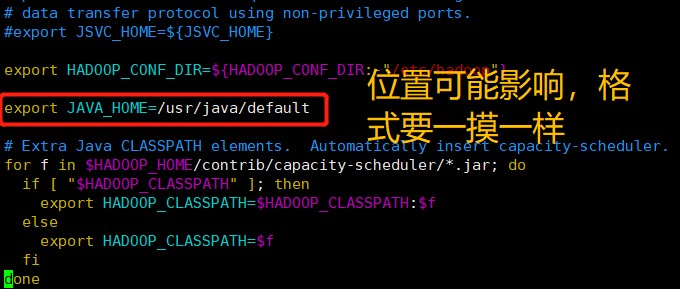
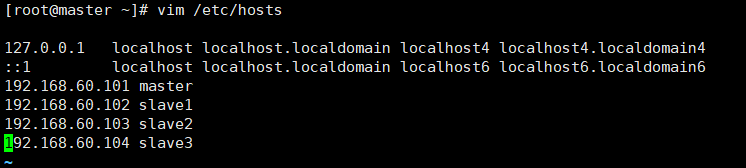
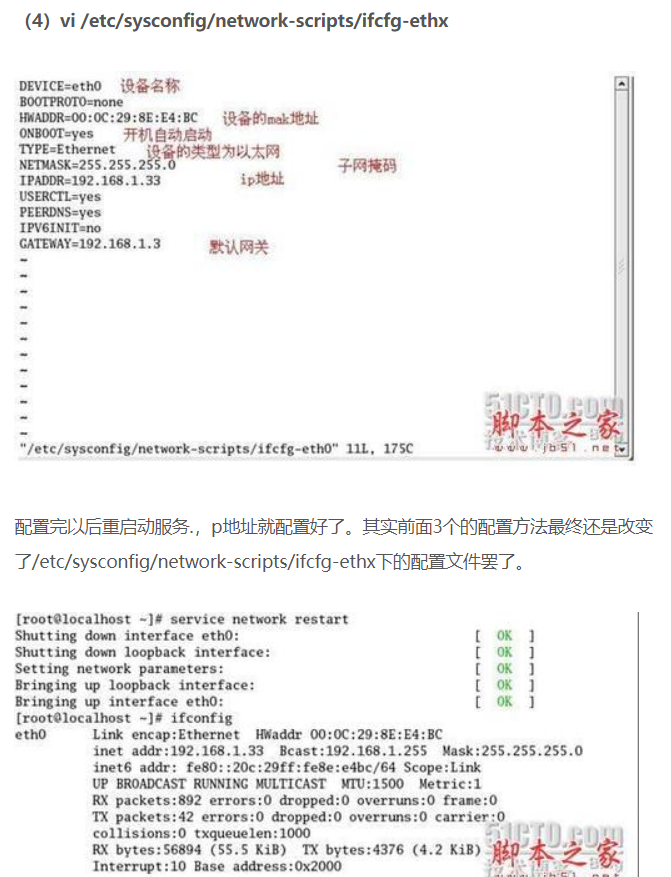
**Linux配置hadoop**

1. **通过xftp将下载好的jdk和hadoop安装包从Windows传到Linux的/usr/local目录下。**
2. **先删除自带的jdk，再进入/usr/local目录下，通过rpm –ivh jdk…安装jdk，安装完后通过java命令看是否安装成功。通过tar –xvf hadoop…解压hadoop。**
3. **解压完进入解压后的hadoop目录下，cd etc/h\*，通过vim更改hadoop-env.sh中的export JAVA\_HOME=java下default的路径。**



1. **通过vim更改/etc/profile，在文件底部加一句：export PATH=$PATH:/usr/local/hadoop/bin:/usr/local/hadoop/sbin，保存后执行source /etc/profile命令，使得在任何目录下敲hadoop命令都能成功。敲hadoop命令看是否成功。**
2. **修改主机名hostnamectl set-hostname defaultname.**
3. **将ip与主机名对应**
4. **完整克隆虚拟机master，克隆三个，分别为slave1,slave2,slave3，克隆完后修改IP地址（注意：要设置每个系统的ipv4地址，设置时只保留type、ip、子网掩码（目前是这样，否则会导致ip不对，初始化namenode时导致namenode与datanode无法通信））。**

**设置方法：**



**设置完后，测试本地电脑能与linux系统相互ping通即可。**

1. **用xshell将虚拟机全部启动（xshell中的工具下有个发送输入键到所以会话，勾选），输入systemctl stop firewalld systemctl disable firewalld 彻底关闭防火墙（重启后也不必重新关闭）。**
2. **Vim修改/usr/local/hadoop/etc/hadoop/下的core-site.xml**

**<configuration>中添加：<property>**

**<name>fs.defaultFS</name>**

**<value>hdfs://master:9000</value>**

**</property>**

1. **启动hadoop**

**关闭master的发送键到所有会话，先对master进行格式化：hdfs namenode -format（最好只初始化一次，否则可能出现datanode启动后自动关闭的情况）,**

**之后启动master的hadoop：hadoop-daemon.sh start namenode**

**输入jps，如果出现NomeNode就说明启动成功。**

**然后启动slave1,slave2,slave3的datanode,输入hadoop-daemon.sh start datanode**

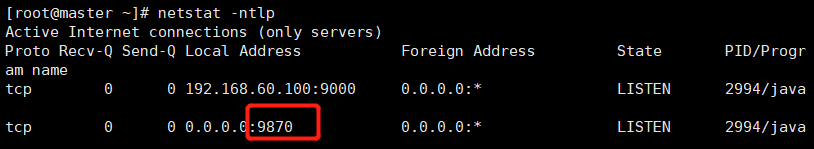
**输入jps,如果出现DataNode就说明启动成功。**

1. **查看集群情况**

**分屏查看namenode情况：****hdfs dfsadmin -report | more**

**通过web端查看namenode情况：**

**netstat -ntlp查看使用的端口**



**在浏览器输入**[**http://192.168.60.100:9870**](http://192.168.60.100:9870)**进行访问**

1. **在master主机上启动集群配置：**

**取消远程登录时输入密码：**

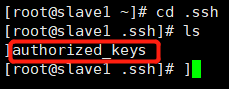
**切换到.ssh目录下：cd .ssh(若无此文件，先用ssh命令进行登陆)**

**ssh-keygen -t rsa 一直回车**

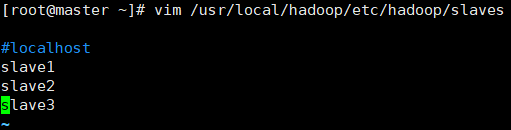
**生成id\_rsa(root用户的私钥)和id\_rsa\_pub(公钥)**

**将公钥拷贝到slave机器上：**

**ssh-copy-id slave1（主机名或ip地址都可以，也需要给master主机copy一份）**

**对应slave机器上有此文件即可，可用ssh slave登录进行测试，看是否还需要密码。**

**配置如下文件：**



**在master上利用start-dfs.sh启动集群。stop-dfs.sh关闭集群。**

1. **操作文件**

**hadoop fs -ls / 查看文件**

**hadoop fs -text /test.data 查看文件内容**

**hadoop fs -put 文件名 / 将某个文件放到hadoop服务器**

**hadoop fs -rm -r -f /hadoop 删除文件夹**

1. **伪分布（部署到同一台物理机上，对java访问来说情况是一样的）**
2. **防止重启服务器hadoop服务出现问题，配置core-site.xml**

**<property>**

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

**<value>/var/hadoop</value>**

**</property>**

1. **Java编写hdfs程序**

**在hdfs-site.xml加入如下代码：（测试时先关闭权限，防止操作文件时出现权限问题）**

**<property>**

**<name>dfs.permissions</name>**

**<value>false</value>**

**</property>**

**导jar包：**

**hadoop/share/hadoop/common 下的common包**

**hadoop/share/hadoop/common/lib下的所有包**

**hadoop/share/hadoop/hdfs下的hdfs包**

**基本操作：**

//访问hadoop的文件  
URL.*setURLStreamHandlerFactory*(new FsUrlStreamHandlerFactory()); //设置以hdfs的形式访问  
URL url = new URL("hdfs://192.168.60.101:9000/helloHadoop.txt");  
InputStream in = url.openStream();  
IOUtils.*copyBytes*(in, System.*out*, 4096, true);

//操作文件  
Configuration conf = new Configuration();  
conf.set("fs.defaultFS", "hdfs://192.168.60.101:9000");  
FileSystem fileSystem = FileSystem.*get*(conf);  
//判断目录是否存在  
System.*out*.println(fileSystem.exists(new Path("/test")));  
//创建目录/文件  
System.*out*.println(fileSystem.mkdirs(new Path("/test")));  
//删除目录/文件  
System.*out*.println(fileSystem.delete(new Path("/test"),true));

//将win上的文件写到Hadoop上  
FSDataOutputStream out = fileSystem.create(new Path("/test.data"),true);  
FileInputStream in = new FileInputStream("F:/BOOTEX.LOG");  
IOUtils.*copyBytes*(in, out, 4096, true);

//获取hadoop下整个目录的信息  
FileStatus[] statuses = fileSystem.listStatus(new Path("/"));  
for (FileStatus status : statuses) {  
 System.*out*.println(status.getPath()); //路径  
 System.*out*.println(status.getPermission()); //权限  
 System.*out*.println(status.getReplication()); //备份几份  
}

**HDFS架构（相当于大硬盘）**