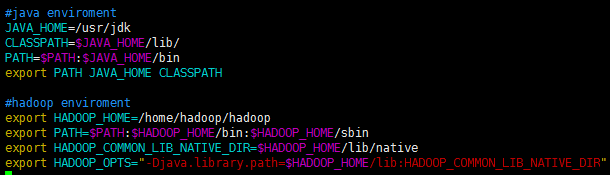
**Hadoop学习笔记**

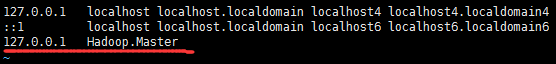
Hadoop环境搭建

由于之前没有记录，简单贴出来配置文件，看一下吧。

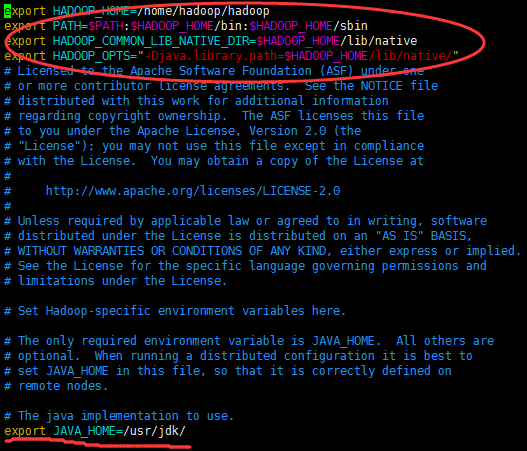
首先。编辑 /etc/profile 配置jdk环境以及hadoop环境



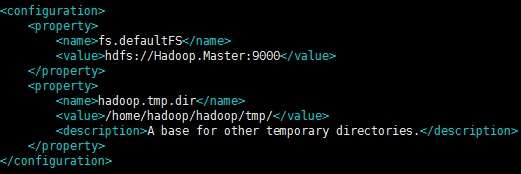
然后，配置一下hostname



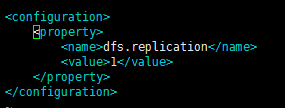
第一个配置文件,hadoop目录下etc/**hadoop-env.sh**



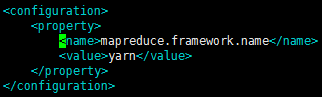
第二个配置文件core-site.xml。**第一个property配置的是HDFS的NameNode的地址，第二个property配置的内容用来指定Hadoop运行时产生的文件的存放目录。**



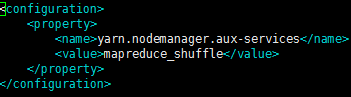
第三个配置文件hdfs-site.xml。**该配置文件用来指定HDFS保存数据副本的数量（现在是伪分布式，所以数量是1，将来的集群副本数量默认是3）。**



第四个配置文件mapred-site.xml。我们在hadoop目录下发现文件列表中只有mapred-site.xml.template而没有mapred-site.xml，因此我们需要先把mapred-site.xml.template的后缀.template去掉（即重命名）



第五个配置文件yarn-site.xml。**需要说明的是，配置的内容是NodeManager获取数据的方式shuffle。**



然后我们需要输入命令格式化我们的hdfs

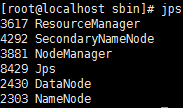
hdfs namenode -format

进入sbin目录下，启动dfs和yarn

./start-dfs.sh

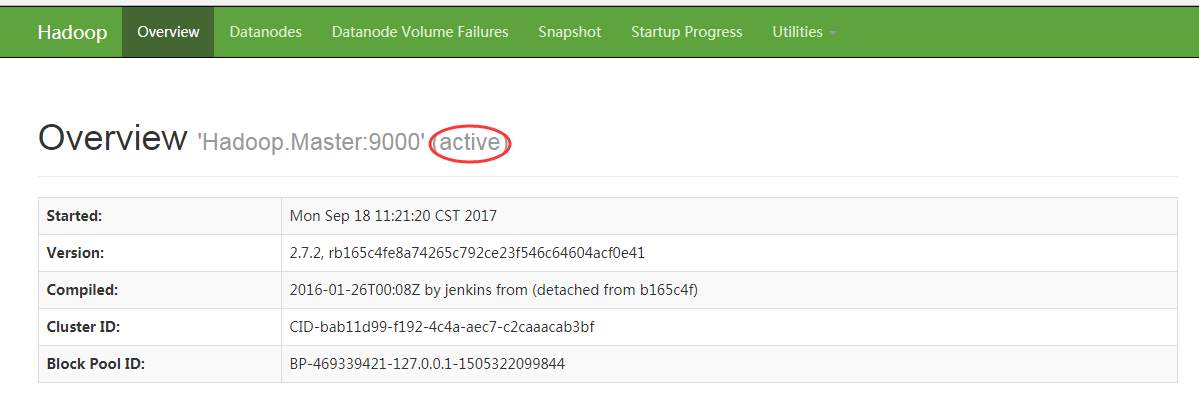
./start-yarn.sh

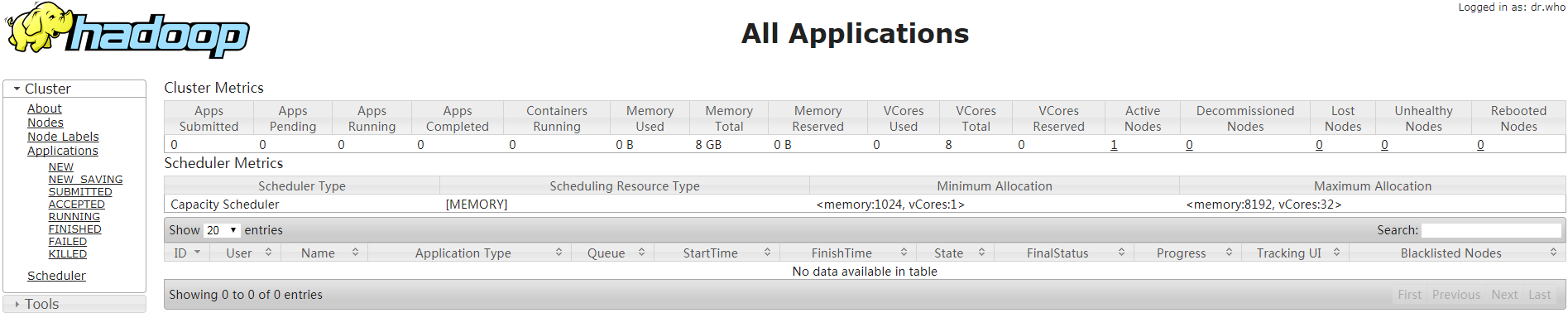
然后，用jps 命令查看启动的进程数，应该为6个才是正常



http://192.168.8.88:50070     (hdfs管理界面)

http://192.168.8.88:8088       (yarn管理界面)

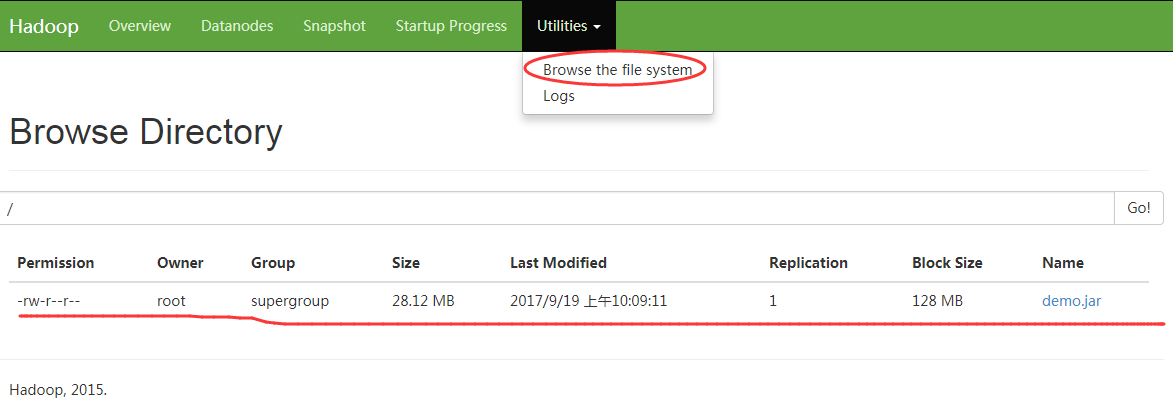




上传一个文件

格式：hadoop fs -put < local file > < hdfs file >

hadoop fs -put /home/testhu/demo.jar hdfs://Hadoop.Master:9000/demo.jar



查看hdfs文件命令

hadoop fs -ls hdfs://Hadoop.Master:9000/



**注：hdfs://Hadoop.Master:9000/代表hdfs的根路径，就是core-site.xml配置文件中配置的。**

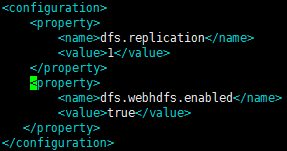
下载命令：

hadoop fs -get hdfs://Hadoop.Master:9000/demo.jar

问题记录：

背景：HDFS 页面中File information无法下载文件。

原因：hdfs-site.xml必须将dfs.webhdfs.enabled属性设置为true，才能使用webhdfs的一些功能。于是修改配置文件



停止hdfs 以及 yarn

stop-dfs.sh

stop-yarn.sh

然后重新启动hdfs以及yarn，此时出现问题，namenode无法启动。

查看hadoop-root-namenode-localhost.localdomain.log 这个namenode的日志发现报错。

java.io.IOException: There appears to be a gap in the edit log. We expected txid 1, but got txid 17.

网上说是元数据被破坏了，我就停止然后在启动元数据就坏了？搞不懂。

这个命令能解决namenode无法启动的问题：hadoop namenode -recover

但是不用还好一用，又来问题了。

datanode又无法启动了

查看datanode的日志hadoop-root-datanode-localhost.localdomain.log 发现报错：

2017-09-19 14:29:56,688 WARN org.apache.hadoop.hdfs.server.common.Storage: java.io.IOException: Incompatible clusterIDs in /home/hadoop/hadoop/tmp/dfs/data: namenode clusterID = CID-26df84e6-a33b-4c02-8f1f-a40042ab1b89; datanode clusterID = CID-bab11d99-f192-4c4a-aec7-c2caaacab3bf

2017-09-19 14:29:56,692 FATAL org.apache.hadoop.hdfs.server.datanode.DataNode: Initialization failed for Block pool <registering> (Datanode Uuid unassigned) service to Hadoop.Master/127.0.0.1:9000. Exiting.

java.io.IOException: All specified directories are failed to load.

…

….

然后又是百度一下。。。

网友给出的答案：

**原因是因为datanode的clusterID 和 namenode的clusterID 不匹配。**

**打开hdfs-site.xml里配置的datanode和namenode对应的目录，分别打开current文件夹里的VERSION，可以看到clusterID项正如日志里记录的一样，确实不一致，修改datanode里VERSION文件的clusterID 与namenode里的一致，再重新启动dfs（执行start-dfs.sh）再执行jps命令可以看到datanode已正常启动。**

**出现该问题的原因：在第一次格式化dfs后，启动并使用了hadoop，后来又重新执行了格式化命令（hdfs namenode -format)，这时namenode的clusterID会重新生成，而datanode的clusterID 保持不变。**

所以，解决办法就是修改VERSION文件中的clusterID

文件的位置在tmp/dfs/data/current/VERSION

然后修改成与namenode的clusterID一样就行了

namenode,datenode都启动了，然后又来问题了：

HDFS的主页面上显示There are 1 missing blocks. The following files may be corrupted

我修改配置之前上传的文件丢失了。

bin/hadoop fsck /

bin/hadoop dfs -rm /demo.jar  
bin/hadoop fsck /

只能删除了。。。

然后一切回归平静

好多疑问，我觉得和我用hadoop namenode -recover 这个命令有关。

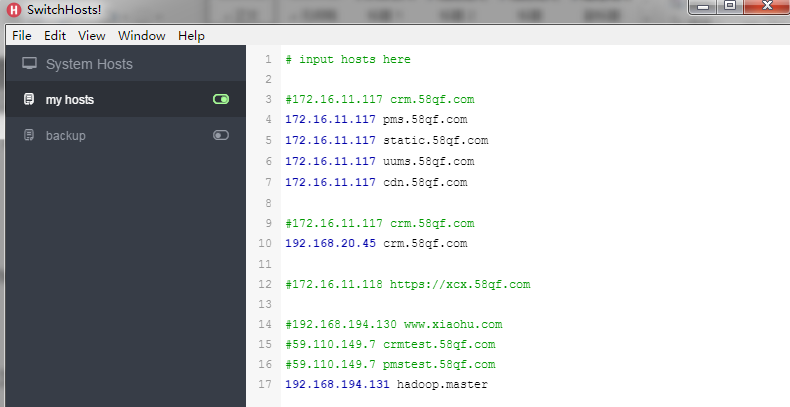
问：

重启一下为啥namenode 就不能启动了？每次namenode启动都要格式化么？

hadoop namenode -recover这个做了些什么？

继续之前的hadoop学习，之前的hadoop的hdfs页面下载文件下载不了，原因是他所跳转的网址是带有host标签的（hadoop.master），而我们访问的机器无法解析这个host标签，也就访问不了。

所以我们需要在访问的机器上配置host（hadoop.master）指向我们的hadoop服务。

由于我本机装了SwitchHosts 所以直接在这上面改了。

配置好之后就可以使用hdfs的web端页面的下载了。

Hadoop教学贴：

<http://blog.csdn.net/u012453843/article/category/6887358/3>

互联网电商项目 淘淘商城从无到有教学贴：

<http://blog.csdn.net/u012453843/article/category/6970308/8>

简单的java下载hdfs中的文件

首先，IDEA的POM文件引入下列的包。

<dependency>  
 <groupId>org.apache.hadoop</groupId>  
 <artifactId>hadoop-common</artifactId>  
 <version>2.7.2</version>  
</dependency>  
<dependency>  
 <groupId>org.apache.hadoop</groupId>  
 <artifactId>hadoop-hdfs</artifactId>  
 <version>2.7.2</version>  
</dependency>

下载文件这两个包就够了，

然后java代码如下：  
FileSystem fs = FileSystem.get(new URI("hdfs://Hadoop.Master:9000/"),new Configuration());  
InputStream is = fs.open(new Path("/demo.jar"));  
OutputStream os = new FileOutputStream("C:\\Users\\Administrator\\Desktop\\dd\\demo.jar");  
IOUtils.copyBytes(is, os, 4096, true);

上面的代码是要下载我hdfs根路径下，上传的一个jar包。

下面是向hdfs上传文件的java代码。

FileSystem fs = FileSystem.get(new URI("hdfs://Hadoop.Master:9000/"),new Configuration(),"root");  
InputStream is = new FileInputStream("C:\\Users\\Administrator\\Desktop\\周报.txt");  
OutputStream os = fs.create(new Path("/test.txt"));  
IOUtils.copyBytes(is, os, 4096, true);

上面的代码是将我桌面上的一个周报文本文件上传到hdfs的根路径下，并重命名为test.txt。

下面是删除代码，将hdfs中的demo.jar删除

/\*删除文件代码\*/  
//第一个参数是要删除的文件或者文件夹  
//第二个参数是是否递归删除，用于删除文件夹以及文件夹下所有文件  
FileSystem fs = FileSystem.get(new URI("hdfs://Hadoop.Master:9000/"),new Configuration(),"root");

Boolean flag = fs.delete(new Path("/demo.jar"),false);  
System.out.println(flag);

下面是创建文件夹代码

/\*创建文件夹\*/

FileSystem fs = FileSystem.get(new URI("hdfs://Hadoop.Master:9000/"),new Configuration(),"root");  
Boolean sign = fs.mkdirs(new Path("/dirTest"));  
System.out.println(sign);

后续的Hadoop知识有则补充，基本都以：

<http://blog.csdn.net/u012453843/article/category/6887358/3>

此教学贴为准