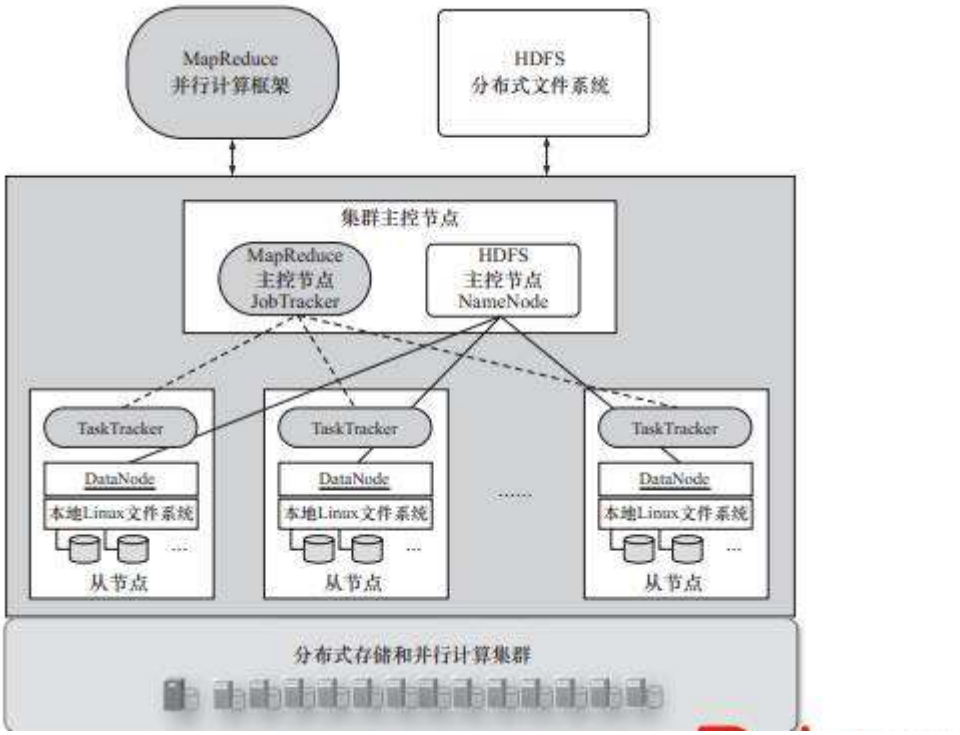
目录

1 集群部署介绍

1.1 Hadoop 简介

Hadoop 是 Apache 软件基金会旗下的一个开源分布式计算平台。以 Hadoop 分布式文件系统（HDFS，Hadoop Distributed Filesystem）和 MapReduce（Google MapReduce 的开源实现）为核心的 Hadoop 为用户提供了系统底层细节透明的分布式基础架构。Hadoop 系统分布式存储与并行计算构架如图所示：[3]



对于 Hadoop 的集群来讲，可以分成两大类角色：Master 和 Salve。一个 HDFS 集群是由一个 NameNode 和若干个 DataNode 组成的。其中 NameNode 作为主服务器，管理文件系统的命名空间和客户端对文件系统的访问操作；集群中的 DataNode 管理存储的数据。MapReduce 框架是由一个单独运行在主节点上的 JobTracker 和运行在每个集群从节点的 TaskTracker 共同组成的。主节点负责调度构成一个作业的所有任务，这些任务分布在不同的从节点上。主节点监控它们的执行情况，并且重新执行之前的失败任务；从节点仅负责由主节点指派的任务。当一个 Job 被提交时，JobTracker 接收到提交作业和配置信息之后，就会将配置信息等分发给从节点，同时调度任务并监控 TaskTracker 的执行。

从上面的介绍可以看出，HDFS 和 MapReduce 共同组成了 Hadoop 分布式系统体系结构的核心。HDFS 在集群上实现分布式文件系统，MapReduce 在集群上实现了分布式计算和任务处理。HDFS 在 MapReduce 任务处理过程中提供了文件操作和存储等支持，MapReduce 在 HDFS 的基础上实现了任务的分发、跟踪、执行等工作，并收集结果，二者相互作用，完成了 Hadoop 分布式集群的主要任务。[4]

1.2 环境说明

集群中包括 3 个节点：1 个 Master，2 个 Salve，节点之间局域网连接，可以相互 ping 通。节点 IP 地址分布如下：

机器名称	IP 地址
Master.Hadoop	192.168.1.2
Slave1.Hadoop	192.168.1.3
Slave2.Hadoop	192.168.1.4

3 个节点上均是 CentOS 6.5 系统，并且有一个相同的用户 jjabc。Master 机器主要配置 NameNode 和 JobTracker 的角色，负责总管分布式数据和分解任务的执行；2 个 Salve 机器配置 DataNode 和 TaskTracker 的角色，负责分布式数据存储以及任务的执行。其实应该还应该有 1 个 Master 机器，用来作为备用。假设 Master 服务器宕机，还有一个备用可以马上启用。后续经验积累一定阶段后补上一台备用 Master 机器。

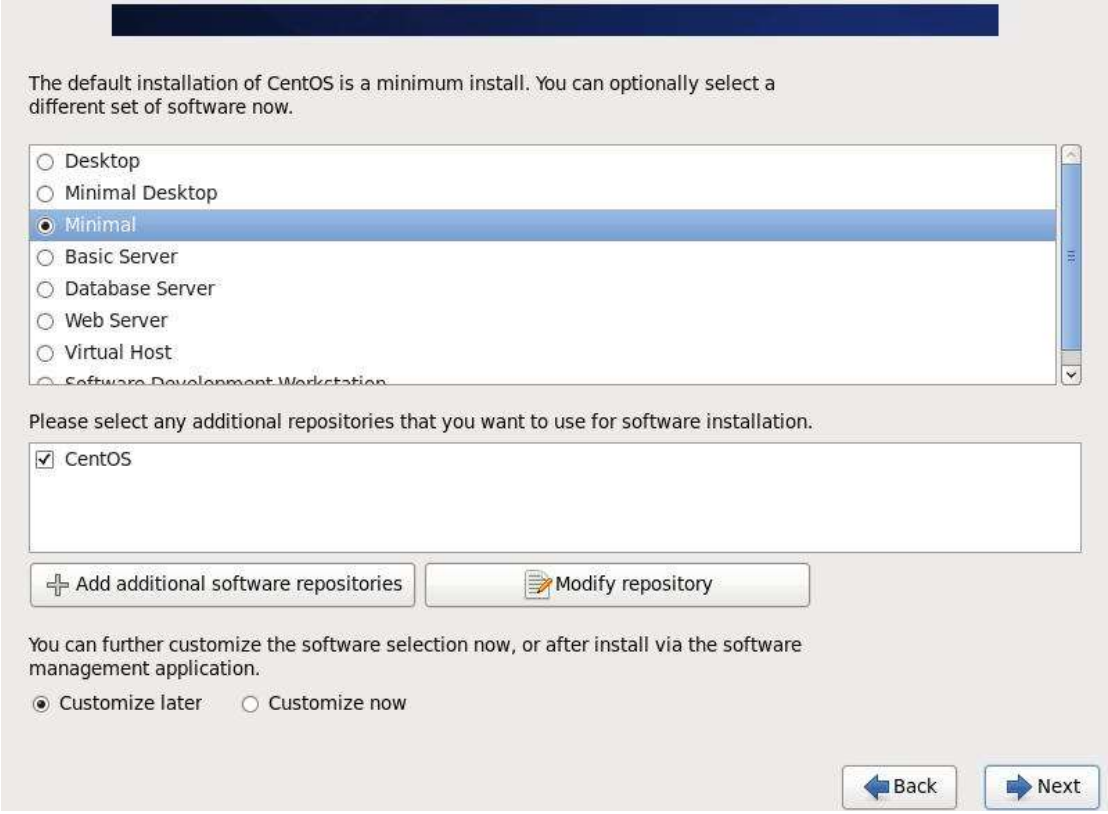
1.3 安装 CentOS 虚拟机

我的实验环境如下表所示：

操作系统	Windows 10, 64-bit
虚拟机软件	VMware Workstation 11.1.0
光盘镜像	CentOS-6.5-x86_64-bin-DVD1.iso

CentOS 6.5 的安装镜像文件有两个 DVD，安装系统只用到第一个镜像文件，即 DVD1 另外一个镜像文件是附带的软件包。[5]

我在安装虚拟机时采用了“最小化安装”，如图所示：



1.4 网络配置

下面的例子我们将以 Master 机器为例（它的主机名为“Master.Hadoop”，IP 为“192.168.1.2”）进行一些主机名配置的相关操作。其他的 Slave 机器以此为参照进行修改。

1.4.1 修改当前机器名称

执行以下命令查看主机名，如果跟规划的不一致，要进行修改。

```
hostname
```

hostname 为“localhost.localdomain”，如图所示：

因使用“hostname”命令查得的“Master”的主机名不为“Master.Hadoop”，与我们预先规划的不一致，故要修改主机名。

当主机名与我们的预先规划不一致，则应将“/etc/sysconfig/network” 文件中的“HOSTNAME”的值改成我们预先规划的名称。

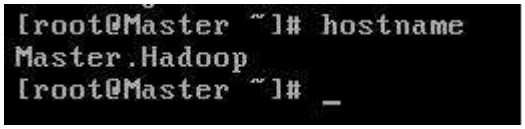
用下面命令进行修改当前机器的主机名。（注：修改系统文件一般用 root 用户）

```
vi /etc/sysconfig/network
```

使用 vi 编辑器编辑“/etc/sysconfig/network”文件，如图所示：

修改“HOSTNAME”为“Master.Hadoop”，如图所示：

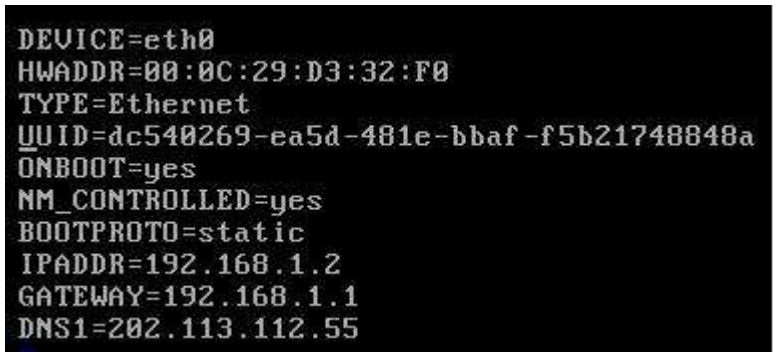
重启系统后可以发现，主机名变成了“Master.Hadoop”,如图所示：



1.4.2 修改当前机器 IP

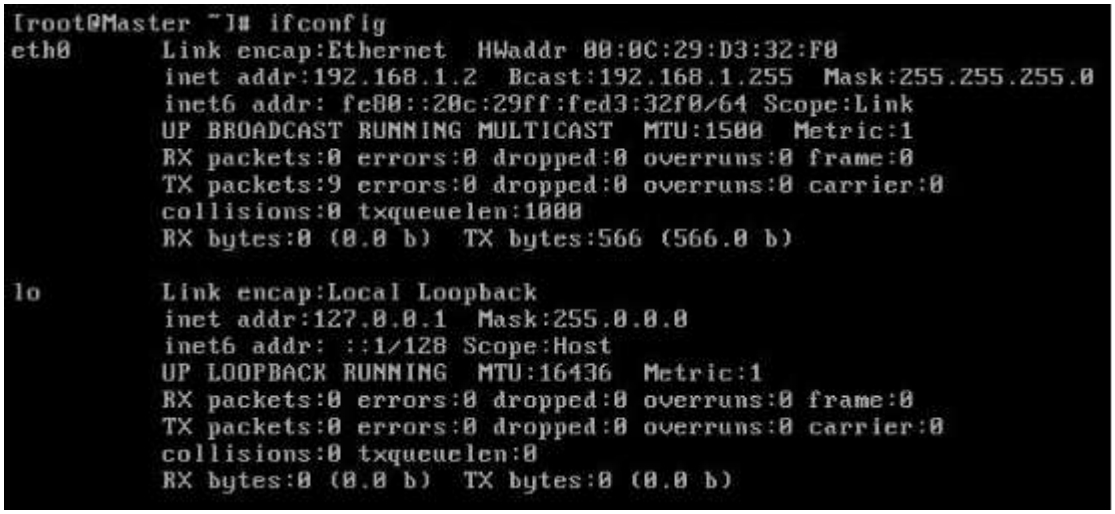
使用 vi 编辑器查看“/etc/sysconfig/network-scripts/ifcfg-eth0”内容，如果 IP 与原规划不符，则执行修改。如图所示：

修改结果如图所示：



在修改后重启网络服务，如图所示：

执行“ifconfig”命令查看 IP，如图所示：



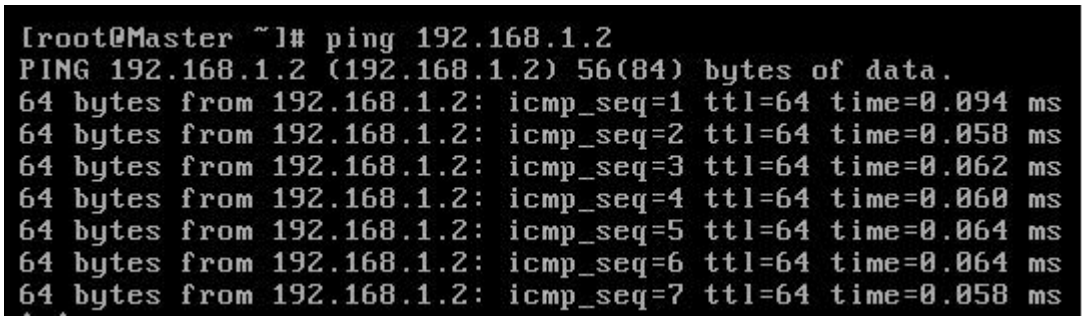
注：Slave1、Slave2 的配置方法与 Master 相同。

1.4.3 配置 hosts 文件

“/etc/hosts”这个文件是用来配置主机将用的 DNS 服务器信息，是记载 LAN 内接续的各主机的对应[HostName 和 IP]用的。当用户在进行网络连接时，首先查找该文件，寻找对应主机名（或域名）对应的 IP 地址。

我们要测试两台机器之间知否连通，一般用"ping 机器的 IP"，如果想用"ping 机器的主机名"发现找不到该名称的机器，解决的办法就是修改"/etc/hosts"这个文件，通过把 LAN 内的各主机的 IP 地址和 HostName 的一一对应写入这个文件的时候，就可以解决问题。

例如，机器“Master.Hadoop”的 IP 地址为“192.168.1.2”。执行命令“ping”进行连接测试。测试结果如图所示：



直接对 IP 地址进行测试，能够 ping 通，但是对主机名进行测试，发现不能 ping 通，提示“unknown host”，这时查看"Master.Hadoop"的"/etc/hosts"文件内容。

发现里面没有“ 192.168.1.2 Master.Hadoop”这一内容，故本机器无法对机器的主机名为 “Master.Hadoop”解析。

在进行 Hadoop 集群配置中，需要在“/etc/hosts”文件中添加集群中所有机器的 IP 与主机名，这样 Master 与所有的 Slave 机器之间不仅可以通过 IP 进行通信，而且还可以通过主机名进行通信。所以在所有的机器上的"/etc/hosts"文件末尾中都要添加如下内容：

192.168.1.2 Master.Hadoop

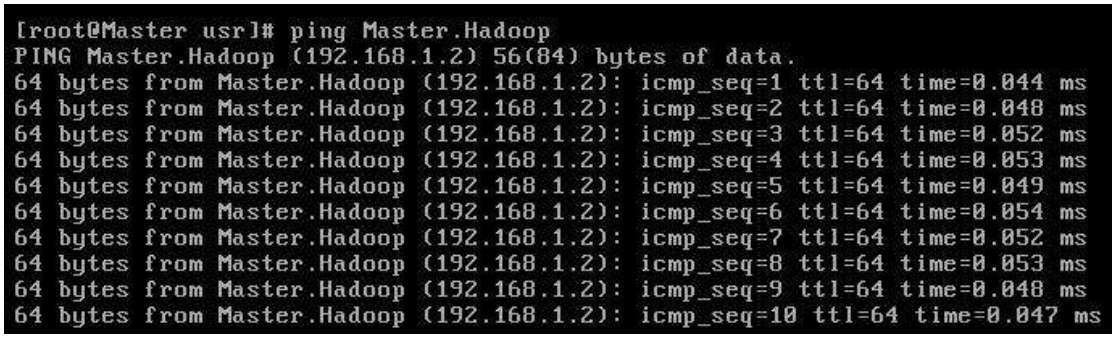
192.168.1.3 Slave1.Hadoop

192.168.1.4 Slave2.Hadoop

使用 vi 编辑器编辑“/etc/hosts”文件，如图所示：

添加结果如图所示：

执行命令“ping Master.Hadoop”的结果如图所示：



由图可知，我们已经能用主机名进行 ping 通了，说明我们刚才添加的内容，在局域网内能进行 DNS 解析了，那么现在剩下的事儿就是在其余的 Slave 机器上进行相同的配置。然后进行测试。（备注：当设置 SSH 无密码验证后，可以“scp”进行复制，然后对原来的“hosts”文件执行覆盖即可。）

至此，3 台机器可以通过主机名互相 ping 通。

1.5 所需软件

1.5.1 JDK 软件

软件版本：jdk-7u25-linux-x64.tar.gz

下载地址：http://vdisk.weibo.com/s/BJD8pmDP8CUsN

1.5.2 Hadoop 软件

软件版本：hadoop1.0.0.tar.gz

下载地址：http://download.csdn.net/detail/link200809/5585821

2 SSH 无密码验证设置

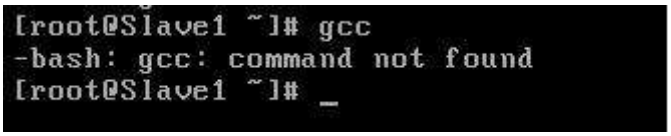
Hadoop 运行过程中需要管理远端 Hadoop 守护进程，在 Hadoop 启动以后，NameNode 是通过 SSH （ Secure Shell）来启动和停止各个 DataNode 上的各种守护进程的。这就必须在节点之间执行指令的时候是不需要输入密码的形式，故我们需要配置 SSH 运用无密码公钥认证的形式，这样 NameNode 使用 SSH 无密码登录并启动 DataName 进程，同样原理， DataNode 上也能使用 SSH 无密码登录到 NameNode。

2.1 安装和启动 SSH 协议

2.1.1 安装 gcc

因 CentOS 虚拟机采用了最小化安装，故尚未安装 ssh 和 gcc。因 ssh 的安装依赖于 gcc，所以首先进行 gcc 的安装。

执行命令“gcc”验证 gcc 是否已经安装好，如图所示：

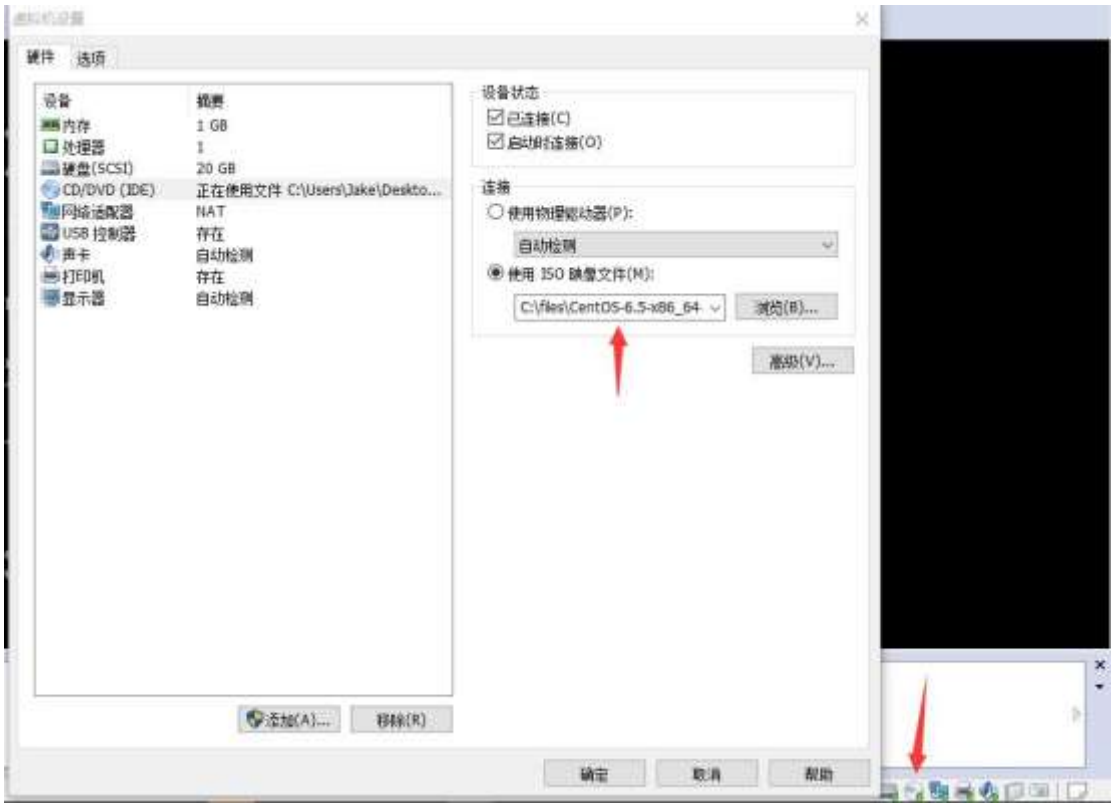


因系统提示“command not found”，故应该安装 gcc。现修改 yum 的更新源，如图所示：

文件“/etc/yum.repos.d/CentOS-Media.repo”的内容如图所示：

删除“file:///media/CentOS/”和 file:///media/cdrom 两行，如图所示：

设置虚拟机的光驱的镜像文件为“CentOS-6.5-x86_64-bin-DVD1.iso”，并将 cdrom 设备挂载到“media”文件夹，如图所示：




```
[root@Slave1 ~]# mount /dev/cdrom /media
mount: block device /dev/sr0 is write-protected, mounting read-only
```

进行 gcc 的安装，如图所示：

```
[root@Slave1 ~]# rpm --import /etc/pki/rpm-gpg/RPM-GPG-KEY-CentOS-6
root@Slave1 ~]# yum --disablerepo=\* --enablerepo=c6-media install gcc_
```

在安装后执行“gcc”命令时，系统不再提示“command not found”，如图所示：

```
Complete!
[root@Slave1 ~]# gcc
gcc: no input files
[root@Slave1 ~]# _
```

至此，“gcc”已经安装成功。

2.1.2 安装 ssh

在 media 文件夹中查找名称中包含“libedit”的文件，如图所示：

```
[root@Master ~]# find /media | grep libedit
/media/Packages/libedit-2.11-4.20080712cvs.1.el6.x86_64.rpm
[root@Master ~]# _
```

执行“rpm”命令安装“libedit”，如图所示：

```
[root@Master ~]# rpm -ivh /media/Packages/libedit-2.11-4.20080712cvs.1.el6.x86_6
4.rpm _
```

使用同样的方式，安装“openssh-clients”和“openssh-server”，如图所示：

```
[root@Master ~]# rpm -ivh /media/Packages/openssh-clients-5.3p1-94.el6.x86_64.rp
m _
```

```
[root@Master ~]# rpm -ivh /media/Packages/openssh-server-5.3p1-94.el6.x86_64.rpm
 _
```

在安装后执行“ssh”命令时，系统不再提示“command not found”，如图所示：

```
[root@Master ~]# ssh
usage: ssh [-1246AaCfgKkMNnqsTtVvXxYy] [-b bind_address] [-c cipher_spec]
           [-D [bind_address:]port] [-e escape_char] [-F configfile]
           [-I pkcs11] [-i identity_file]
           [-L [bind_address:]port:host:hostport]
           [-l login_name] [-m mac_spec] [-O ctl_cmd] [-o option] [-p port]
           [-R [bind_address:]port:host:hostport] [-S ctl_path]
           [-W host:port] [-w local_tun[:remote_tun]]
           [user@]hostname [command]
[root@Master ~]# _
```

至此，ssh 已经安装完毕。（要为集群的每台机器都安装 ssh。）

2.2 引入 RSA 公钥认证，实现无密码登录

Hadoop 运行过程中需要管理远端 Hadoop 守护进程，在 Hadoop 启动以后，NameNode 是通过 SSH （ Secure Shell）来启动和停止各个 DataNode 上的各种守护进程的。这就必须在节点之间执行指令的时候是不需要输入密码的形式，故我们需要配置 SSH 运用无密码公钥认证的形式，这样 NameNode 使用 SSH 无密码登录并启动 DataName 进程，同样原理， DataNode 上也能使用 SSH 无密码登录到 NameNode。

在 Master、Slave1、Slave2 结点均创建用户“jjabc”（密码为“654321”），如图所示：

```
[root@Master ~]# useradd jjabc
[root@Master ~]# passwd jjabc
Changing password for user jjabc.
New password:
BAD PASSWORD: it is too simplistic/systematic
BAD PASSWORD: is too simple
Retype new password:
passwd: all authentication tokens updated successfully.
[root@Master ~]# _
```

切换到“jjabc”用户，如图所示：

```
[root@Master ~]# su jjabc
```

生成秘钥对，如图所示：

```
[jjabc@Master root]$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/jjabc/.ssh/id_rsa):
Created directory '/home/jjabc/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/jjabc/.ssh/id_rsa.
Your public key has been saved in /home/jjabc/.ssh/id_rsa.pub.
The key fingerprint is:
03:41:a8:c5:9a:39:cf:a0:cc:93:17:85:3b:72:44:50 jjabc@Master.Hadoop
The key's randomart image is:
+--[ RSA 2048]----+
|  .+E.oo         |
|    o+. .        |
|   .*o .         |
|  .B=    .       |
| o.+=o   S       |
| .= .o    .      |
|  o              |
|                 |
|                 |
+-----------------+
```

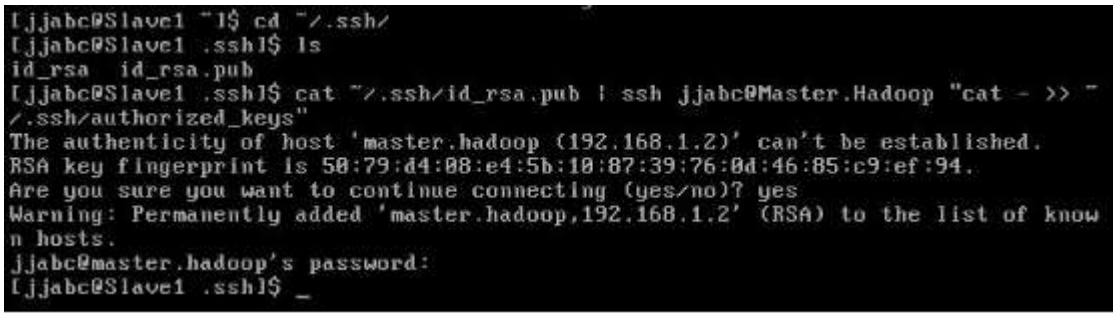
查看秘钥对（公钥和私钥），如图所示：

```
[jjabc@Master root]$ ls ~/.ssh
id_rsa  id_rsa.pub
```

到Slave1机器的“jjabc”用户目录下产生密钥对。借助ssh把jjabc@Slave1的公钥“~/.ssh/id_rsa.pub”的内容追加到jjabc@Master的“~/.ssh/authorized_keys”中，如图所示：

```
[jjabc@Slave1 root]$ ssh-keygen -t rsa_
```

```
Enter file in which to save the key (/home/jjabc/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/jjabc/.ssh/id_rsa.
Your public key has been saved in /home/jjabc/.ssh/id_rsa.pub.
The key fingerprint is:
13:e8:41:96:0d:a9:45:5c:39:62:70:01:22:e3:d6:6b jjabc@Slave1.Hadoop
The key's randomart image is:
+--[ RSA 2048]----+
|o  . o=BB..      |
|.o.. +B.+        |
| o  . +o...      |
|.    o. . .      |
|    E  . S       |
|   .     .       |
|                 |
|                 |
|                 |
+-----------------+
```



注：也在 Slave2 机器执行与 Slave1 机器的相同的操作，将 jjabc@Slave1 的公钥追加到 jjabc@Master 的“~/.ssh/authorized_keys”中。[6]

此时，authorized_keys 的内容如图所示：

此时，再将 jjabc 的“id_rsa.pub”的文件内容追加到“authorized_keys”中，如图所示：

修改“authorized_keys”文件的权限为“600”，如图所示：

把“jjabc@Master”的“authorized_keys”文件传输给“jjabc@Slave1”和“jjabc@Slave2”，这样 Master、Slave1 和 Slave2 共同拥有了“authorized_keys”，如图所示：

```
[jjabc@Master .ssh]$ scp ~/.ssh/authorized_keys jjabc@Slave1.Hadoop:~/.ssh/_
```

```
[jjabc@Master .ssh]$ scp ~/.ssh/authorized_keys jjabc@Slave2.Hadoop:~/.ssh/
```

```
[jjabc@Master .ssh]$ scp ~/.ssh/authorized_keys jjabc@Slave1.Hadoop:~/.ssh/
jjabc@slave1.hadoop's password:
authorized_keys                               100% 1203     1.2KB/s   00:00
```

在 Master、Slave1、Slave2 结点中，均登录 root 用户，通过 vi 编辑器编辑“/etc/ssh/sshd_config”文件，去掉“RSAAuthentication yes”、“AuthorizedKeysFile .ssh/authorized_keys”和“PubkeyAuthentication yes”和这三行的“#”注释，如图所示：

```
# vi /etc/ssh/sshd_config
```

```
#MaxSessions 10

RSAAuthentication yes
PubkeyAuthentication yes
AuthorizedKeysFile      .ssh/authorized_keys
#AuthorizedKeysCommand none
#AuthorizedKeysCommandRunAs nobody

# For this to work you will also need host keys in
#RhostsRSAAuthentication no
```

分别在 Master、Slave1、Slave2 机器中测试连接，如图所示：

```
[root@Master ~]# su jjabc
[jjabc@Master root]$ ssh Slave1.Hadoop
Last login: Fri Feb 26 21:49:40 2016 from master.hadoop
[jjabc@Slave1 ~]$ ^C
[jjabc@Slave1 ~]$ exit
logout
Connection to Slave1.Hadoop closed.
[jjabc@Master root]$ ssh Slave2.Hadoop
Last login: Fri Feb 26 21:47:13 2016 from master.hadoop
[jjabc@Slave2 ~]$ exit
logout
Connection to Slave2.Hadoop closed.
[jjabc@Master root]$ _
```

```
[root@Slave1 ~]# su jjabc
[jjabc@Slave1 root]$ ssh Master.Hadoop
Last login: Fri Feb 26 21:49:34 2016 from slave2.hadoop
[jjabc@Master ~]$ exit
logout
Connection to Master.Hadoop closed.
[jjabc@Slave1 root]$ ssh Slave2.Hadoop
Last login: Sat Feb 27 01:03:36 2016 from master.hadoop
[jjabc@Slave2 ~]$ exit
logout
Connection to Slave2.Hadoop closed.
[jjabc@Slave1 root]$ _
```

```
[jjabc@Slave2 root]$ ssh Master.Hadoop
Last login: Sat Feb 27 01:05:22 2016 from slave1.hadoop
[jjabc@Master ~]$ exit
logout
Connection to Master.Hadoop closed.
[jjabc@Slave2 root]$ ssh Slave1.Hadoop
Last login: Sat Feb 27 01:00:18 2016 from master.hadoop
[jjabc@Slave1 ~]$ exit
logout
Connection to Slave1.Hadoop closed.
[jjabc@Slave2 root]$ _
```

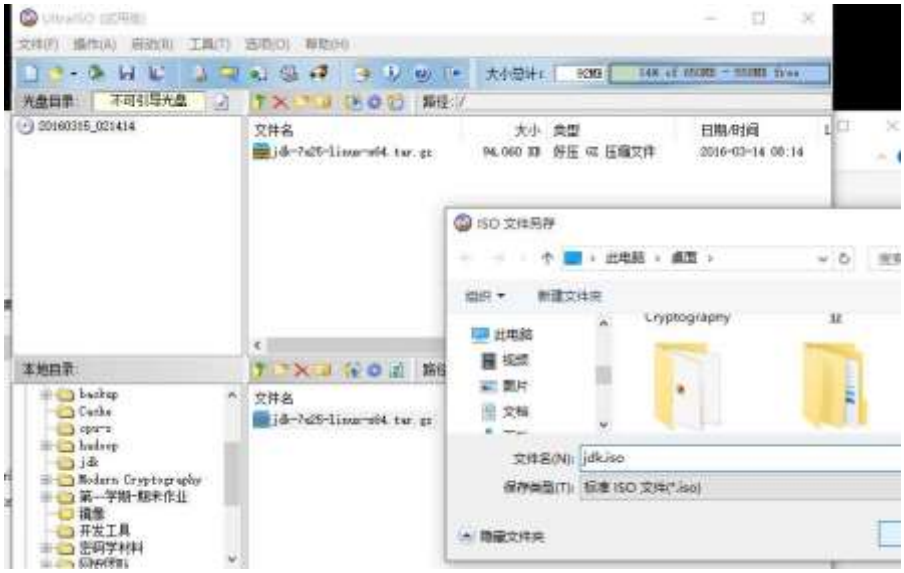
至此，SSH 免密码登陆远程服务器已经安装完毕。

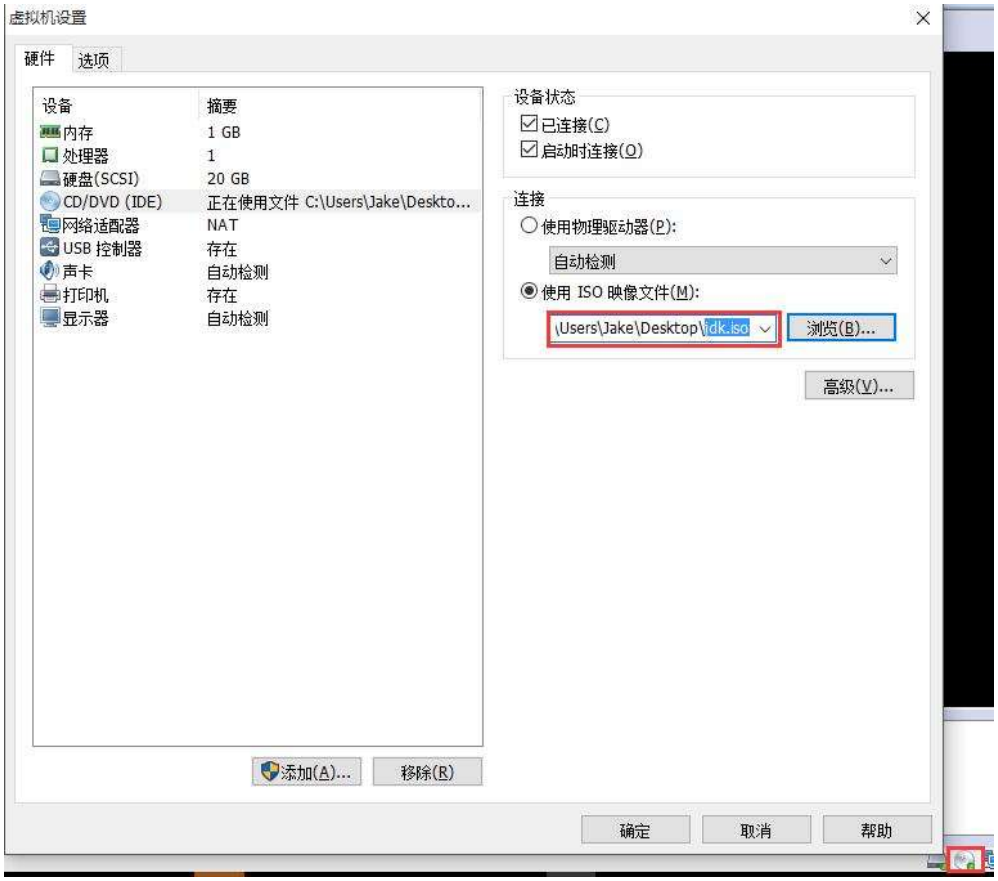
3 Java 环境安装

所有的机器上都应安装 JDK，现在就先在 Master 服务器安装，然后其他服务器按照步骤重复进行即可。安装 JDK 以及配置环境变量，需要以“root”的身份进行。

3.1 安装 JDK

首先将所使用的 JDK——“jdk-7u25-linux-x64.tar.gz”使用 ultraiso 做成 iso 镜像，并将之设置为虚拟机的光驱镜像文件，如图所示：





以 root 身份登录“Master.Hadoop”。接着在“/usr”下创建“java”文件夹。接着挂载 cdrom 设备到“media”文件夹下，如图所示：

```
[root@Master /]# mount /dev/cdrom /media/
mount: block device /dev/sr0 is write-protected, mounting read-only
[root@Master /]# cd /media
[root@Master media]# ls
jdk-7u25-linux-x64.tar.gz
```

将 media 文件夹里的“jdk-7u25-linux-x64.tar.gz”复制到“/usr/java/”文件夹里，如图所示：

```
[root@Master media]# cp jdk-7u25-linux-x64.tar.gz  /usr/java/_
```

进入“/usr/java”目录解压 jdk 的压缩文件，如图所示：

```
[root@Master java]# cd /usr/java
[root@Master java]# ls
jdk-7u25-linux-x64.tar.gz
```

```
[root@Master java]# tar -zxvf jdk-7u25-linux-x64.tar.gz
```

解压结果如图所示：

```
jdk1.7.0_25/man/man1/java.1
jdk1.7.0_25/man/man1/jcmd.1
jdk1.7.0_25/man/man1/xjc.1
jdk1.7.0_25/man/man1/jarsigner.1
jdk1.7.0_25/man/man1/appletviewer.1
jdk1.7.0_25/man/man1/javafxpackager.1
jdk1.7.0_25/man/man1/pack200.1
jdk1.7.0_25/man/man1/keytool.1
jdk1.7.0_25/man/man1/extcheck.1
jdk1.7.0_25/man/man1/jmap.1
jdk1.7.0_25/man/man1/jstatd.1
jdk1.7.0_25/man/man1/javadoc.1
jdk1.7.0_25/THIRDPARTYLICENSEREADME.txt
jdk1.7.0_25/COPYRIGHT
[root@Master java]# _
```

```
[root@Master java]# cd /usr/java
[root@Master java]# ls
jdk1.7.0_25  jdk-7u25-linux-x64.tar.gz
```

3.2 配置环境变量

使用 vi 编辑器编辑文件“/etc/profile”，如图所示：

```
[root@Master /]# vi /etc/profile_
```

在文件“/etc/profile”的末尾配置 Java 环境变量，如图所示：

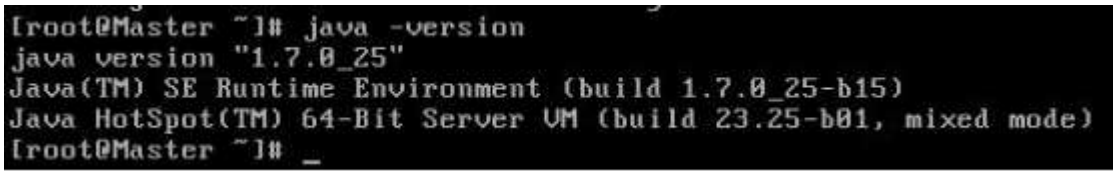
```
unset i
unset -f pathmunge

# set java environment
export JAVA_HOME=/usr/java/jdk1.7.0_25
export CLASSPATH=.:$CLASSPATH:JAVA_HOME/lib:$JAVA_HOME/jre/lib
export PATH=$PATH:$JAVA_HOME/bin:$JAVA_HOME/jre/bin
```

执行命令“source /etc/profile”使环境变量生效，如图所示：

3.3 验证安装成功

执行命令“java -version”验证 Java 环境是否配置成功，如图所示：

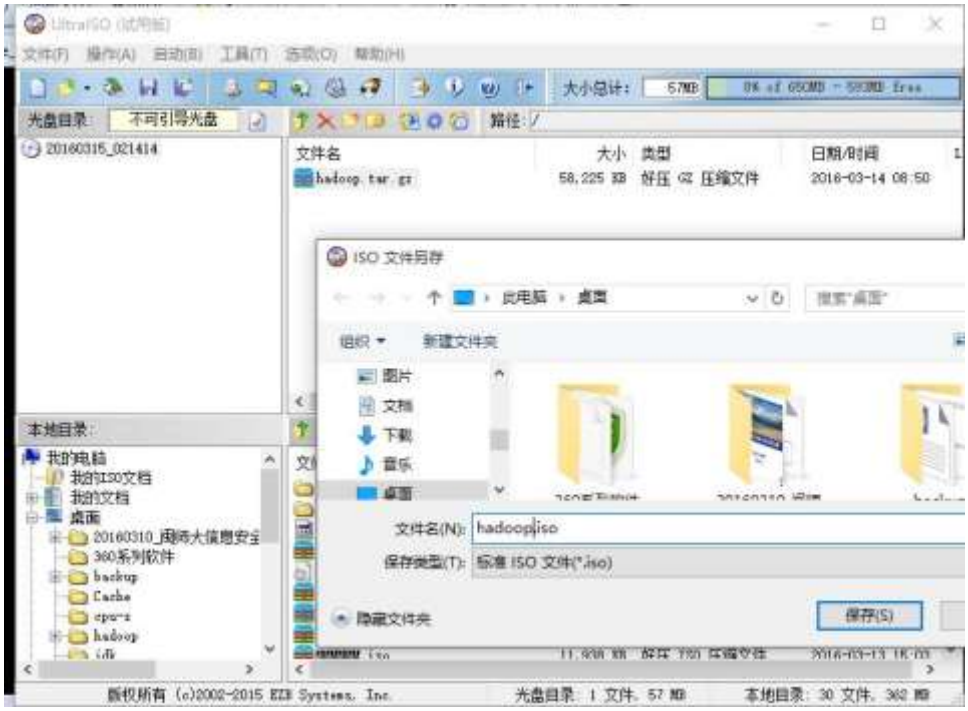


因 Java 的版本信息被打印出来，由此我们可以判断 Master 机器上的 Java 环境变量已经配置成功。应执行同样的操作为 Slava1 和 Slave2 机器配置 Java 环境变量。

4 Hadoop 集群安装

4.1 安装 Hadoop

首先将所使用的“hadoop1.0.0.tar.gz”做成 iso 镜像，并将之设置为虚拟机的光驱镜像文件，如图所示：



接着挂载 cdrom 设备到“media”文件夹，如图所示：

```
mount: block device /dev/sr0 is write-protected, mounting read-only
```

将 hadoop.tar.gz 复制到/usr/文件夹下，如图所示：

```
[root@Master media]# cp hadoop.tar.gz /usr
```

进入“/usr/”目录解压“hadoop.tar.gz”，如图所示：

```
[root@Master usr]# tar -zxvf hadoop.tar.gz _
```

```
[root@Master usr]# ls
bin   games    hadoop.tar.gz  java   lib64    local  share  tmp
etc   hadoop   include        lib    libexec  sbin   src
```

修改 hadoop 文件夹的 owner 为“jjabc.jjabc”，如图所示：

```
[root@Master usr]# chown -R jjabc.jjabc hadoop
```

```
[root@Master usr]# ll
total 68
dr-xr-xr-x.  2 root  root  12288 Feb 26 05:42 bin
drwxr-xr-x.  2 root  root   4096 Sep 23  2011 etc
drwxr-xr-x.  2 root  root   4096 Sep 23  2011 games
drwxrwxrwx. 16 jjabc jjabc  4096 Mar 14 20:27 hadoop
drwxr-xr-x. 32 root  root   4096 Feb 25 23:57 include
drwxr-xr-x.  3 root  root   4096 Mar 14 16:28 java
dr-xr-xr-x. 10 root  root   4096 Feb 25 23:57 lib
dr-xr-xr-x. 25 root  root  12288 Feb 26 05:41 lib64
drwxr-xr-x. 10 root  root   4096 Feb 25 23:57 libexec
drwxr-xr-x. 12 root  root   4096 Feb 25 19:04 local
dr-xr-xr-x.  2 root  root   4096 Feb 25 19:09 sbin
drwxr-xr-x. 61 root  root   4096 Feb 25 19:09 share
drwxr-xr-x.  4 root  root   4096 Feb 25 19:04 src
lrwxrwxrwx.  1 root  root     10 Feb 25 19:04 tmp -> ../var/tmp
[root@Master usr]# _
```

将 hadoop 的环境变量添加到“/etc/profile”文件末尾，如图所示：

```
# set java environment
export JAVA_HOME=/usr/java/jdk1.7.0_25
export CLASSPATH=.:$CLASSPATH:JAVA_HOME/lib:$JAVA_HOME/jre/lib
export PATH=$PATH:$JAVA_HOME/bin:$JAVA_HOME/jre/bin

# set hadoop path
export HADOOP_HOME=/usr/hadoop
export PATH=$PATH:$HADOOP_HOME/bin
"/etc/profile" 88L, 2060C written
```

4.2 配置 Hadoop

4.2.1 配置 hadoop-env.sh

修改配置文件“/usr/hadoop/conf/”，将“JAVA_HOME”添加到该文件的末尾。如图所示：

```
[root@Master hadoop]# cd conf
[root@Master conf]# ls
capacity-scheduler.xml      hadoop-policy.xml      slaves
configuration.xsl           hdfs-site.xml          ssl-client.xml.example
core-site.xml               log4j.properties       ssl-server.xml.example
fair-scheduler.xml          mapred-queue-acls.xml  taskcontroller.cfg
hadoop-env.sh               mapred-site.xml
hadoop-metrics2.properties  masters
[root@Master conf]# vi hadoop-env.sh _
```

```
# set java environment
export JAVA_HOME=/usr/java/jdk1.7.0_25
"hadoop-env.sh" 57L, 2339C written
[root@Master conf]# _
```

4.2.2 配置 core-site.xml

修改配置文件“/usr/hadoop/conf/core-site.xml”,配置 HDFS 的地址和端口号。如图所示：

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>

<!-- Put site-specific property overrides in this file. -->

<configuration>

   <property>
        <name>fs.default.name</name>
        <value>hdfs://192.168.1.2:9000</value>
    </property>

   <property>
        <name>hadoop.tmp.dir</name>
        <value>/usr/hadoop/tmp</value>
        <description>A base for other temporary directories</description>
   </property>
</configuration>
```

4.2.3 配置 hdfs-site.xml

使用 vi 编辑器修改配置文件“/usr/hadoop/conf/hdfs-site.xml”。如图所示：

```
[root@Master conf]# vi hdfs-site.xml _
```

```
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>

<!-- Put site-specific property overrides in this file. -->

<configuration>
<property>

        <name>dfs.replication</name>

        <value>1</value>


    </property>
</configuration>
~
~
~
~
~
~
~
~
"hdfs-site.xml" 15L, 270C written
[root@Master conf]# _
```

4.2.4 配置 mapred-site.xml

使用 vi 编辑器修改“mapred-site.xml”文件，配置 JobTracker 的地址和端口，如图所示：

```
[root@Master conf]# vi mapred-site.xml _
```

```
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>

<!-- Put site-specific property overrides in this file. -->

<configuration>
  <property>

        <name>mapred.job.tracker</name>

        <value>http://192.168.1.2:9001</value>

    </property>
</configuration>
~
~
~
~
~
~
~
~
~
~
"mapred-site.xml" 14L, 296C written
[root@Master conf]# _
```

4.2.5 配置 masters 文件

使用 vi 编辑器配置 masters 文件，如图所示：

```
[root@Master conf]# vi masters _
```

```
[root@Master conf]# more masters
192.168.1.2
```

4.2.6 配置 slaves 文件

使用 vi 编辑器配置 slaves 文件，如图所示：

```
[root@Master conf]# vi slaves _
```

```
192.168.1.3
192.168.1.4
```

现在在 Master 机器上的 Hadoop 配置就结束了，剩下的就是配置 Slave 机器上的 Hadoop。

将 Master 上配置好的 hadoop 所在文件夹"/usr/hadoop"复制到所有的 Slave 的"/usr"目录下（实际上 Slave 机器上的 slaves 文件是不必要的， 但复制了也没问题）。用以下命令格式进行。（备注：此时用户可以为 hadoop 也可以为 root。）如图所示：

```
[root@Master conf]# scp -r /usr/hadoop root@Slave1.Hadoop:/usr/_
```

```
[root@Slave1 ~]# cd /
[root@Slave1 /]# ls
bin   dev  home  lib64       media  opt   root  selinux  srv  usr
boot  etc  lib   lost+found  mnt    proc  sbin  srv      tmp  var
[root@Slave1 /]# cd usr/
[root@Slave1 usr]# ls
bin  games    include  lib    libexec  sbin   src
etc  hadoop   java     lib64  local    share  tmp
[root@Slave1 usr]# ll
total 68
dr-xr-xr-x.  2 root root 12288 Feb 26 18:11 bin
drwxr-xr-x.  2 root root  4096 Sep 23  2011 etc
drwxr-xr-x.  2 root root  4096 Sep 23  2011 games
drwxrwxrwx. 16 root root  4096 Feb 27 22:35 hadoop
drwxr-xr-x. 32 root root  4096 Feb 26 18:10 include
drwxr-xr-x.  3 root root  4096 Feb 27 19:53 java
dr-xr-xr-x. 10 root root  4096 Feb 26 18:10 lib
dr-xr-xr-x. 25 root root 12288 Feb 26 18:11 lib64
drwxr-xr-x. 10 root root  4096 Feb 26 18:10 libexec
drwxr-xr-x. 12 root root  4096 Feb 25 21:03 local
dr-xr-xr-x.  2 root root  4096 Feb 25 21:10 sbin
drwxr-xr-x. 61 root root  4096 Feb 25 21:10 share
drwxr-xr-x.  4 root root  4096 Feb 25 21:03 src
lrwxrwxrwx.  1 root root    10 Feb 25 21:03 tmp -> ../var/tmp
[root@Slave1 usr]# _
```

修改 hadoop 文件夹的 owner 为“jjabc.jjabc”，如图所示：

```
[root@Slave1 usr]# ls
bin  games    include  lib    libexec  sbin   src
etc  hadoop   java     lib64  local    share  tmp
[root@Slave1 usr]# ll
total 68
dr-xr-xr-x.  2 root  root  12288 Feb 26 18:11 bin
drwxr-xr-x.  2 root  root   4096 Sep 23  2011 etc
drwxr-xr-x.  2 root  root   4096 Sep 23  2011 games
drwxrwxrwx. 16 jjabc jjabc  4096 Mar 14 17:33 hadoop
drwxr-xr-x. 32 root  root   4096 Feb 26 18:10 include
drwxr-xr-x.  3 root  root   4096 Mar 14 16:32 java
dr-xr-xr-x. 10 root  root   4096 Feb 26 18:10 lib
dr-xr-xr-x. 25 root  root  12288 Feb 26 18:11 lib64
drwxr-xr-x. 10 root  root   4096 Feb 26 18:10 libexec
drwxr-xr-x. 12 root  root   4096 Feb 25 21:03 local
dr-xr-xr-x.  2 root  root   4096 Feb 25 21:10 sbin
drwxr-xr-x. 61 root  root   4096 Feb 25 21:10 share
drwxr-xr-x.  4 root  root   4096 Feb 25 21:03 src
lrwxrwxrwx.  1 root  root     10 Feb 25 21:03 tmp -> ../var/tmp
[root@Slave1 usr]# _
```

接着在“Slave1.Hadoop”上修改“/etc/profile”文件（配置 java 环境变量的文件），将以下语句添加到末尾，并使其有效（source /etc/profile）：

```
# set hadoop environment
export HADOOP_HOME=/usr/hadoop
export PATH=$PATH :$HADOOP_HOME/bin
```

如图所示：

```
# set hadoop environment
export HADOOP_HOME=/usr/hadoop
export PATH=$PATH:$HADOOP_HOME/bin
"/etc/profile" 87L, 2066C written
```

注:对 Slave2 也执行同样操作。

4.3 启动并验证

4.3.1 关闭防火墙

在 Master、Slave1、Slave2 结点均要切换到 root 用户关闭防火墙，如图所示：

```
[root@Master conf]# service iptables stop
iptables: Setting chains to policy ACCEPT: filter          [  OK  ]
iptables: Flushing firewall rules:                         [  OK  ]
iptables: Unloading modules:                               [  OK  ]
[root@Master conf]# _
```

4.3.2 格式化 HDFS 文件系统

在 Master 机器格式化 namenode 结点，如图所示：

```
[jjabc@Master usr]$ hadoop namenode -format_
```

```
STARTUP_MSG:   build = https://svn.apache.org/repos/asf/hadoop/common/branches/b
ranch-1.0 -r 1214675; compiled by 'hortonfo' on Thu Dec 15 16:36:35 UTC 2011
************************************************************/
Re-format filesystem in /usr/hadoop/tmp/dfs/name ? (Y or N) Y
16/03/15 11:49:34 INFO util.GSet: VM type       = 64-bit
16/03/15 11:49:34 INFO util.GSet: 2% max memory = 19.33375 MB
16/03/15 11:49:34 INFO util.GSet: capacity      = 2^21 = 2097152 entries
16/03/15 11:49:34 INFO util.GSet: recommended=2097152, actual=2097152
16/03/15 11:49:34 INFO namenode.FSNamesystem: fsOwner=jjabc
16/03/15 11:49:34 INFO namenode.FSNamesystem: supergroup=supergroup
16/03/15 11:49:34 INFO namenode.FSNamesystem: isPermissionEnabled=true
16/03/15 11:49:34 INFO namenode.FSNamesystem: dfs.block.invalidate.limit=100
16/03/15 11:49:34 INFO namenode.FSNamesystem: isAccessTokenEnabled=false accessK
eyUpdateInterval=0 min(s), accessTokenLifetime=0 min(s)
16/03/15 11:49:34 INFO namenode.NameNode: Caching file names occuring more than
10 times
16/03/15 11:49:35 INFO common.Storage: Image file of size 111 saved in 0 seconds
.
16/03/15 11:49:35 INFO common.Storage: Storage directory /usr/hadoop/tmp/dfs/nam
e has been successfully formatted.
16/03/15 11:49:35 INFO namenode.NameNode: SHUTDOWN_MSG:
/************************************************************
SHUTDOWN_MSG: Shutting down NameNode at Master.Hadoop/192.168.1.2
************************************************************/
[jjabc@Master root]$ _
```

4.3.3 启动 Hadoop

在 Master 机器执行“start-all.sh”命令，如图所示：

```
[jjabc@Master usr]$ start-all.sh
Warning: $HADOOP_HOME is deprecated.

starting namenode, logging to /usr/hadoop/libexec/../logs/hadoop-jjabc-namenode-
Master.Hadoop.out
192.168.1.4: starting datanode, logging to /usr/hadoop/libexec/../logs/hadoop-jj
abc-datanode-Slave2.Hadoop.out
192.168.1.3: starting datanode, logging to /usr/hadoop/libexec/../logs/hadoop-jj
abc-datanode-Slave1.Hadoop.out
192.168.1.2: starting secondarynamenode, logging to /usr/hadoop/libexec/../logs/
hadoop-jjabc-secondarynamenode-Master.Hadoop.out
starting jobtracker, logging to /usr/hadoop/libexec/../logs/hadoop-jjabc-jobtrac
ker-Master.Hadoop.out
192.168.1.4: starting tasktracker, logging to /usr/hadoop/libexec/../logs/hadoop
-jjabc-tasktracker-Slave2.Hadoop.out
192.168.1.3: starting tasktracker, logging to /usr/hadoop/libexec/../logs/hadoop
-jjabc-tasktracker-Slave1.Hadoop.out
[jjabc@Master usr]$ _
```

可以通过以上启动日志看出，首先启动 namenode 接着启动 datanode1、datanode2，…然后启动 secondarynamenode。再启动 jobtracker，然后启动 tasktracker1、tasktracker2，…。

4.3.4 验证 Hadoop

（1）验证方法一：使用“jps”命令

在 Master 上执行“jps”命令查看进程，如图所示：

```
[jjabc@Master root]$ jps
3423 Jps
2300 JobTracker
2217 SecondaryNameNode
3258 SecondaryNameNode
3104 NameNode
[jjabc@Master root]$ _
```

在 Slave1 上执行“jps”命令查看进程，如图所示：

```
[jjabc@Slave1 root]$ jps
1821 DataNode
1578 TaskTracker
1903 TaskTracker
1946 Jps
[jjabc@Slave1 root]$ _
```

在 Slave2 上执行“jps”命令查看进程，如图所示：

```
[jjabc@Slave2 root]$ jps
1854 DataNode
1611 TaskTracker
1936 TaskTracker
1989 Jps
[jjabc@Slave2 root]$ _
```

（2）验证方法二：使用“hadoop dfsadmin -report”命令查看 Hadoop 集群的状态，如图所示：

注：如果发现 datanode 没有起来，可能要在 Master、Slave1、Slave2 结点执行这 3 步操作。

1）先删除"/usr/hadoop/tmp"

```
rm -rf /usr/hadoop/tmp
```

2）创建"/usr/hadoop/tmp"文件夹

```
mkdir /usr/hadoop/tmp
```

3）删除"/tmp"下以"hadoop"开头文件

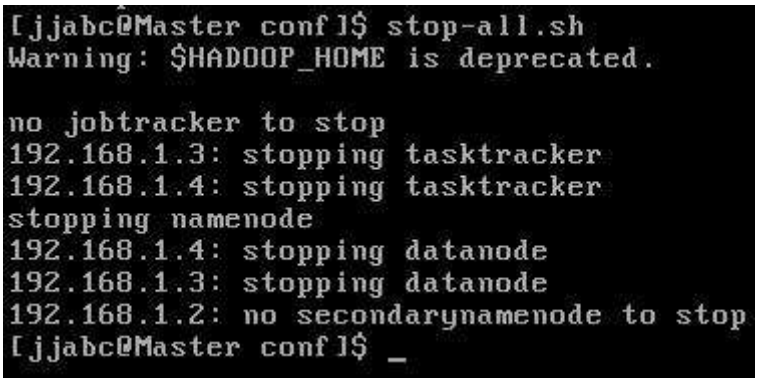
```
rm -rf /tmp/hadoop*
```

最后在 Master 结点执行以下命令

```
hadoop namenode -format
```

4.3.5 关闭 Hadoop

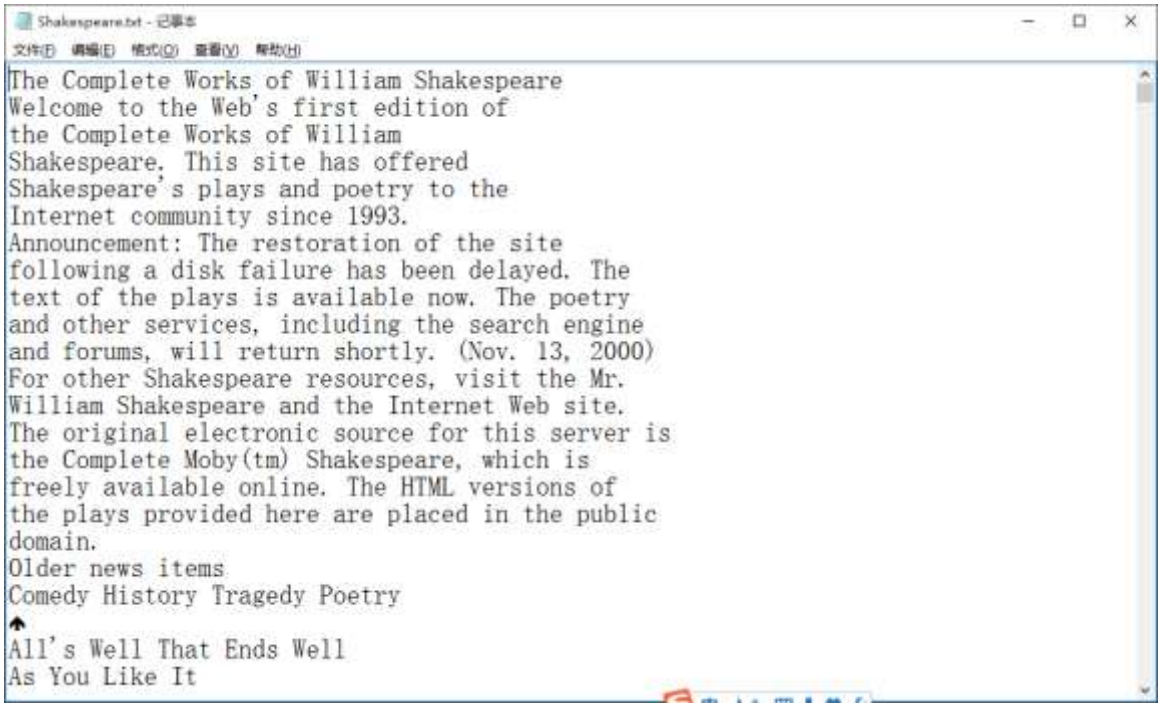
执行“stop-all.sh”命令进行关闭，如图所示：



5 实验部分

5.1 准备工作

利用搭建好的 Hadoop 环境，使用其中的统计词频的程序 wordcount，利用程序统计莎士比亚文章“Shakespheare.txt”的词频。该文件的大小为 9.70 MB。该文件的内容如图所示：



5.1.1 创建本地示例文件

首先在“/home/jjabc”目录下创建文件夹“file”。

```
[jjabc@Master /]$ mkdir /home/jjabc/file
```

执行“cp”命令，将“Shakespheare.txt”文件放入/home/jjabc/file 目录下，如图所示：

```
[root@Master media]# ls
Shakespeare.txt
[root@Master media]# cp Shakespeare.txt  /home/jjabc/file_
```

```
[root@Master media]# cd  /home/jjabc/file
[root@Master file]# ls
file1.txt  file2.txt  Shakespeare.txt
```

将“Shakespheare.txt”的 owner 设置为 jjabc.jjabc，如图所示：

```
[root@Master file]# ll
total 9948
-rw-rw-r--. 1 jjabc jjabc       12 Mar 15 00:40 file1.txt
-rw-rw-r--. 1 jjabc jjabc       13 Mar 15 00:41 file2.txt
-r-xr-xr-x. 1 jjabc jjabc 10175778 Mar 15 12:27 Shakespeare.txt
[root@Master file]# _
```

5.1.2 在 HDFS 上创建输入文件夹

在 HDFS 上创建输入文件夹“input”，如图所示：

```
[root@Master file]# su jjabc
[jjabc@Master file]$ hadoop fs -mkdir input
Warning: $HADOOP_HOME is deprecated.

[jjabc@Master file]$ hadoop fs -ls
Warning: $HADOOP_HOME is deprecated.

Found 1 items
drwxr-xr-x   - jjabc supergroup          0 2016-03-15 12:29 /user/jjabc/input
[jjabc@Master file]$ _
```

5.1.3 上传本地 file 中文件到集群的 input 目录下

上传本地 file 中的“Shakespeare.txt”到集群的 input 目录下,如图所示：

```
[jjabc@Master file]$ hadoop fs -put ~/file/Shakespeare.txt input_
```

```
[jjabc@Master file]$ hadoop fs -ls input
Warning: $HADOOP_HOME is deprecated.

Found 1 items
-rw-r--r--   1 jjabc supergroup   10175778 2016-03-15 12:31 /user/jjabc/input/Sh
akespeare.txt
[jjabc@Master file]$ _
```

5.2 运行例子

5.2.1 在集群上运行 WordCount 程序

在集群上运行 WordCount 程序，如图所示：

```
[jjabc@Master file]$ hadoop jar /usr/hadoop/hadoop-examples-1.0.0.jar wordcount
input ouput_
```

5.2.2 MapReduce 执行过程显示信息

MapReduce 执行过程显示信息，如图所示：

```
[jjabc@Master file]$ hadoop jar /usr/hadoop/hadoop-examples-1.0.0.jar wordcount
input output
Warning: $HADOOP_HOME is deprecated.

16/03/15 12:37:56 INFO input.FileInputFormat: Total input paths to process : 1
16/03/15 12:37:56 INFO mapred.JobClient: Running job: job_201603151155_0002
16/03/15 12:37:57 INFO mapred.JobClient:  map 0% reduce 0%
16/03/15 12:38:21 INFO mapred.JobClient:  map 49% reduce 0%
16/03/15 12:38:24 INFO mapred.JobClient:  map 100% reduce 0%
16/03/15 12:38:48 INFO mapred.JobClient:  map 100% reduce 100%
```

5.3 查看结果

5.3.1 查看 HDFS 上 output 目录内容

生成了三个文件，统计结果在“part-r-00000”文件中，如图所示：

```
[jjabc@Master file]$ hadoop fs -ls output
Warning: $HADOOP_HOME is deprecated.

Found 3 items
-rw-r--r--   1 jjabc supergroup          0 2016-03-15 12:38 /user/jjabc/output/_
SUCCESS
drwxr-xr-x   - jjabc supergroup          0 2016-03-15 12:37 /user/jjabc/output/_
logs
-rw-r--r--   1 jjabc supergroup     707043 2016-03-15 12:38 /user/jjabc/output/p
art-r-00000
[jjabc@Master file]$ _
```

5.3.2 查看结果输出文件内容

单词的词频被统计出来了，如图所示：

```
[jjabc@Master file]$ hadoop fs -cat output/part-r-00000_
```

```
Yare,   2
Yaughan:        1
Ye      49
Ye're   3
Ye?     2
Yea     4
Yea,    232
Yea.    2
Yea;    4
Yead    2
Yearly  4
Yedward;        2
Yerk    2
Yes     2
Yes,    275
Yes.    13
Yes:    3
Yes;    15
Yesterday       2
Yet     527
Yet,    103
Yet,--  4
Yield   14
Yield,  9
Yield:_ 1
```

参考文献

[1]刘赫男,罗霄,高晓东.并行计算的现状与发展[J].煤 2001.

[2]刘鹏.云计算[M].北京:电子工业出版社,2010.7.

[3]黄宜华. 深入理解大数据：大数据处理与编程实践[M].北京:机械工业出版社,2014.8.

[4] pointsand.centos 安装配置 hadoop 超详细过程[EB/OL].[2014-11-21].

http://www.centoscn.com/image-text/install/2014/1121/4158.html

[5] qihang01. CentOS 6.2 最小化 Minimal 安装图解教程[EB/OL].[2012-08-19].

http://www.dedecms.com/knowledge/servers/linux-bsd/2012/0819/8398.html

[6] marsprj.SSH 无密码登录[EB/OL].[2013-02-26].

http://www.cnblogs.com/marsprj/archive/2013/02/26/2933839.html

[7] 虾皮工作室. Hadoop 集群（第 6 期）_WordCount 运行详解[EB/OL]

.[2013-02-26].

http://www.cnblogs.com/marsprj/archive/2013/02/26/2933839.html