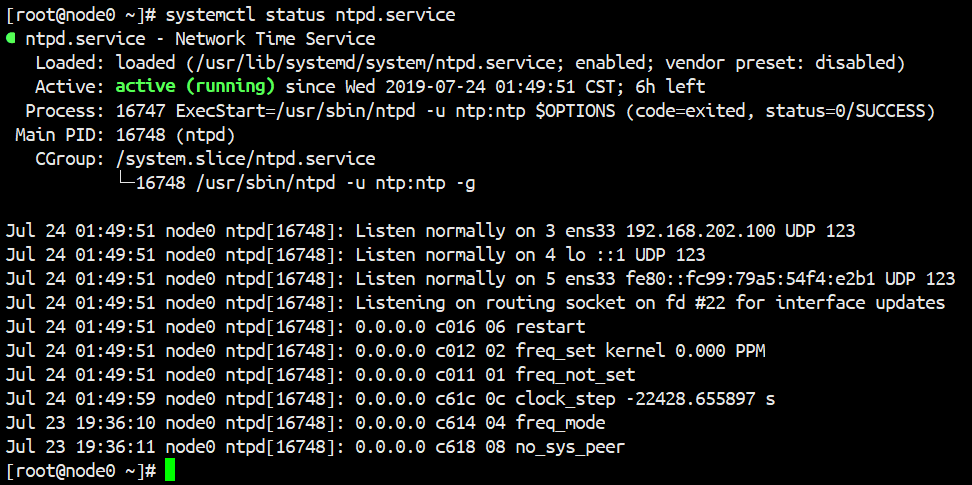
systemctl status ntpd.service



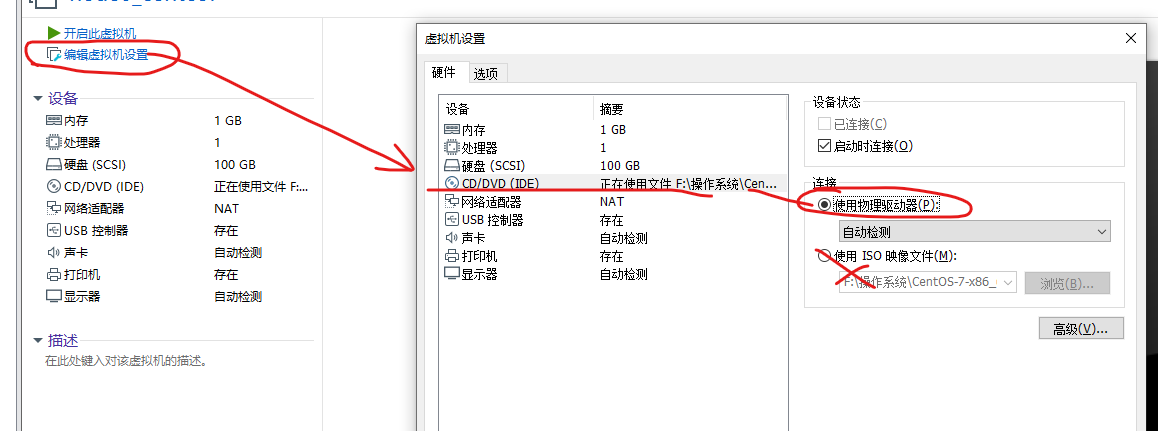
启动该服务：



可以看到时间已经和互联网同步了。



## 关闭虚拟机，拍快照





# 克隆虚拟机

四台虚拟机

1、分别设置四台虚拟机的/etc/hostname

node1

node2

node3

node4

以及/etc/sysconfig/network-scripts/ifcfg-ens33

IPADDR=192.168.202.101

IPADDR=192.168.202.102

IPADDR=192.168.202.103

IPADDR=192.168.202.104

2、分别设置4台虚拟机的/etc/hosts文件，添加

192.168.202.101 node1

192.168.202.102 node2

192.168.202.103 node3

192.168.202.104 node4

3、关闭四台虚拟机，拍摄快照

poweroff

# 部署hadoop集群

hadoop集群是namenode-ha以及resourcemanager-ha的

## 配置四台虚拟机免密钥登录

4个都执行：ssh-keygen -t dsa -P '' -f ~/.ssh/id\_dsa

node1:

cat ~/.ssh/id\_dsa.pub >> ~/.ssh/authorized\_keys

scp ~/.ssh/authorized\_keys node2:~/.ssh/

node2:

cat ~/.ssh/id\_dsa.pub >> ~/.ssh/authorized\_keys

scp ~/.ssh/authorized\_keys node3:~/.ssh/

node3:

cat ~/.ssh/id\_dsa.pub >> ~/.ssh/authorized\_keys

scp ~/.ssh/authorized\_keys node4:~/.ssh/

node4:

cat ~/.ssh/id\_dsa.pub >> ~/.ssh/authorized\_keys

scp ~/.ssh/authorized\_keys node1:~/.ssh/

scp ~/.ssh/authorized\_keys node2:~/.ssh/

scp ~/.ssh/authorized\_keys node3:~/.ssh/

ssh node1

exit

ssh node2

exit

ssh node3

exit

ssh node4

exit

ssh localhost

exit

ssh 127.0.0.1

exit

node1:

ssh node1

exit

ssh node2

exit

ssh node3

exit

ssh node4

exit

ssh localhost

exit

ssh 127.0.0.1

exit

node2:

ssh node1

exit

ssh node2

exit

ssh node3

exit

ssh node4

exit

ssh localhost

exit

ssh 127.0.0.1

exit

node3:

ssh node1

exit

ssh node2

exit

ssh node3

exit

ssh node4

exit

ssh localhost

exit

ssh 127.0.0.1

exit

## 上传包

hadoop-2.6.5.tar.gz、jdk-1.7.0\_80.rpm、zookeeper-3.4.6.tar.gz

上传到node1上：

hadoop-2.6.5.tar.gz jdk-8u191-linux-x64.rpm zookeeper-3.4.6.tar.gz

将它们拷贝到其他三台虚拟机上：

scp hadoop-2.6.5.tar.gz jdk-8u191-linux-x64.rpm zookeeper-3.4.6.tar.gz node2:~

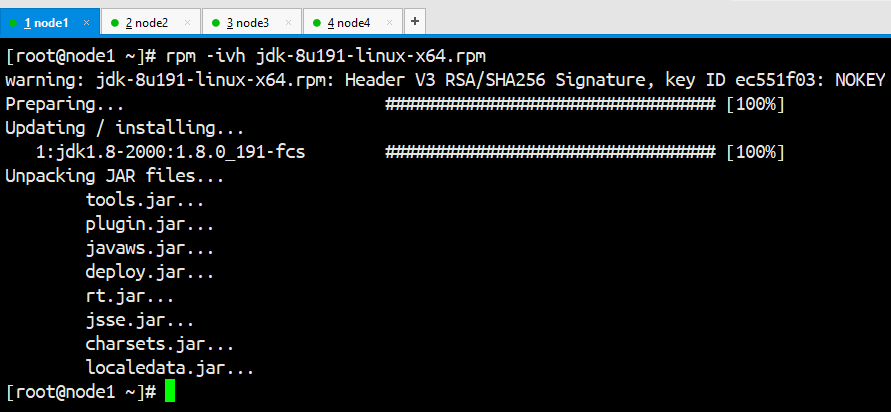
scp hadoop-2.6.5.tar.gz jdk-8u191-linux-x64.rpm zookeeper-3.4.6.tar.gz node3:~

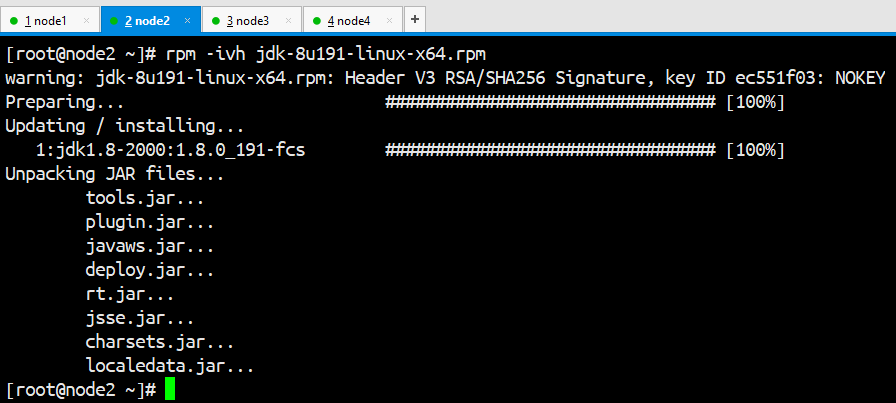
scp hadoop-2.6.5.tar.gz jdk-8u191-linux-x64.rpm zookeeper-3.4.6.tar.gz node4:~

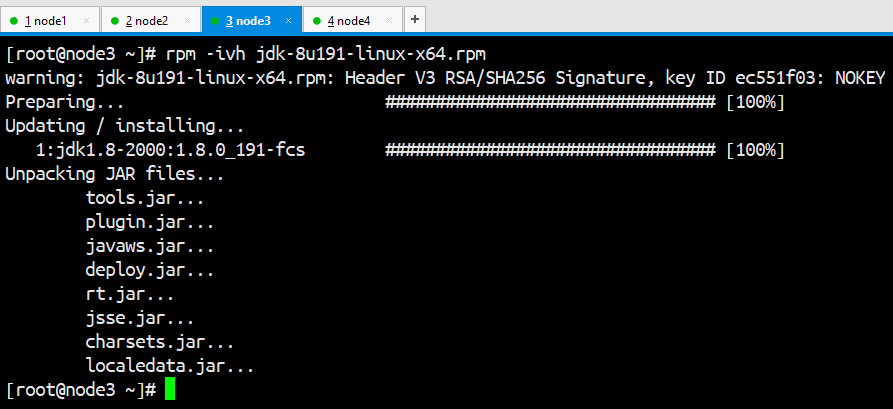
## 安装jdk并配置环境变量

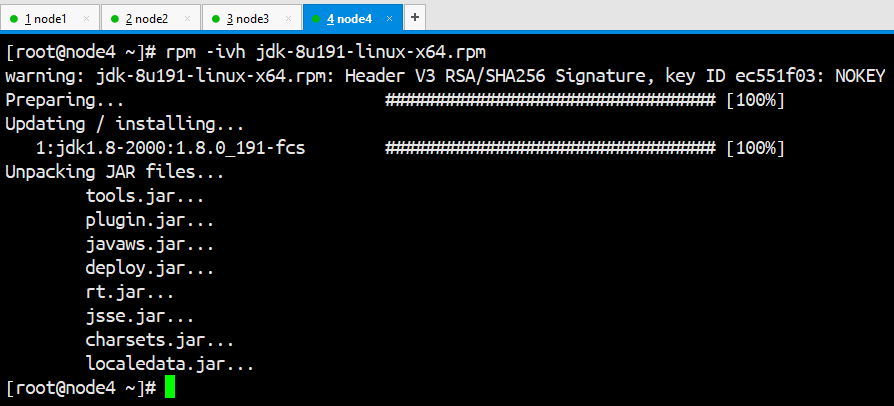
四台都执行：

rpm -ivh jdk-8u191-linux-x64.rpm



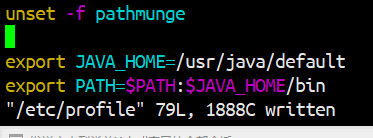






在node1上配置java的环境变量：

vim /etc/profile



将/etc/profile文件拷贝到其他三台服务器

scp /etc/profile node2:/etc/

scp /etc/profile node3:/etc/

scp /etc/profile node4:/etc/

在四台服务器让配置生效：

. /etc/profile或者source /etc/profile

然后在四台虚拟机上运行jps看是否有输出，如果有输出则表示配置成功。

## 在node2、node3、node4上安装和配置zookeeper

在node2、node3、node4上执行解压，解压到/opt目录

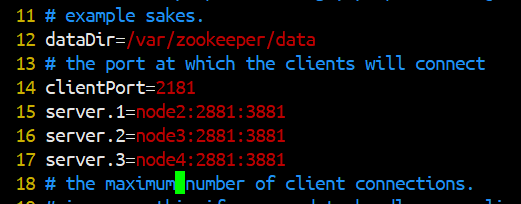
tar -zxf zookeeper-3.4.6.tar.gz -C /opt

在node2上配置zookeeper

cd /opt/zookeeper-3.4.6/conf

cp zoo\_sample.cfg zoo.cfg

vim zoo.cfg



mkdir -p /var/zookeeper/data

echo 1 > /var/zookeeper/data/myid

scp zoo.cfg node3:`pwd`

scp zoo.cfg node4:`pwd`

node3上执行：

mkdir -p /var/zookeeper/data

echo 2 > /var/zookeeper/data/myid

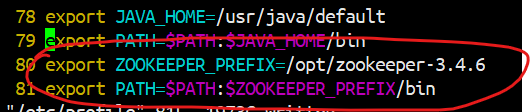
node4上执行：

mkdir -p /var/zookeeper/data

echo 3 > /var/zookeeper/data/myid

在node2、node3和node4上设置zookeeper的环境变量：

vim /etc/profile



保存退出之后让配置生效

. /etc/profile

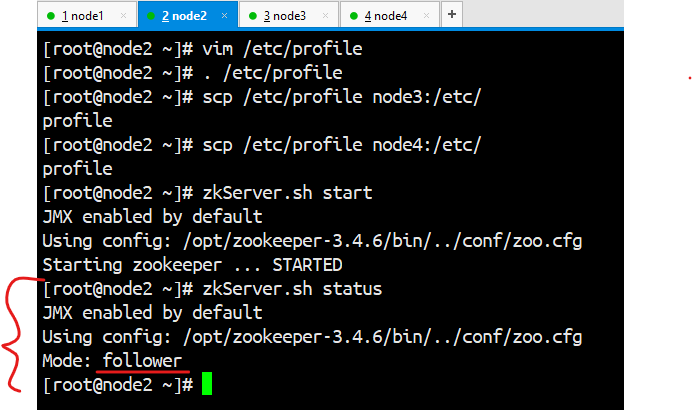
然后将/etc/profile文件拷贝到node3和node4：

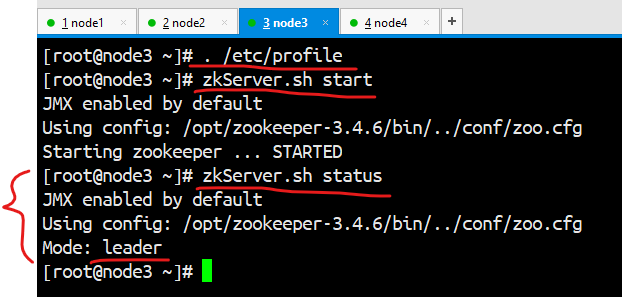
scp /etc/profile node3:/etc/

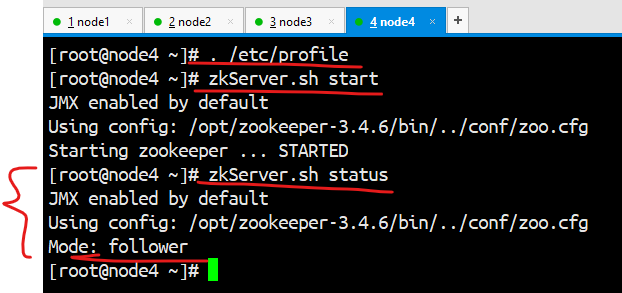
scp /etc/profile node4:/etc/

并在node3和node4执行. /etc/profile让它生效。

启动node2、node3和node4上的zookeeper



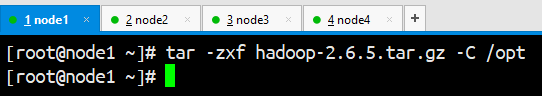


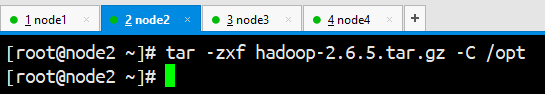


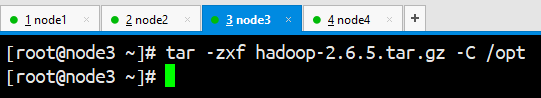
在三台服务器都启动zookeeper之后查看zookeeper的状态，node2是follower，node3是leader，node4是follower。

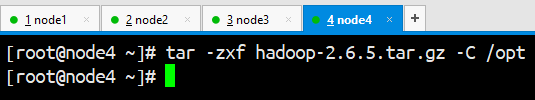
## 解压并配置hadoop

1、在4台服务器上解压hadoop并配置



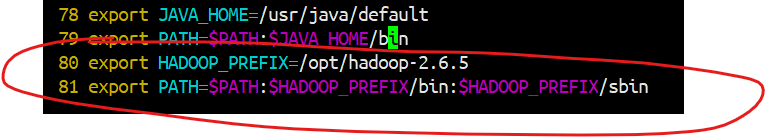






在node1上配置hadoop的环境变量：

vim /etc/profile

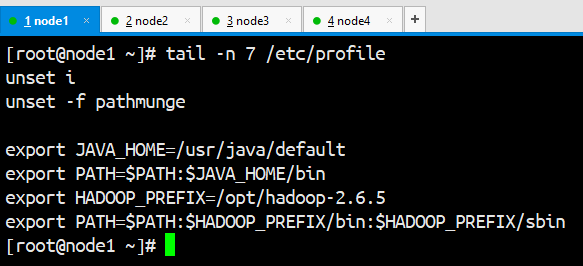


将第80和81行拷贝并追加到node2、node3和node4的/etc/profile文件中

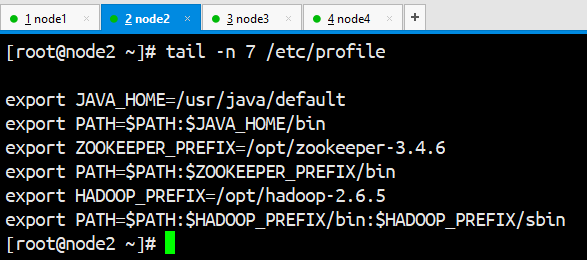
不能将/etc/profile文件拷贝到node2，node3和node4，会覆盖zookeeper的环境变量配置

并在四台虚拟机上执行. /etc/profile让配置生效。

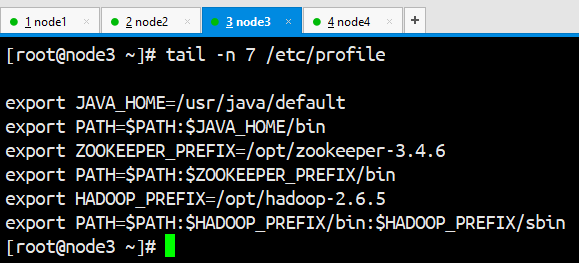
node1的profile文件：



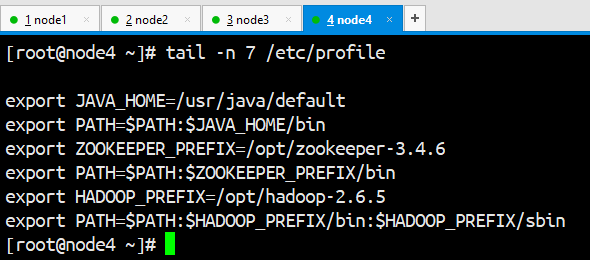
node2的profile文件：



node3的profile文件：



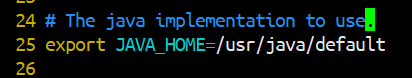
node4的profile文件：



### hadoop-env.sh

cd /opt/hadoop-2.6.5/etc/hadoop

vi hadoop-env.sh，设置JAVA\_HOME：



### core-site.xml:

|  |
| --- |
| <configuration>  <property>  <name>fs.defaultFS</name>  <value>hdfs://mycluster</value>  </property>  <property>  <name>hadoop.tmp.dir</name>  <value>/var/hadoop/ha</value>  </property>  <!-- 指定每个zookeeper服务器的位置和客户端端口号 -->  <property>  <name>ha.zookeeper.quorum</name>  <value>node2:2181,node3:2181,node4:2181</value>  </property>  </configuration> |

### hdfs-site.xml:

|  |
| --- |
| <configuration>  <!-- 指定block默认副本个数 -->  <property>  <name>dfs.replication</name>  <value>2</value>  </property>  <!-- 用于解析fs.defaultFS中hdfs://mycluster中的mycluster地址 -->  <property>  <name>dfs.nameservices</name>  <value>mycluster</value>  </property>  <!-- mycluster下面由两个namenode服务支撑 -->  <property>  <name>dfs.ha.namenodes.mycluster</name>  <value>nn1,nn2</value>  </property>  <!--指定nn1的地址和端口号，发布的是一个hdfs://的服务-->  <property>  <name>dfs.namenode.rpc-address.mycluster.nn1</name>  <value>node1:8020</value>  </property>  <property>  <name>dfs.namenode.rpc-address.mycluster.nn2</name>  <value>node2:8020</value>  </property>  <!--指定三台journal服务器的地址-->  <property>  <name>dfs.namenode.shared.edits.dir</name>  <value>qjournal://node1:8485;node2:8485;node3:8485/mycluster</value>  </property>  <!-- 指定客户端查找active的namenode的策略：  会给所有namenode发请求，以决定哪个是active的 -->  <property>  <name>dfs.client.failover.proxy.provider.mycluster</name>  <value>org.apache.hadoop.hdfs.server.namenode.ha.ConfiguredFailoverProxyProvider</value>  </property>  <!--在发生故障切换的时候，ssh到对方服务器，将namenode进程kill掉 kill -9 55767-->  <property>  <name>dfs.ha.fencing.methods</name>  <value>sshfence</value>  </property>  <property>  <name>dfs.ha.fencing.ssh.private-key-files</name>  <value>/root/.ssh/id\_dsa</value>  </property>  <!-- 指定journalnode在哪个目录存放edits log文件 -->  <property>  <name>dfs.journalnode.edits.dir</name>  <value>/var/hadoop/ha/jnn</value>  </property>  <!--启用自动故障切换-->  <property>  <name>dfs.ha.automatic-failover.enabled</name>  <value>true</value>  </property>  </configuration> |

### slaves:

|  |
| --- |
| node2  node3  node4 |

### mapred-site.xml:

|  |
| --- |
| <configuration>  <property>  <name>mapreduce.framework.name</name>  <value>yarn</value>  </property>  </configuration> |

### yarn-site.xml:

|  |
| --- |
| <configuration>  <!-- Site specific YARN configuration properties -->  <property>  <name>yarn.nodemanager.aux-services</name>  <value>mapreduce\_shuffle</value>  </property>  <!--启用ResourceManager的高可用-->  <property>  <name>yarn.resourcemanager.ha.enabled</name>  <value>true</value>  </property>  <!--指代ResourceManager HA的两台RM的逻辑名称 -->  <property>  <name>yarn.resourcemanager.cluster-id</name>  <value>rmhacluster1</value>  </property>  <!--指定该高可用ResourceManager下的两台ResourceManager的逻辑名称-->  <property>  <name>yarn.resourcemanager.ha.rm-ids</name>  <value>rm1,rm2</value>  </property>  <!--指定第一台ResourceManager服务器所在的主机名称 -->  <property>  <name>yarn.resourcemanager.hostname.rm1</name>  <value>node3</value>  </property>  <property>  <name>yarn.resourcemanager.hostname.rm2</name>  <value>node4</value>  </property>  <!--指定resourcemanager的web服务器的主机名和端口号-->  <property>  <name>yarn.resourcemanager.webapp.address.rm1</name>  <value>node3:8088</value>  </property>  <property>  <name>yarn.resourcemanager.webapp.address.rm2</name>  <value>node4:8088</value>  </property>  <!--做ResourceManager HA故障切换用到的zookeeper集群地址 -->  <property>  <name>yarn.resourcemanager.zk-address</name>  <value>node2:2181,node3:2181,node4:2181</value>  </property>  </configuration> |

### 配置文件拷贝

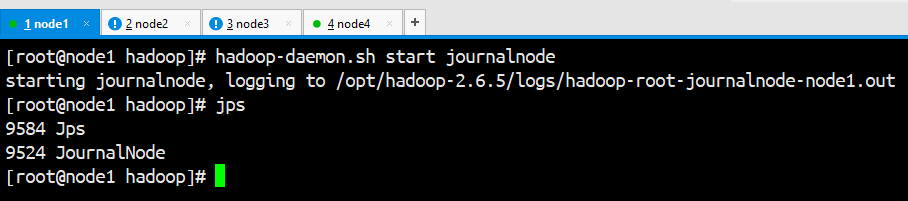
将/opt/hadoop-2.6.5/etc/hadoop/下的所有配置文件拷贝到node2、node3、node4上：

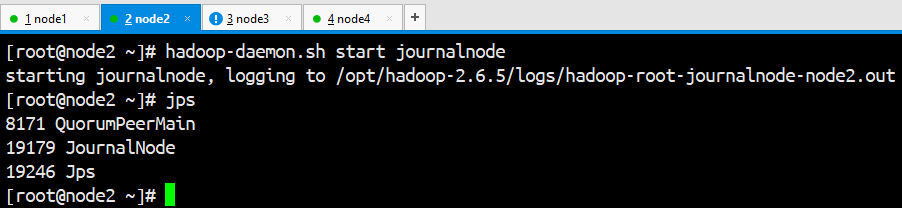
scp -r \* node2:`pwd` && scp -r \* node3:`pwd` && scp -r \* node4:`pwd`

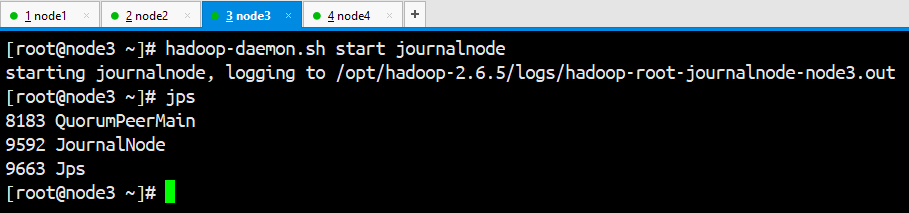
### 启动journalnode

在node1、node2和node3上启动journalnode

hadoop-daemon.sh start journalnode







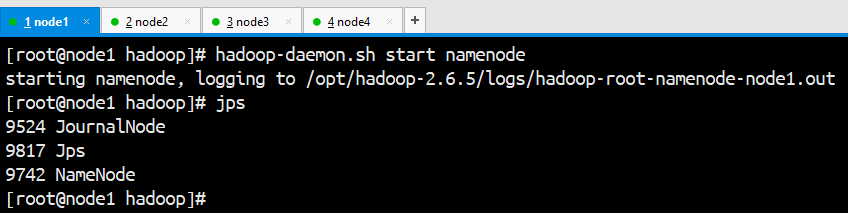
### 格式化HDFS

在node1上格式化hdfs：

hdfs namenode -format

启动namenode进程：

hadoop-daemon.sh start namenode



在node2上同步格式化信息：

hdfs namenode -bootstrapStandby

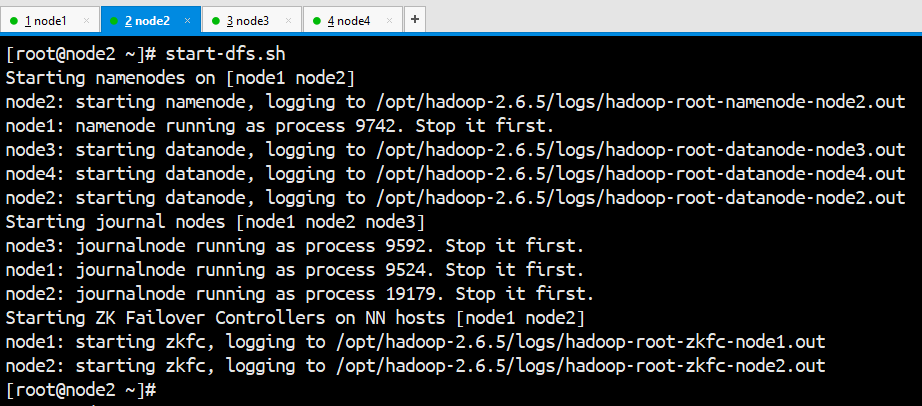
格式化zookeeper：

hdfs zkfc -formatZK

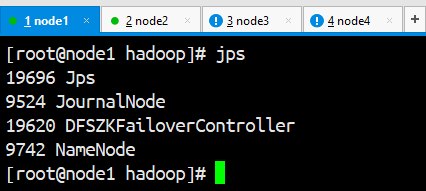
### 启动hadoop集群

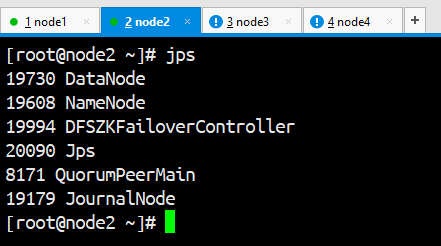
在node1或者node2或者node3或者node4上任意一台，运行start-dfs.sh启动集群

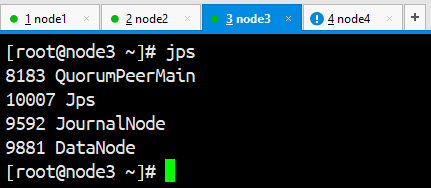
start-dfs.sh

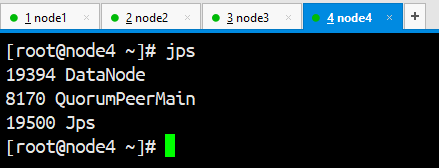


查看各台服务器的进程：



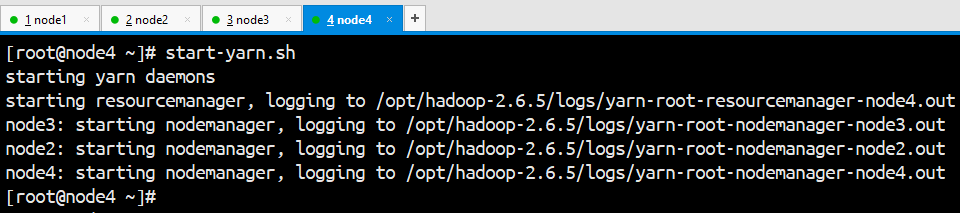






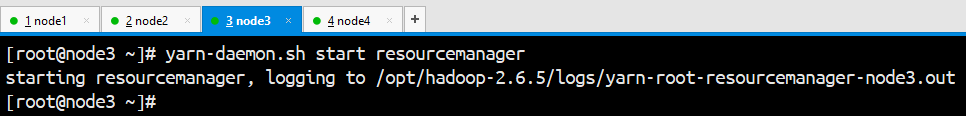
在node3或node4上启动yarn：

start-yarn.sh

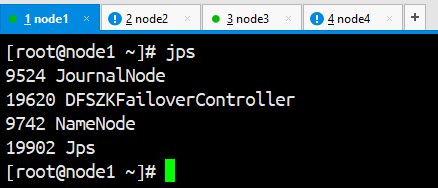


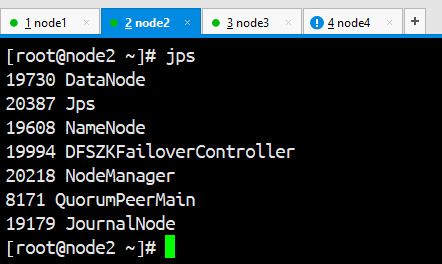
在node4或node3上启动resourcemanager：

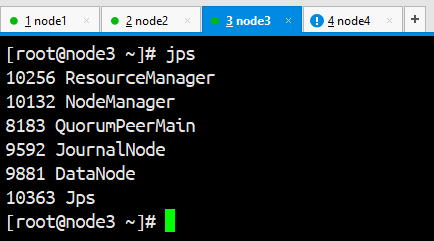
yarn-daemon.sh start resourcemanager

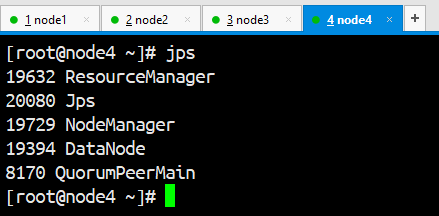


查看各台服务器的进程：





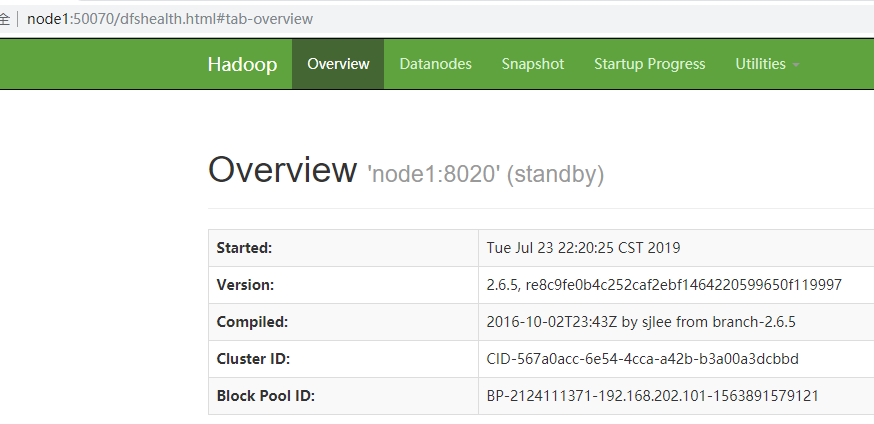




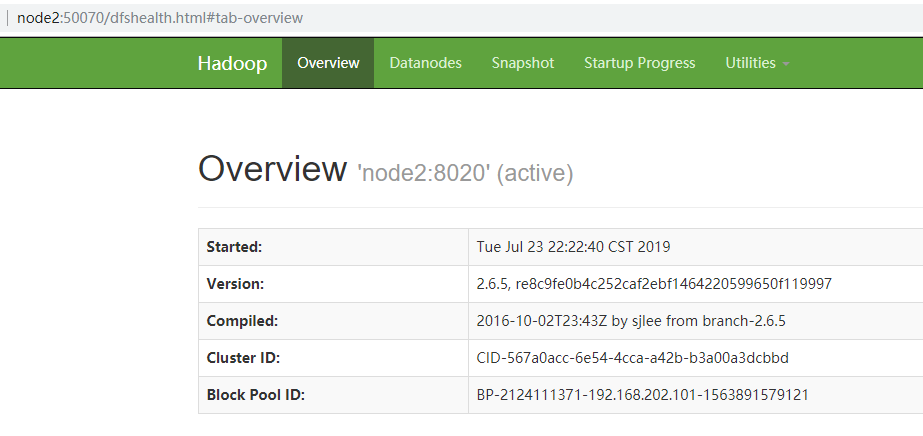
### 在webUI访问

查看namenode状态：

<http://node1:50070>

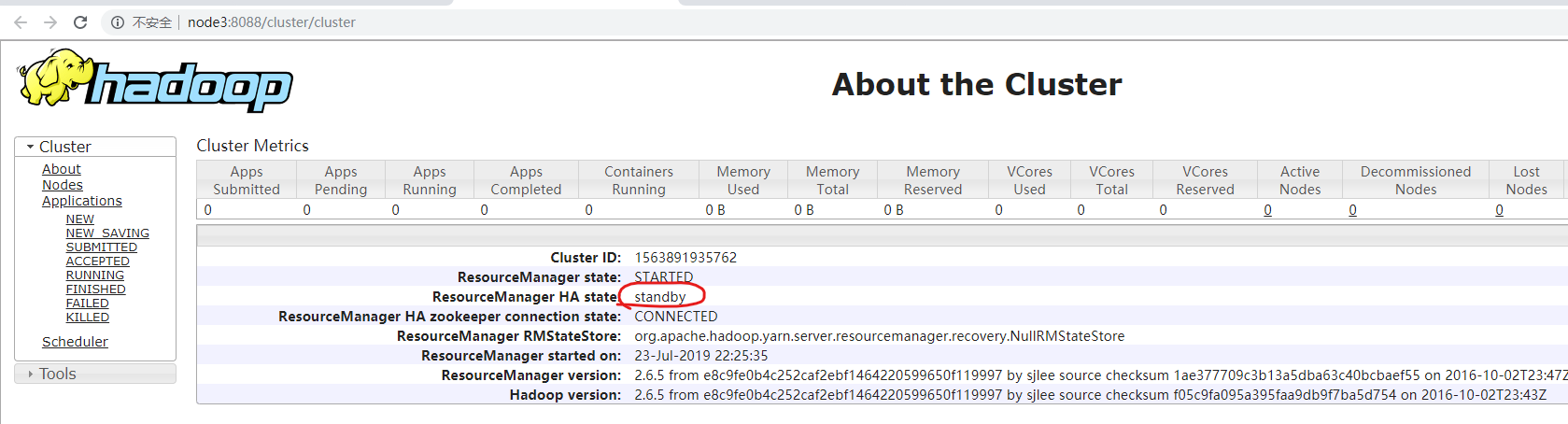


<http://node2:50070>

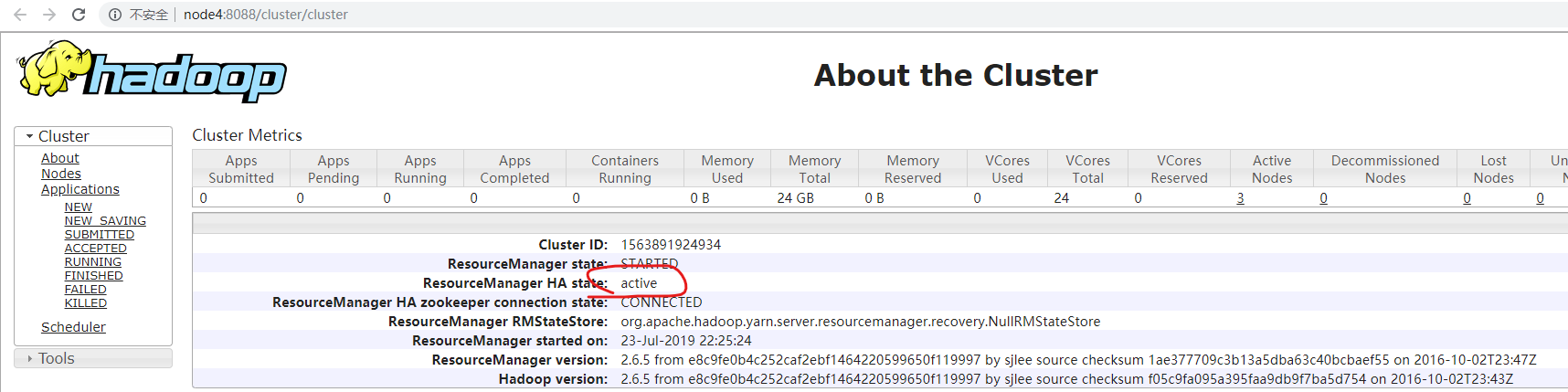


查看resourcemanager状态：

<http://node3:8088/cluster/cluster>

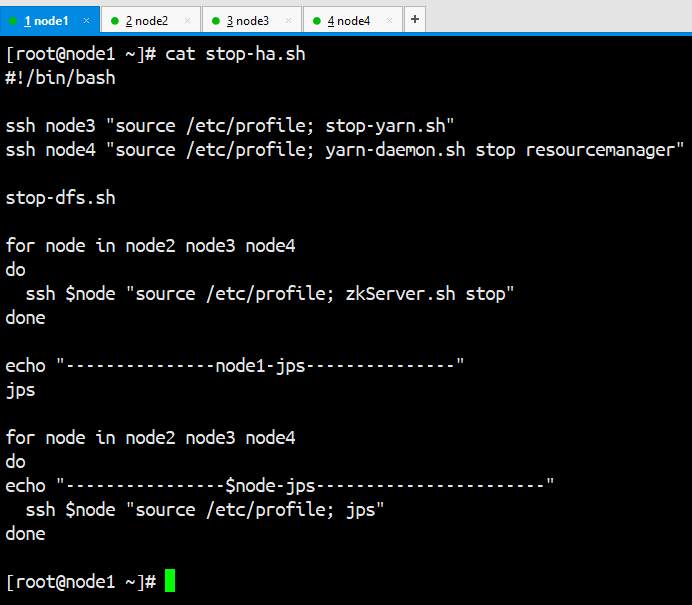


<http://node4:8088/cluster/cluster>



### 停止集群和启动集群脚本

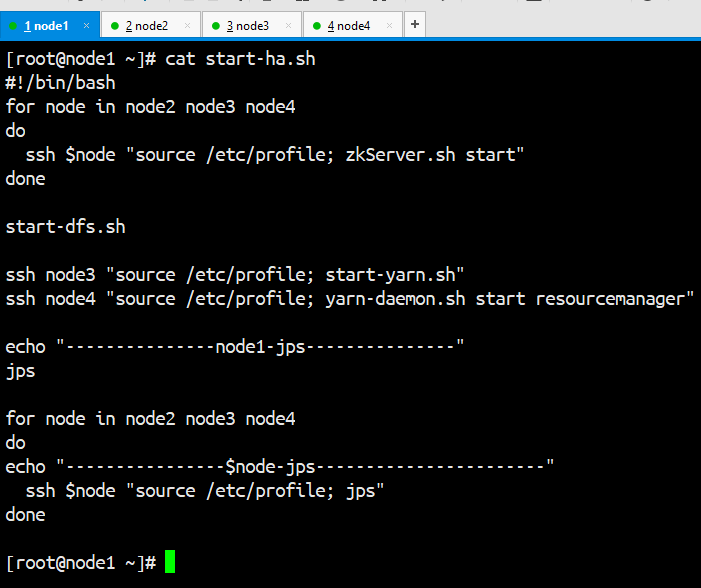
停止脚本：



运行效果：

|  |
| --- |
| [root@node1 ~]# bash stop-ha.sh  stopping yarn daemons  stopping resourcemanager  node2: stopping nodemanager  node4: stopping nodemanager  node3: stopping nodemanager  no proxyserver to stop  stopping resourcemanager  Stopping namenodes on [node1 node2]  node2: stopping namenode  node1: stopping namenode  node4: stopping datanode  node3: stopping datanode  node2: stopping datanode  Stopping journal nodes [node1 node2 node3]  node3: stopping journalnode  node2: stopping journalnode  node1: stopping journalnode  Stopping ZK Failover Controllers on NN hosts [node1 node2]  node2: stopping zkfc  node1: stopping zkfc  JMX enabled by default  Using config: /opt/zookeeper-3.4.6/bin/../conf/zoo.cfg  Stopping zookeeper ... STOPPED  JMX enabled by default  Using config: /opt/zookeeper-3.4.6/bin/../conf/zoo.cfg  Stopping zookeeper ... STOPPED  JMX enabled by default  Using config: /opt/zookeeper-3.4.6/bin/../conf/zoo.cfg  Stopping zookeeper ... STOPPED  ---------------node1-jps---------------  20761 Jps  ----------------node2-jps-----------------------  21021 Jps  ----------------node3-jps-----------------------  19300 Jps  ----------------node4-jps-----------------------  20637 Jps  [root@node1 ~]# |

启动脚本：



运行效果：

|  |
| --- |
| [root@node1 ~]# bash start-ha.sh  JMX enabled by default  Using config: /opt/zookeeper-3.4.6/bin/../conf/zoo.cfg  Starting zookeeper ... STARTED  JMX enabled by default  Using config: /opt/zookeeper-3.4.6/bin/../conf/zoo.cfg  Starting zookeeper ... STARTED  JMX enabled by default  Using config: /opt/zookeeper-3.4.6/bin/../conf/zoo.cfg  Starting zookeeper ... STARTED  Starting namenodes on [node1 node2]  node2: starting namenode, logging to /opt/hadoop-2.6.5/logs/hadoop-root-namenode-node2.out  node1: starting namenode, logging to /opt/hadoop-2.6.5/logs/hadoop-root-namenode-node1.out  node4: starting datanode, logging to /opt/hadoop-2.6.5/logs/hadoop-root-datanode-node4.out  node2: starting datanode, logging to /opt/hadoop-2.6.5/logs/hadoop-root-datanode-node2.out  node3: starting datanode, logging to /opt/hadoop-2.6.5/logs/hadoop-root-datanode-node3.out  Starting journal nodes [node1 node2 node3]  node2: starting journalnode, logging to /opt/hadoop-2.6.5/logs/hadoop-root-journalnode-node2.out  node3: starting journalnode, logging to /opt/hadoop-2.6.5/logs/hadoop-root-journalnode-node3.out  node1: starting journalnode, logging to /opt/hadoop-2.6.5/logs/hadoop-root-journalnode-node1.out  Starting ZK Failover Controllers on NN hosts [node1 node2]  node2: starting zkfc, logging to /opt/hadoop-2.6.5/logs/hadoop-root-zkfc-node2.out  node1: starting zkfc, logging to /opt/hadoop-2.6.5/logs/hadoop-root-zkfc-node1.out  starting yarn daemons  starting resourcemanager, logging to /opt/hadoop-2.6.5/logs/yarn-root-resourcemanager-node3.out  node4: starting nodemanager, logging to /opt/hadoop-2.6.5/logs/yarn-root-nodemanager-node4.out  node2: starting nodemanager, logging to /opt/hadoop-2.6.5/logs/yarn-root-nodemanager-node2.out  node3: starting nodemanager, logging to /opt/hadoop-2.6.5/logs/yarn-root-nodemanager-node3.out  starting resourcemanager, logging to /opt/hadoop-2.6.5/logs/yarn-root-resourcemanager-node4.out  ---------------node1-jps---------------  21313 DFSZKFailoverController  21369 Jps  20954 NameNode  21148 JournalNode  ----------------node2-jps-----------------------  21249 DataNode  21683 Jps  21336 JournalNode  21130 QuorumPeerMain  21453 DFSZKFailoverController  21182 NameNode  21550 NodeManager  ----------------node3-jps-----------------------  20994 ResourceManager  20787 DataNode  21093 NodeManager  20727 QuorumPeerMain  20874 JournalNode  21278 Jps  ----------------node4-jps-----------------------  20801 DataNode  20914 NodeManager  20984 ResourceManager  20747 QuorumPeerMain  21117 Jps  [root@node1 ~]# |

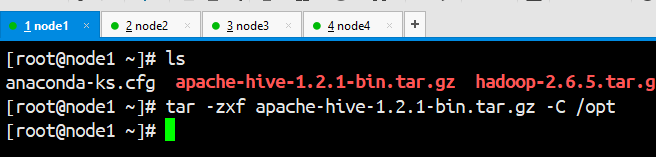
### 日志

哪台服务器上的hadoop哪个进程有问题，就到/opt/hadoop-2.6.5/logs中查找日志查看。

# 使用hive

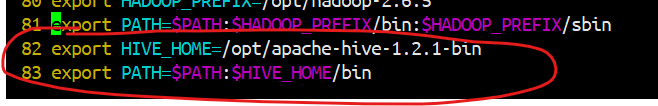
将hive的tar.gz上传到node1

解压到/opt目录



配置hive的环境变量

vim /etc/profile



让配置生效：. /etc/profile

进入/opt/apache-hive-1.2.1-bin/conf

cp hive-default.xml.template hive-site.xml

vim hive-site.xml

|  |
| --- |
| <configuration>  <property>  <name>hive.metastore.warehouse.dir</name>  <value>/user/hive/warehouse</value>  </property>  <property>  <name>javax.jdo.option.ConnectionURL</name>  <value>jdbc:mysql://node4:3306/hive?createDatabaseIfNotExist=true</value>  </property>  <property>  <name>javax.jdo.option.ConnectionDriverName</name>  <value>com.mysql.jdbc.Driver</value>  </property>  <property>  <name>javax.jdo.option.ConnectionUserName</name>  <value>root</value>  </property>  <property>  <name>javax.jdo.option.ConnectionPassword</name>  <value>123456</value>  </property>  </configuration> |

将hadoop/share/hadoop/yarn/lib中的jline-0.9.4.jar替换为hive/lib中的jline-2.2.1.jar

cp /opt/apache-hive-1.2.1-bin/lib/jline-2.12.jar /opt/hadoop-2.6.5/share/hadoop/yarn/lib/

mv /opt/hadoop-2.6.5/share/hadoop/yarn/lib/jline-0.9.94.jar /opt/hadoop-2.6.5/share/hadoop/yarn/lib/jline-0.9.94.jar.bak

将jline-0.9.94.jar.bak和jline-2.12.jar拷贝到其他的所有虚拟机中，并将其他所有虚拟机中的/opt/hadoop-2.6.5/share/hadoop/yarn/lib/jline-0.9.94.jar删除

scp jline-\* node2:`pwd` && scp jline-\* node3:`pwd` && scp jline-\* node4:`pwd`

ssh node2 "cd `pwd`; rm -f jline-0.9.94.jar"

ssh node3 "cd `pwd`; rm -f jline-0.9.94.jar"

ssh node4 "cd `pwd`; rm -f jline-0.9.94.jar"

将mysql的jdbc驱动jar拷贝到hive/lib中。

mysql-connector-java-5.1.39-bin.jar

**安装和配置mysql**

在node4上安装mysql-server

拷贝mysql相关的rpm到node4：

yum install net-tools

rpm -e --nodeps `rpm -qa | grep mariadb`

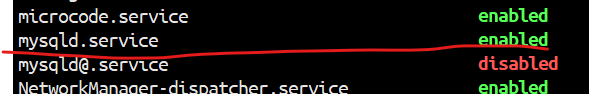
rpm -ivh mysql-community-common-5.7.27-1.el7.x86\_64.rpm

rpm -ivh mysql-community-libs-5.7.27-1.el7.x86\_64.rpm

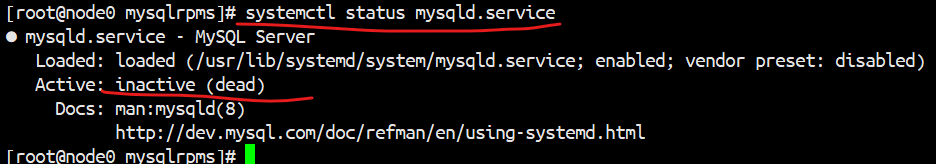
rpm -ivh mysql-community-client-5.7.27-1.el7.x86\_64.rpm

rpm -ivh mysql-community-server-5.7.27-1.el7.x86\_64.rpm

systemctl list-unit-files

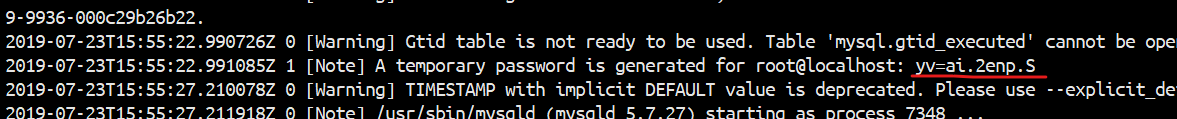


systemctl status mysqld.service

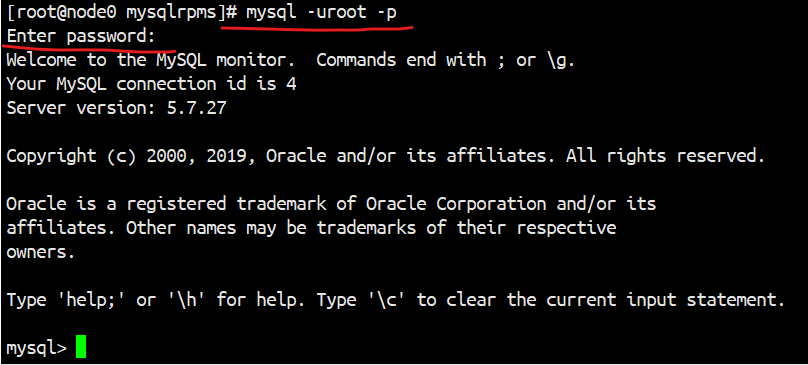


systemctl start mysqld.service

密码在/var/log/mysqld.log中



登录成功



配置远程root登录：

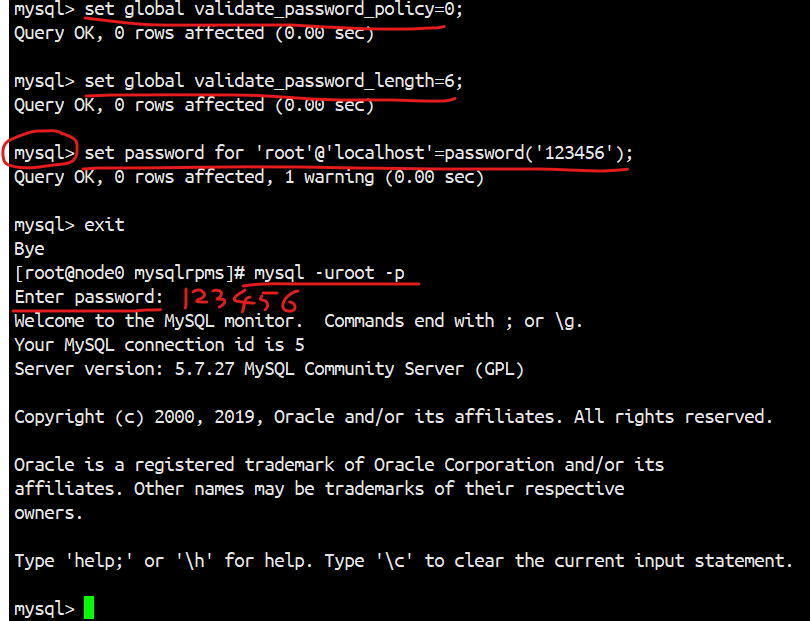
首先，修改root登录的密码为123456：

修改默认密码：

mysql> set global validate\_password\_policy=0;

mysql> set global validate\_password\_length=6;

mysql> set password for 'root'@'localhost'=password('123456');



给root设置远程登录权限

mysql> GRANT ALL PRIVILEGES ON \*.\* TO 'root'@'%' IDENTIFIED BY '123456' WITH GRANT OPTION;

mysql> FLUSH PRIVILEGES;

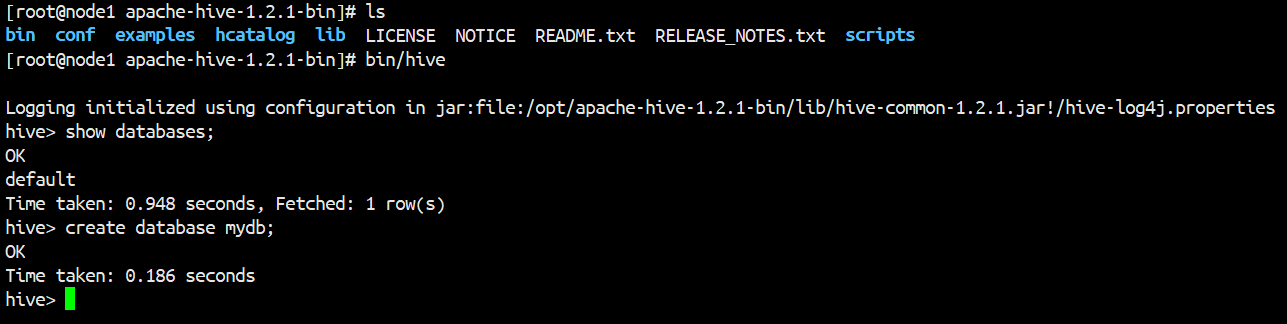
启动hive的metastore：

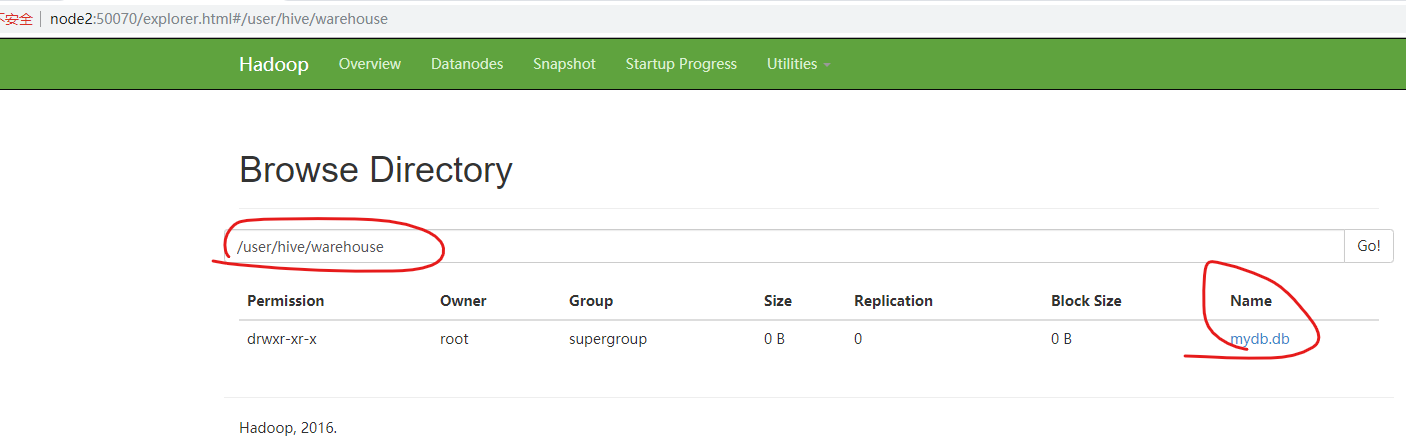


hive客户端配置：

重新解压一份hive，在conf/hive-site.xml中配置如下：

|  |
| --- |
| <configuration>  <property>  <name>hive.metastore.warehouse.dir</name>  <value>/user/hive/warehouse</value>  </property>  <property>  <name>hive.metastore.local</name>  <value>false</value>  </property>  <property>  <name>hive.metastore.uris</name>  <value>thrift://node1:9083</value>  </property>  </configuration> |





hive配置完成。