Hadoop是一个开源的，可靠的，可扩展的分布式计算框架

14.访问浏览器进行验证 (用自己的地址访问)

Hadoop NameNode：http://10.122.x.x:50070/

Secondary namenode: http://10.122.x.x:50090/

ResourceManager: http://10.122.x.x:8088/

DataNode: http://10.122.x.x:50075/

**安装配置hadoop环境**

1. 环境准备

关闭服务其中的防火墙\selinux\配置网络\配置服务器间免密码登录

|  |  |  |  |
| --- | --- | --- | --- |
| 主机 | 主机名 | Ip | 进程规划 |
| Hadoop1 | Hadoop01.zhang.com | 192.168.75.3 | JobHistoryServer  DataNode  NameNode  NodeManager |
| Hadoop2 | Hadoop02.zhang.com | 192.168.75.8 | NodeManager  WebAppProxyServer  ResourceManager  DataNode |
| Hadoop3 | Hadoop03.zhang.com | 192.168.75.9 | NodeManager  SecondaryNameNode  DataNode |

核实环境情况：

[root@muban ~]# iptables -L

Selinux状态：

[root@muban ~]# getenforce

Disabled

修改主机的主机名，以及hosts文件：

[root@muban ~]# cat /etc/sysconfig/network

NETWORKING=yes

HOSTNAME=hadoop03.zhang.com

Hosts文件：

[root@muban ~]# cat /etc/hosts

127.0.0.1 localhost localhost.localdomain localhost4 localhost4.localdomain4

::1 localhost localhost.localdomain localhost6 localhost6.localdomain6

192.168.75.3 hadoop01.zhang.com

192.168.75.8 hadoop02.zhang.com

192.168.75.9 hadoop03.zhang.com

配置完成后重新启动一下服务器，是我们配置文件生效，主要是主机名

创建安装资源存放目录

mkdir source

配置免密码登录：

创建密钥

[root@hadoop01 ~]# ssh-keygen -t rsa

拷贝密钥

ssh-copy-id -i ~/.ssh/id\_rsa.pub [root@192.168.75.3](mailto:root@192.168.75.3)

ssh-copy-id -i ~/.ssh/id\_rsa.pub root@192.168.75.8

ssh-copy-id -i ~/.ssh/id\_rsa.pub root@192.168.75.9

上面的操作在每一台机器中都需要操作：

下面配置我们的java环境，因为我们的hadoop是使用java编写的需要配置java依赖环境

[root@hadoop01 ~]# mkdir /usr/local/java

[root@hadoop01 ~]# cd source/

[root@hadoop01 source]# tar -zxvf jdk1.8.0\_111.tar.gz -C /usr/local/java

配置java环境：

环境变量配置：

Vim /etc/profile #添加：

#java env

JAVA\_HOME=/usr/local/java/jdk1.8.0\_111

PATH=$JAVA\_HOME/bin:$PATH

CLASSPATH=.:$JAVA\_HOME/lib/dt.jar:$JAVA\_HOME/lib/tools.jar

export JAVA\_HOME

export PATH

export HISTTIMEFORMAT="[%Y.%m.%d %H:%M:%S]"

export TIME\_STYLE='+%Y-%m-%d %H:%M:%S'

source /etc/profile

使用java –version看是否安装成功

[root@hadoop03 source]# java -version

java version "1.8.0\_111"

Java(TM) SE Runtime Environment (build 1.8.0\_111-b14)

Java HotSpot(TM) 64-Bit Server VM (build 25.111-b14, mixed mode)

出现以上输出说明java环境配置成功

在其他服务器中进行进行一下如上操作

解压配置hadoop环境

[root@hadoop01 source]# tar -zxf hadoop-2.7.3.tar.gz

[root@hadoop01 source]# mv hadoop-2.7.3 /usr/local/

修改配置文件：

切换目录

[root@hadoop01 hadoop]# cd /usr/local/hadoop-2.7.3/etc/hadoop

1\修改

vim hadoop-env.sh

#export JAVA\_HOME=${JAVA\_HOME}

export JAVA\_HOME=/usr/local/java/jdk1.8.0\_111

2\修改

vim core-site.xml

<configuration>

<!--设置namenode节点以及入口-->

<property>

<name>fs.defaultFS</name>

<value>hdfs://192.168.75.3:8020</value>

</property>

<!--设置缓存存放位置，linux会默认清楚缓存，设置以后缓存就不会清楚-->

<property>

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

<value>/usr/local/hadoop-2.7.3/data/tmp</value>

</property>

<!--开启垃圾回收机制-->

<property>

<name>fs.trash.interval</name>

<value>10080</value>

</property>

</configuration>

3\修改

vim hdfs-site.xml

<configuration>

<property>

<name>dfs.replication</name>

<value>3</value>

</property>

<!--设置权限-->

<property>

<name>dfs.permissions.enabled</name>

<value>false</value>

</property>

<!--设置namenode的节点和入口地址-->

<property>

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

<value>192.168.75.3:50070</value>

</property>

<!--设置secondary.namenode的节点和入口地址-->

<property>

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

<value>192.168.75.9:50090</value>

</property>

<!--设置一下卸载节点的文件 将对应的节点主机名写进去 使用命令刷新即可卸载-->

<property>

<name>dfs.hosts.exclude</name> <value>/usr/local/hadoop-2.7.3/etc/hadoop/dfs-hosts-exclude.conf</value>

</property>

</configuration>

4\修改

mapred-site.xml

首先拷贝一下模板

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

修改添加如下内容：

<configuration>

<!--制定我们的框架为yarn-->

<property>

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

<value>yarn</value>

</property>

<!--制定jobhistory-->

<property>

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

<value>192.168.75.3:10020</value>

</property>

<!--制定jobhistory的web端口-->

<property>

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

<value>192.168.75.3:19888</value>

</property>

<!--开启ubertask模式，针对mr的优化-->

<property>

<name>mapreduce.job.ubertask.enable</name>

<value>true</value>

</property>

</configuration>

5、修改

vim yarn-site.xml

<configuration>

<!-- Site specific YARN configuration properties -->

<!--制定我们的mapreduce为shuffle-->

<property>

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

<value>mapreduce\_shuffle</value>

</property>

<!--指定resourcemanager地址-->

<property>

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

<value>192.168.75.8</value>

</property>

<property>

<name>yarn.resourcemanger.resource-tracker.address</name>

<value>192.168.75.8:8031</value>

</property>

<!--web Application proxy 安全任务主机-->

<property>

<name>yarn.web-proxy.address</name>

<value>192.168.75.8:8888</value>

</property>

<!--开启日志-->

<property>

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

<value>true</value>

</property>

<!--日志删除时间 -1禁用，单位是秒-->

<property>

<name>yarn.log-aggregation.retain-seconds</name>

<value>604800</value>

</property>

<!--yarn内存 根据自己的主机设置 建议设置大一点 下面的这个是测试 不能设置太小，太小会导致NodeManager进程关闭-->

<property>

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

<value>1024</value>

</property>

<!--yarn-CPU 根据自己的主机设置 建议设置大 一点 下面的这个是测试-->

<property>

<name>yarn.nodemanager.resource.cpu-vcores</name>

<value>1</value>

</property>

</configuration>

6\修改

[root@hadoop01 hadoop]# cat slaves

192.168.75.3

192.168.75.8

192.168.75.9

进行一下namenode格式化：

[root@hadoop01 hadoop-2.7.3]# bin/hadoop namenode –format

18/01/11 12:08:30 INFO common.Storage: Storage directory /usr/local/hadoop-2.7.3/data/tmp/dfs/name has been successfully formatted.

有入上输出证明格式化成功

经我们安装的hadoop拷贝到另外的两台主机中：

在我们的namenode节点中启动我们的服务

[root@hadoop01 hadoop-2.7.3]# sbin/start-dfs.sh

使用jps查看一下启动的进程：

[root@hadoop01 hadoop-2.7.3]# jps

1878 DataNode

1786 NameNode

2109 Jps

[root@hadoop02 local]# jps

1549 DataNode

1647 Jps

[root@hadoop03 local]# jps

1682 Jps

1635 SecondaryNameNode

1545 DataNode

启动yarn，我们配置是在hadoop2中启动的，因此我们就在这个上启动，在其他node上启动会报错：

[root@hadoop02 hadoop-2.7.3]# sbin/start-yarn.sh

这里注意我们配合的时候一定要给yarn分配的内存大于1G要不启动成功后程序会自动退出

启动成功以后我们在使用jps看一下进程：

[root@hadoop02 hadoop-2.7.3]# jps

3590 NodeManager

3976 Jps

1693 ResourceManager

1549 DataNode

[root@hadoop01 hadoop]# jps

1878 DataNode

1786 NameNode

4684 Jps

3453 NodeManager

[root@hadoop03 hadoop]# jps

2816 NodeManager

1635 SecondaryNameNode

1545 DataNode

3242 Jps

会多出nodemanager这个进程

启动jobhistory 启动方式是在namenode节点中启动：

[root@hadoop01 hadoop-2.7.3]# sbin/mr-jobhistory-daemon.sh start historyserver

[root@hadoop01 hadoop-2.7.3]# jps

4448 JobHistoryServer

1878 DataNode

1786 NameNode

4732 Jps

3453 NodeManager

在yarn节点中启动yarn安全防护进程：

[root@hadoop02 hadoop-2.7.3]# sbin/yarn-daemon.sh start proxyserver

[root@hadoop02 hadoop-2.7.3]# jps

3590 NodeManager

2392 WebAppProxyServer

1693 ResourceManager

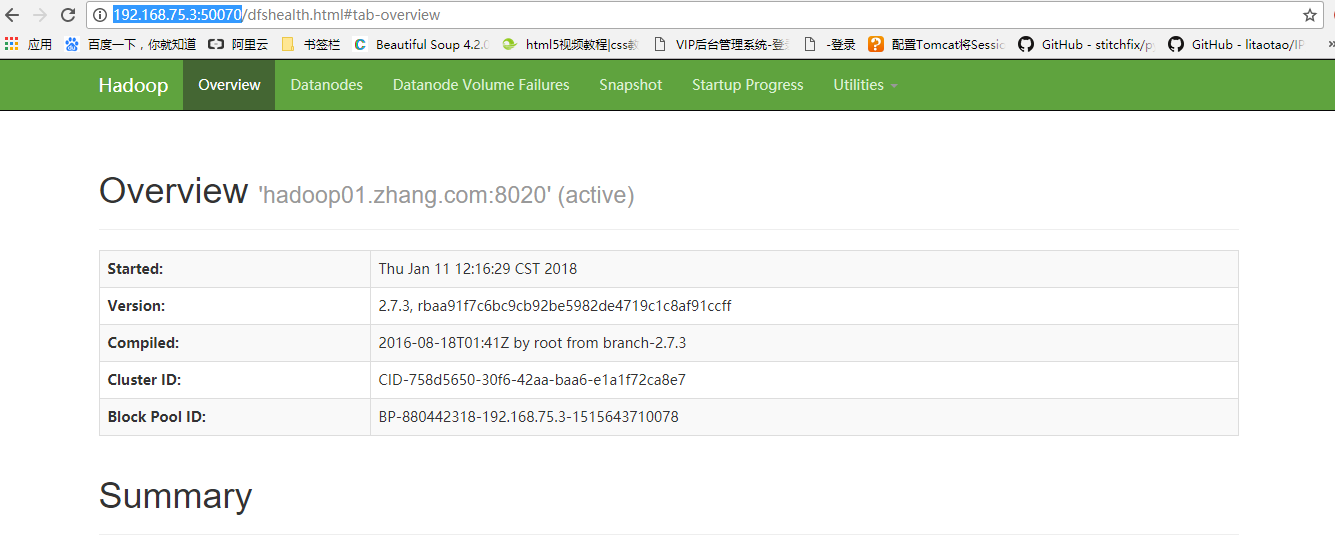
4013 Jps

1549 DataNode

通过web端查看节点情况：

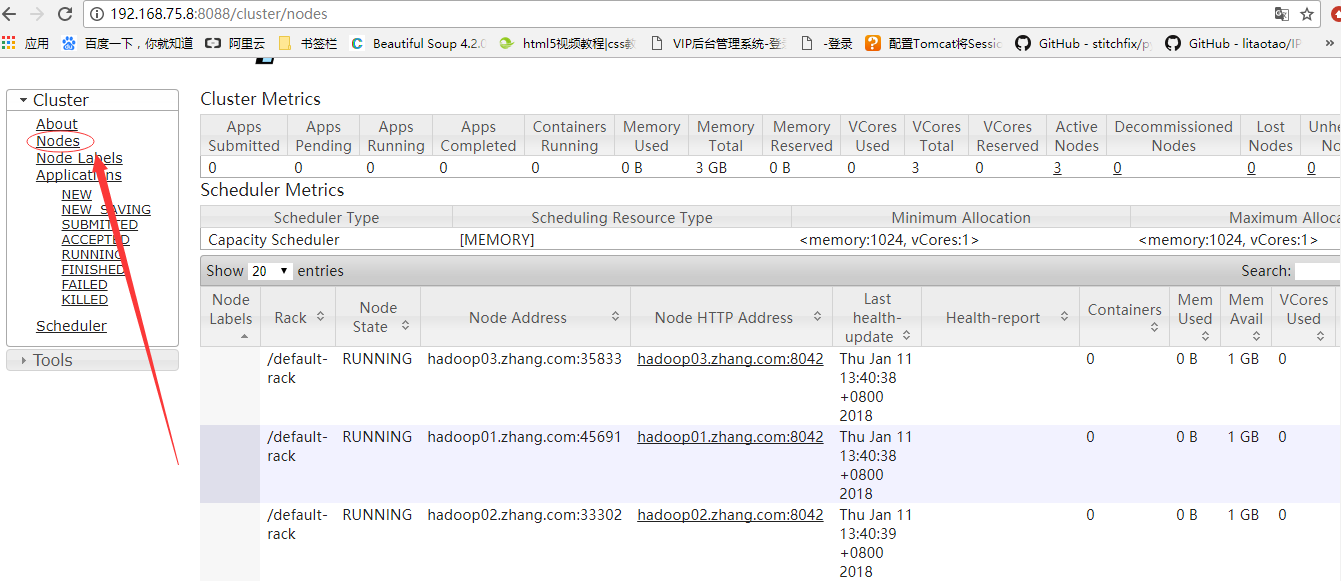
查看hdfs:

访问：<http://192.168.75.3:50070>



查看yarn

访问: http://192.168.75.8:8088

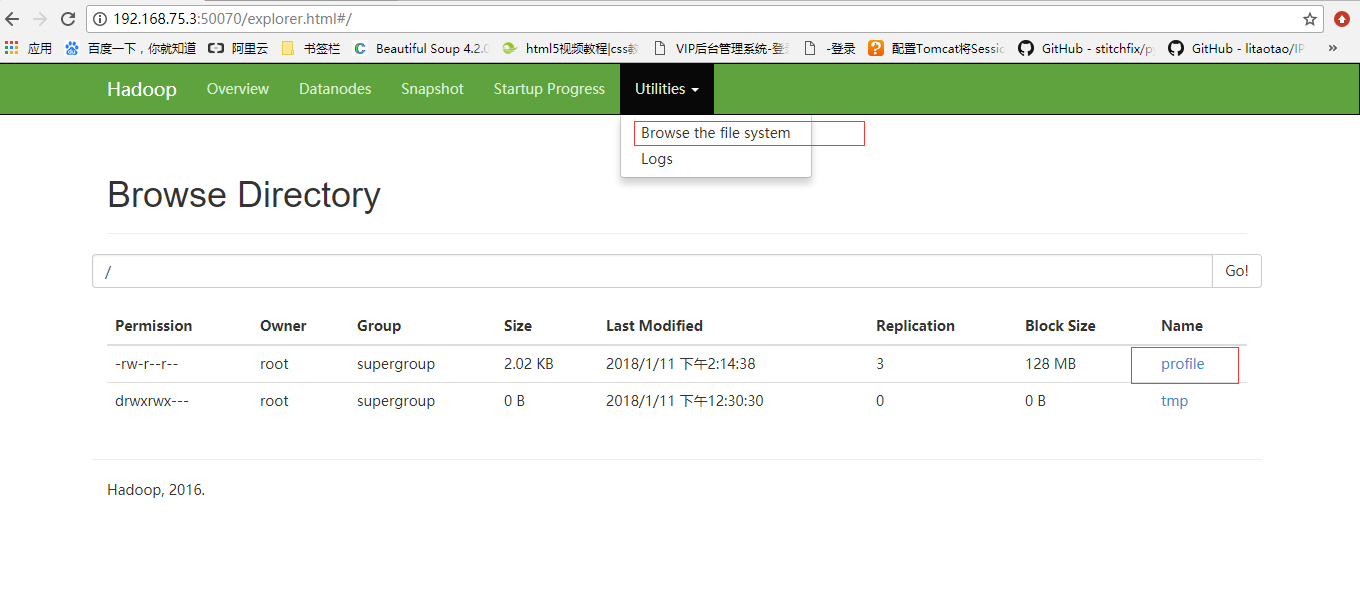


可以看到存活的节点

测试一下集群

上传文件：

[root@hadoop01 hadoop-2.7.3]# bin/hdfs dfs -put /etc/profile /profile

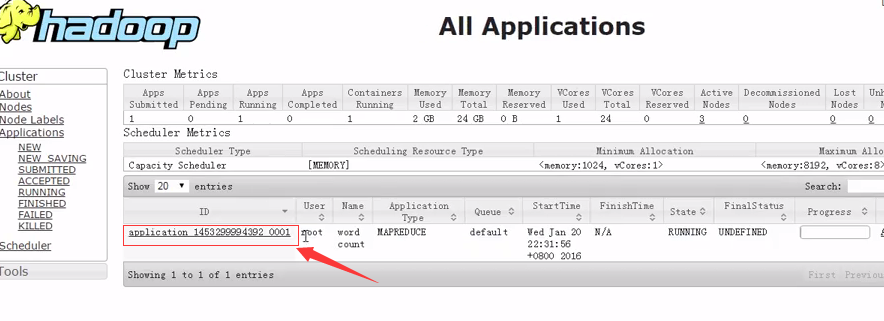


通过界面可以查看到我们的文件已经上传成功

运行一个词频统计：

[root@hadoop01 hadoop-2.7.3]# bin/hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.3.jar wordcount /profile /out

我们在我们的yarn界面中可以看到启动了一个计算进程：



可以看到多了一个进程，我们可以点击进入看到具体的进程状态

添加新的节点进入集群

Hadoop4

设置主机名：

[root@hadoop04 ~]# cat /etc/sysconfig/network

NETWORKING=yes

HOSTNAME=hadoop04.zhang.com

关闭防火墙和selinux:

修改hosts 加入hadoop4的解析：所有的节点都需要添加

[root@hadoop01 hadoop-2.7.3]# cat /etc/hosts

192.168.75.3 hadoop01.zhang.com

192.168.75.8 hadoop02.zhang.com

192.168.75.9 hadoop03.zhang.com

192.168.75.10 hadoop04.zhang.com

生成密钥，可设置免密码登录：

ssh-keygen –t rsa

发布自己的密钥给集群内的其他主机

ssh-copy-id -i ~/.ssh/id\_rsa.pub [root@hadoop01.zhang.com](mailto:root@hadoop01.zhang.com)

ssh-copy-id -i ~/.ssh/id\_rsa.pub root@hadoop02.zhang.com

ssh-copy-id -i ~/.ssh/id\_rsa.pub root@hadoop03.zhang.com

ssh-copy-id -i ~/.ssh/id\_rsa.pub root@hadoop04.zhang.com

同时其他主机的密钥也要在给hadoop4分享一下：

下面开始添加主机，

主要修改的是namenode和ResourceManager节点：

首先修改一下ResourceManager节点在slaves中添加新的节点：

[root@hadoop02 hadoop-2.7.3]# cat etc/hadoop/slaves

192.168.75.3

192.168.75.8

192.168.75.9

192.168.75.10

加入hadoop4对应的主机

在其他主机中也做一下对应的修改：

在hadoop4中搭建java环境：

拷贝java安装包：

scp -r /usr/local/java/ root@hadoop04.zhang.com:/usr/local/

修改/etc/profile文件，加入java环境变量配置：

拷贝一下hadoop安装包到hadoop04中：

# scp -r /usr/local/hadoop-2.7.3/ root@hadoop04.zhang.com:/usr/local/

在相关节点启动我们的hdfs和yarn进程：

[root@hadoop01 hadoop-2.7.3]# sbin/start-dfs.sh

[root@hadoop02 hadoop-2.7.3]# sbin/start-yarn.sh

你也可以使用刷新节点：

刷新hdfs节点命令

# bin/hdfs dfsadmin –refreshNodes

刷新计算节点：

# bin/yarn rmadmin -refreshNodes

动态卸载我们添加的节点：

在配置文件hdfs-site.xml中加入如下配置：

<!--设置一下卸载节点的文件 将对应的节点主机名写进去 使用命令刷新即可卸载-->

<property>

<name>dfs.hosts.exclude</name> <value>/usr/local/hadoop-2.7.3/etc/hadoop/dfs-hosts-exclude.conf</value>

</property>

创建dfs-hosts-exclude.conf文件，这个文件中可以添加我们要卸载的主机名：

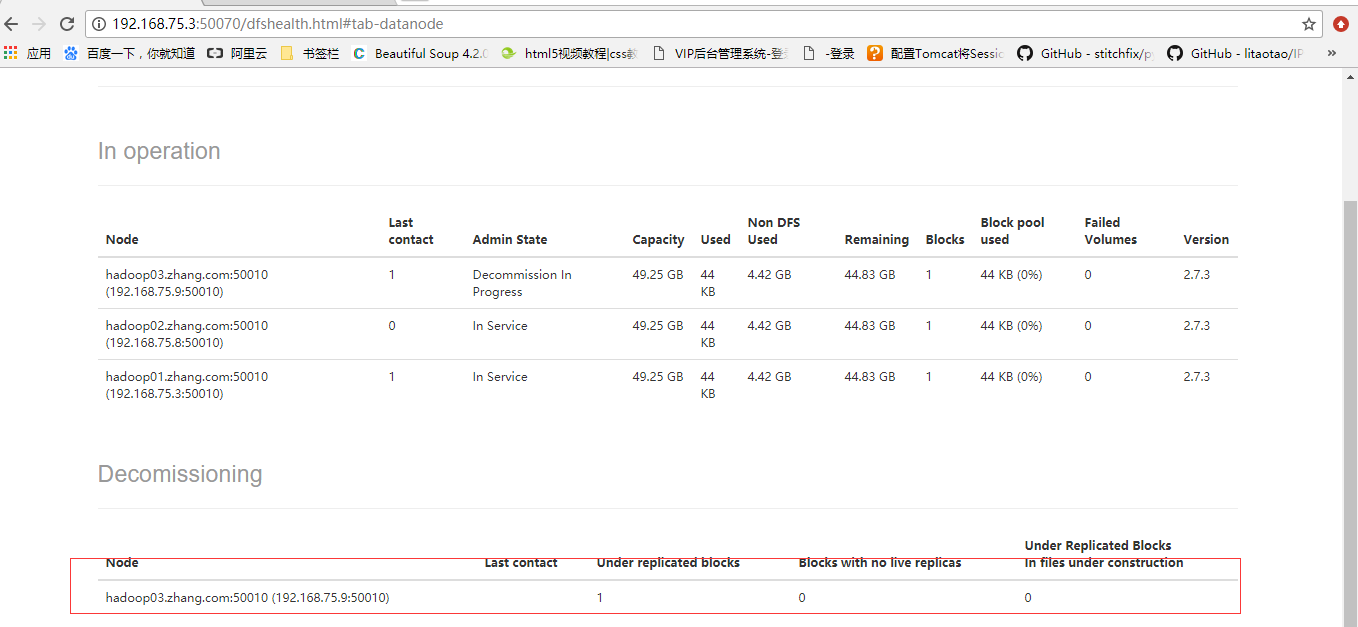
如：

[root@hadoop01 hadoop]# cat dfs-hosts-exclude.conf

hadoop4.zhang.com

刷新一下：

# bin/hdfs dfsadmin –refreshNodes



界面中我们可以看到正在卸载的主机：

同是我们也是可以使用命令查看的：

# bin/hdfs dfsadmin –report

下载了hdfs节点，我们在尝试卸载一下yarn计算节点：

方法如下：

首先在yarn-site.xml中加入如下配置

[root@hadoop02 hadoop]# vim yarn-site.xml

<!--设置一下卸载节点的文件 将对应的节点主机名写进去 使用命令刷新即可卸载-->

<property>

<name>yarn.resourcemanager.nodes.exclude-path</name>

<value>/usr/local/hadoop-2.7.3/etc/hadoop/yarn-hosts-exclude.conf</value>

</property>

创建制定的卸载配置文件：

# touch /usr/local/hadoop-2.7.3/etc/hadoop/yarn-hosts-exclude.conf

加入你要卸载的主机名：

[root@hadoop02 hadoop-2.7.3]# cat /usr/local/hadoop-2.7.3/etc/hadoop/yarn-hosts-exclude.conf

hadoop04.zhang.com

刷新计算节点：

# bin/yarn rmadmin –refreshNodes

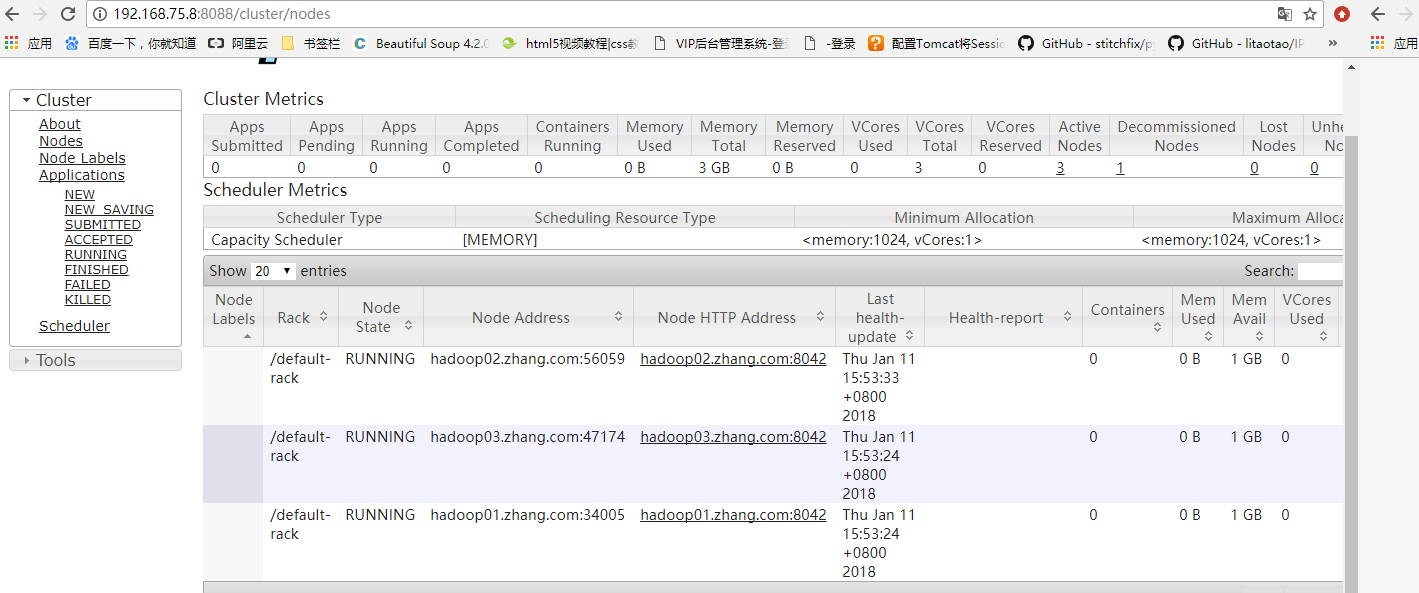
其他yarn的shell操作可以在如下链接中学习：

http://hadoop.apache.org/docs/r2.7.3/hadoop-yarn/hadoop-yarn-site/YarnCommands.html

yarn的高可用配置参考：

http://hadoop.apache.org/docs/r2.7.3/hadoop-yarn/hadoop-yarn-site/ResourceManagerHA.html

在yarn界面中查看发现计算节点已经卸载：



**Yarn高可用**

参考文档：

http://hadoop.apache.org/docs/r2.7.3/hadoop-yarn/hadoop-yarn-site/ResourceManagerHA.html

在原有的集群基础上操作：

在yarn-site.xml中添加如下配置：

<!--开启yarn的ha-->

<property>

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

<value>true</value>

</property>

<!--制定集群的id-->

<property>

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

<value>cluster1</value>

</property>

<!--指定运行resourcemanager的两个id,也就是rm1,rm2-->

<property>

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

<value>rm1,rm2</value>

</property>

<property>

<!--制定rm1的地址-->

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

<value>hadoop02.zhang.com</value>

</property>

<!--制定rm2的地址-->

<property>

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

<value>hadoop03.zhang.com</value>

</property>

<!--制定rm1的端口-->

<property>

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

<value>hadoop02.zhang.com:8088</value>

</property>

<!--制定rm2的端口-->

<property>

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

<value>hadoop03.zhang.com:8088</value>

</property>

<!--配置自己的zk集群-->

<property>

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

<value> hadoop01.zhang.com:2181,hadoop02.zhang.com:2181,hadoop03.zhang.com:2181</value>

</property>

配置完成，需要经配置的hadoop同步到其他hadoop服务器中

我们需要在hadoop02中启动我们的yarn:

[root@hadoop02 hadoop-2.7.3]# ./sbin/start-yarn.sh

我们使用jps核实一下进程：

[root@hadoop01 hadoop]# jps

1492 Jps

1214 QuorumPeerMain

1374 NodeManager

[root@hadoop02 hadoop-2.7.3]# jps

1520 NodeManager

1552 Jps

1414 ResourceManager

1114 QuorumPeerMain

[root@hadoop03 zookeeper]# jps

1847 NodeManager

1691 QuorumPeerMain

1965 Jps

启动完成后最重要的一步来了，那就是我们需要在我们的备份节点中手动启动一下ResourceManager进程：

[root@hadoop03 hadoop-2.7.3]#

sbin/yarn-daemon.sh start resourcemanager

starting resourcemanager, logging to /usr/local/hadoop-2.7.3/logs/yarn-root-resourcemanager-hadoop03.zhang.com.out

[root@hadoop03 hadoop-2.7.3]# jps

1847 NodeManager

2009 ResourceManager

2041 Jps

1691 QuorumPeerMain

通过浏览器访问测试：

<http://192.168.75.8:8088/cluster>

当我们访问<http://192.168.75.9:8088/cluster>的时候，浏览器会主动跳转到<http://192.168.75.8:8088/cluster上，因为我们的hadoop03>只是我们的备份

在shell命令行中使用命令核实：

[root@hadoop03 hadoop-2.7.3]# bin/yarn rmadmin -getServiceState rm1

active

[root@hadoop03 hadoop-2.7.3]# bin/yarn rmadmin -getServiceState rm2

Standby

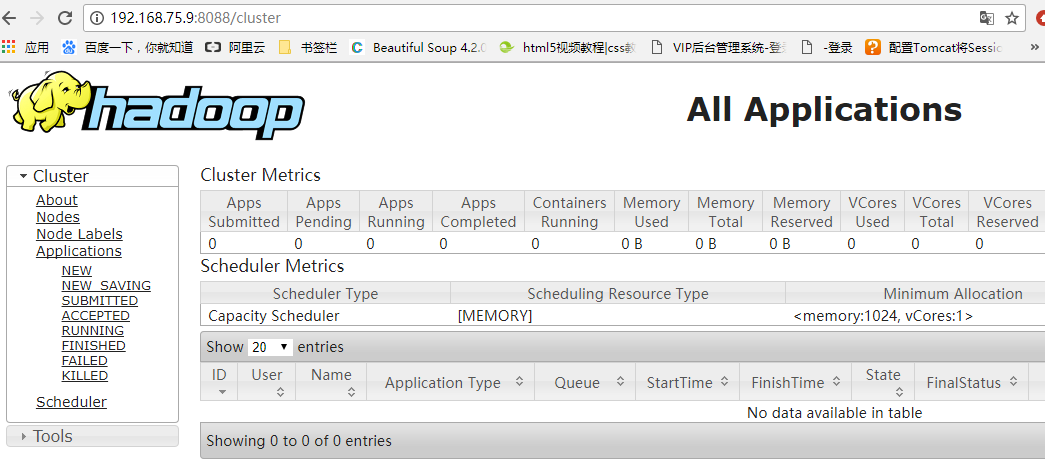
我们尝试在hadoop02中杀掉我们的yarn进程，再次核实一下对应几点的状态：

[root@hadoop03 hadoop-2.7.3]# bin/yarn rmadmin -getServiceState rm2

active

发现我们的hadoop03已经从备份状态转换为活跃状态

直接在浏览器中访问：



访问不在跳转了：

我们链接我们的zk可以查看到数据：



为了使我们程序完美，我们需要配置ResourceManager支持重启：

参考：

<http://hadoop.apache.org/docs/r2.7.3/hadoop-yarn/hadoop-yarn-site/ResourceManagerRestart.html>

配置了ResourceManager restart后，会将一些程序的信息存储在一个地方，我们这里建议存储在zk中：

修改配置文件：

Yarn-site.xml:

<!--开启resourcemanager restart-->

<property>

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

<value>true</value>

</property>

<!--把信息存储在zk中-->

<property>

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

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

</property>

这里在单独配置的时候是需要配置一下ZK的地址的，但是我们上一部已经配置了，这里就可以省略：

# 配置了NodeManager Restart

参考：

修改配置文件：

Yarn-site.xml:

<!--开启NodeManager Restart-->

<property>

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

<value>true</value>

</property>

<!--将信息存储在文件中-->

<property>

<name>yarn.nodemanager.recovery.dir</name>

<value>/usr/local/hadoop-2.7.3/data/yarn-nm-recovery</value>

</property>

<!--设置nodemanager的存储地址-->

<property>

<name>yarn.nodemanager.address</name>

<value>0.0.0.0:45454</value>

配置以后同步我们的配置文件到其他主机上：

重启服务即可，这个和我们的一些信息就会存储在zk和我们制定的文件中

[root@hadoop01 hadoop-2.7.3]# ./sbin/start-dfs.sh

[root@hadoop01 hadoop-2.7.3]# jps

2738 NameNode

2968 NodeManager

2840 DataNode

3197 Jps

1214 QuorumPeerMain

[root@hadoop02 hadoop-2.7.3]# ./sbin/start-yarn.sh

[root@hadoop02 hadoop-2.7.3]# jps

3285 Jps

2965 DataNode

3161 NodeManager

1114 QuorumPeerMain

3053 ResourceManager

[root@hadoop03 hadoop-2.7.3]# sbin/yarn-daemon.sh start resourcemanager

[root@hadoop03 hadoop-2.7.3]# jps

3506 ResourceManager

3682 Jps

1691 QuorumPeerMain

3340 DataNode

3422 NodeManager

3567 SecondaryNameNode

**namenode高可用**

参考连接：

http://hadoop.apache.org/docs/r2.7.3/hadoop-project-dist/hadoop-hdfs/HDFSHighAvailabilityWithQJM.html

配置如下：

修改hdfs-site.xml

<configuration>

<!--指定hdfsd的nameservice的名称为mycluster-->

<property>

<name>dfs.nameservices</name>

<value>mycluster</value>

</property>

<!--指定两个namenode的名称-->

<property>

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

<value>nn1,nn2</value>

</property>

<!--配置nn1，nn2的rpc通信端口-->

<property>

<name>dfs.namenode.rpc-address.mycluster.nn1</name>

<value>hadoop01.zhang.com:8020</value>

</property>

<property>

<name>dfs.namenode.rpc-address.mycluster.nn2</name>

<value>hadoop02.zhang.com:8020</value>

</property>

<!--设置namenode的节点和入口地址-->

<property>

<name>dfs.namenode.http-address.mycluster.nn1</name>

<value>hadoop01.zhang.com:50070</value>

</property>

<property>

<name>dfs.namenode.http-address.mycluster.nn2</name>

<value>hadoop02.zhang.com:50070</value>

</property>

<!--指定namenode的元数据存储在journalnode中的路径-->

<property>

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

<value>qjournal://hadoop01.zhang.com:8485;hadoop02.zhang.com:8485;hadoop03.zhang.com:8485/mycluster</value>

</property>

<!--配置失败自动切换的方式-->

<property>

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

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

</property>

<!--配置隔离机制-->

<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\_rsa</value>

</property>

<!--指定连接的用户名和端口 没有特殊情况不需要使用

<property>

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

<value>sshfence([[root][:22]])</value>

</property>-->

<property>

<name>dfs.ha.fencing.ssh.connect-timeout</name>

<value>30000</value>

</property>

<!--设置备份数量-->

<property>

<name>dfs.replication</name>

<value>3</value>

</property>

<!--设置权限-->

<property>

<name>dfs.permissions.enabled</name>

<value>false</value>

</property>

<!--设置备份数量-->

<property>

<name>dfs.replication</name>

<value>3</value>

</property>

<!--设置权限-->

<property>

<name>dfs.permissions.enabled</name>

<value>false</value>

</property> </configuration>

修改core-site.xml

<configuration>

<!--指定nameservice的名称为mycluster-->

<property>

<name>fs.defaultFS</name>

<value>hdfs://mycluster</value>

</property>

<!--编辑日志文件存储的路径-->

<property>

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

<value>/usr/local/hadoop-2.7.3/data/jn</value>

</property>

<!--设置缓存存放位置，linux会默认清楚缓存，设置以后缓存就不会清楚-->

<property>

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

<value>/usr/local/hadoop-2.7.3/data/tmp</value>

</property>

<!--客户端RPC重试次数-->

<property>

<name>ipc.client.connect.max.retries</name>

<value>100</value>

</property>

<!--客户端重试时间间隔-->

<property>

<name>ipc.client.connect.retry.interval</name>

<value>10000</value>

</property> </configuration>

创建一下编辑日志文件的目录： 这个操作需要在其他主机上都操作一下：

[root@hadoop01 data]# mkdir jn

[root@hadoop01 jn]# pwd

/usr/local/hadoop-2.7.3/data/jn

scp -r hadoop-2.7.3/ [root@hadoop03.zhang.com:/usr/local/](mailto:root@hadoop03.zhang.com:/usr/local/)

scp -r hadoop-2.7.3/ root@hadoop02.zhang.com:/usr/local/

现在我们的hadoop01中启动我们的编辑日志：

[root@hadoop01 hadoop-2.7.3]# sbin/hadoop-daemon.sh start journalnode

使用相同的方式在其他两台服务器中启动：

[root@hadoop01 hadoop-2.7.3]# jps

4025 JournalNode

4074 Jps

3708 QuorumPeerMain

[root@hadoop02 hadoop-2.7.3]# jps

1324 QuorumPeerMain

1452 JournalNode

1501 Jps

[root@hadoop03 hadoop-2.7.3]# jps

1472 JournalNode

1521 Jps

1320 QuorumPeerMain

第一步，我们需要进行一下格式化：

[root@hadoop01 hadoop-2.7.3]# bin/hadoop namenode –format

同步一下配置

在我们定义的第二个namenode中同步信息：

[root@hadoop02 hadoop-2.7.3]# bin/hdfs namenode –help

查看一下帮助

[root@hadoop02 hadoop-2.7.3]# bin/hdfs namenode –bootstrapStandby

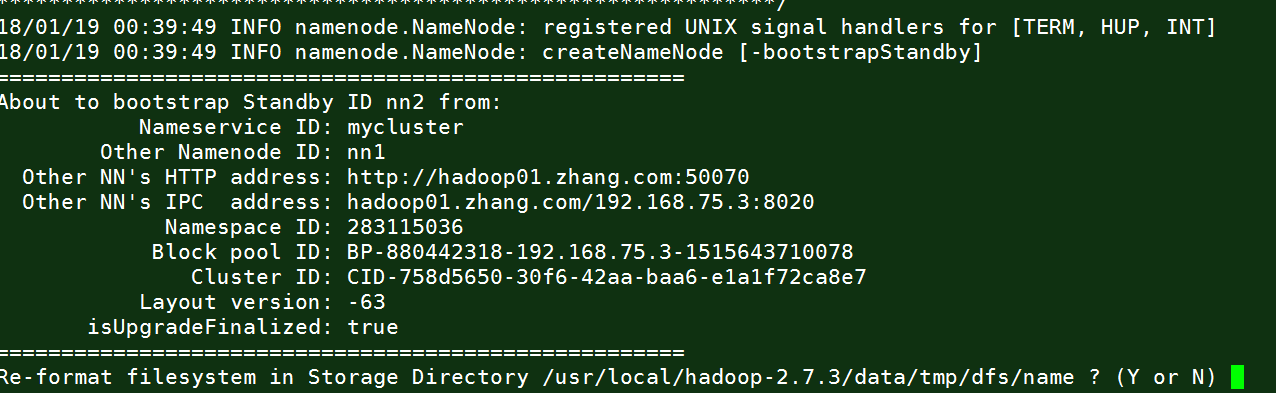
使用如上命令进行一下同步，同步的时候要注意一个细节，就是我们的namenode已经启动了，要不会报错的：

Namenode的启动方式：

[root@hadoop01 hadoop-2.7.3]# sbin/hadoop-daemon.sh start namenode

再次同步一下源数据信息：（在备用节点操作）

[root@hadoop02 hadoop-2.7.3]# bin/hdfs namenode -bootstrapStandby



启动三台的datanode

[root@hadoop01 hadoop-2.7.3]# sbin/hadoop-daemon.sh start datanode

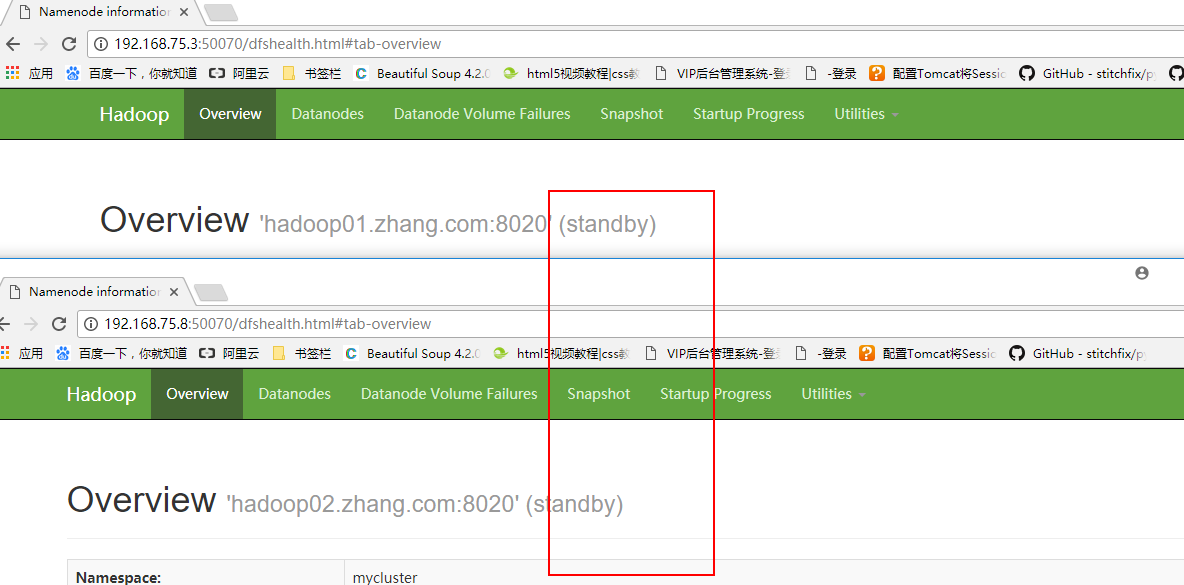
查看一下集群的状态：

[root@hadoop02 hadoop-2.7.3]# bin/hdfs haadmin -getServiceState nn1

standby

[root@hadoop02 hadoop-2.7.3]# bin/hdfs haadmin -getServiceState nn2

Standby

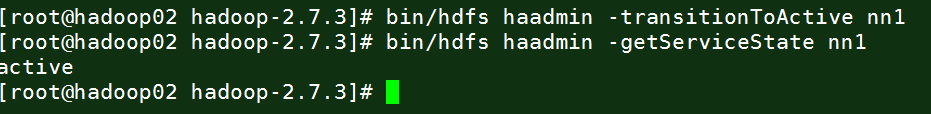


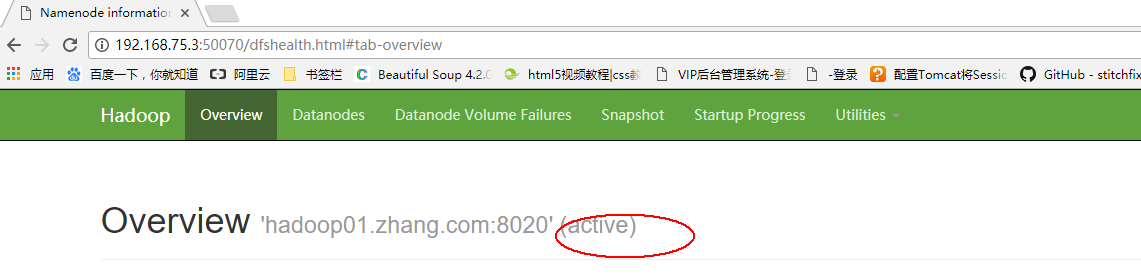
都是备份状态，我们可以尝试一下手动激活：

[root@hadoop02 hadoop-2.7.3]# bin/hdfs haadmin -transitionToActive nn1

[root@hadoop02 hadoop-2.7.3]# bin/hdfs haadmin -getServiceState nn1

active





这个需要我们手动切换，下面我们进行一下自动的故障转移：

修改hdfs-site.xml

<!--配置故障自定转移 HA--->

<property>

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

<value>true</value>

</property>

修改core-site.xml

<!--配置自己的zk集群-->

<property>

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

<value>hadoop01.zhang.com:2181,hadoop02.zhang.com:2181,hadoop03.zhang.com:2181</value>

</property>

记得一定要启动我们的ZK集群

同步一下配置：

scp -r hadoop-2.7.3/ root@hadoop02.zhang.com:/usr/local/

首先启动我们的namenode节点：

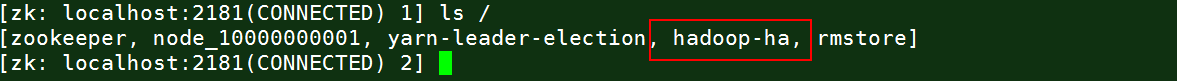
[root@hadoop01 hadoop-2.7.3]# sbin/hadoop-daemon.sh start namenode

[root@hadoop02 hadoop-2.7.3]# sbin/hadoop-daemon.sh start namenode

格式化zkfc

root@hadoop01 hadoop-2.7.3]# bin/hdfs zkfc -formatZK

连接zk集群看一下：

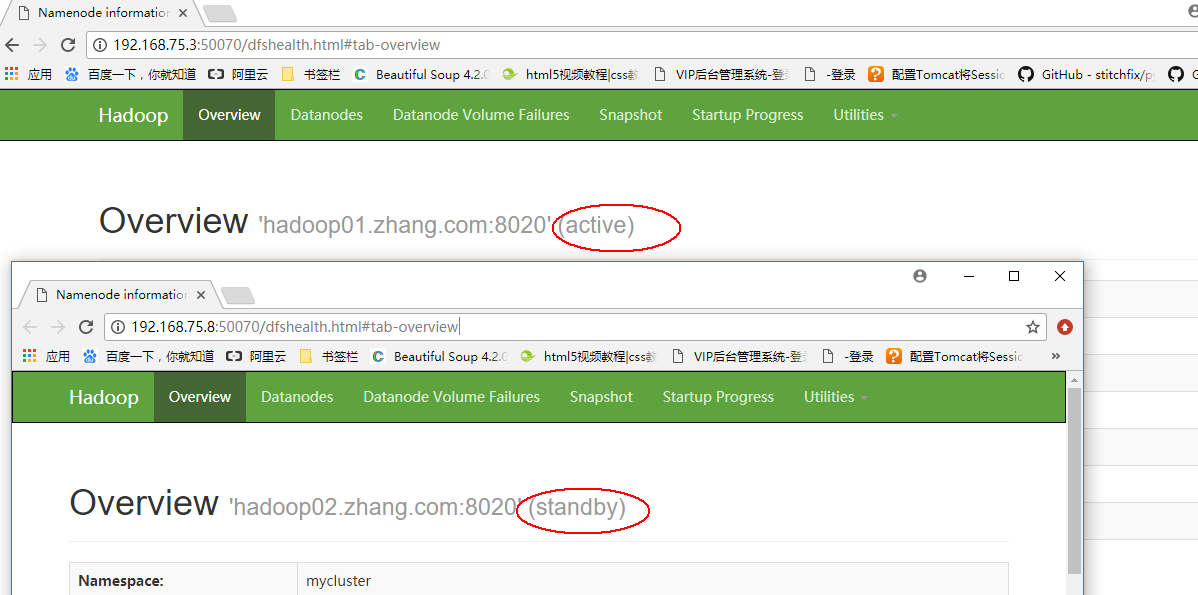


重启一下进程：

[root@hadoop01 hadoop-2.7.3]# sbin/stop-dfs.sh

[root@hadoop01 hadoop-2.7.3]# sbin/start-dfs.sh

登入浏览器核实：



测试一下，模拟宕机namenode

[root@hadoop01 hadoop-2.7.3]# jps

4289 NodeManager

4994 NameNode

5284 JournalNode

2405 QuorumPeerMain

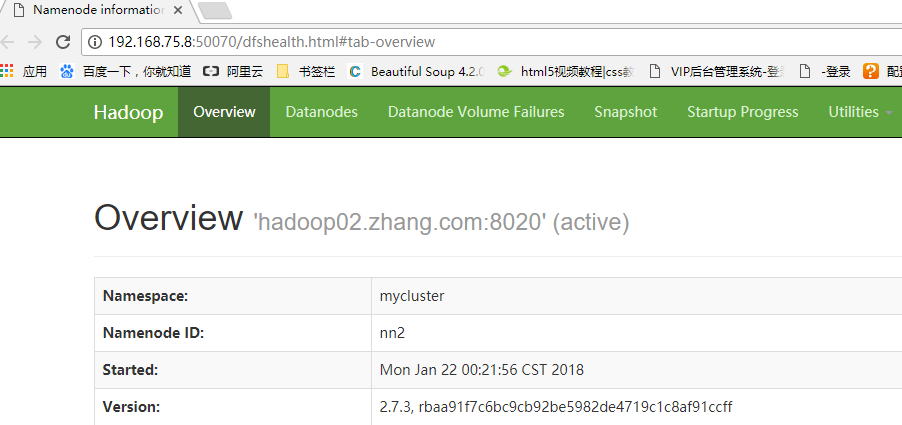
5093 DataNode

5513 Jps

5421 DFSZKFailoverController

杀掉namenode进程

[root@hadoop01 hadoop-2.7.3]# kill -9 4994



核实一下已经切换