实验 3 Hadoop 及相关环境的安装与 WordCount 测试

*实验说明： 开源分布式平台 Hadoop 可以聚合多个计算机形成集群，在各个节点上安装配置完 Hadoop 后可以直接提交分布式代码到集群计算。本次实验可以在个人电脑上用 VMware 完成，或使用天河二号上的 OpenStack 平台创建内存为 2G 的虚拟机资源完成。

实验目的：了解和掌握 Hadoop 的安装及其在各个节点的配置方法，明确各节点（master、slave）的组织方式和工作原理，完成 WordCount 样例的执行，确保配置成功。

实验环境：64 位 ubuntu14.04 虚拟机（master，slave1 [，slave2]），创建好对应连接的 XShell 或者 guacamole，默认虚拟机之间可相互 ping 通，需要的软件如下：

- hadoop-2.6.0.tar.gz
- jdk-8u60-linux-x64.tar.gz

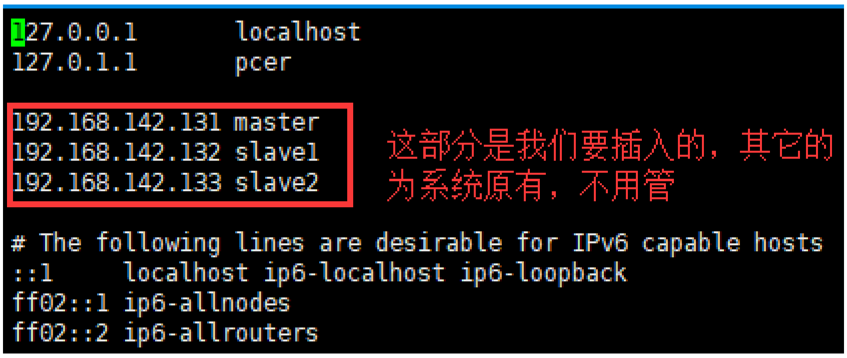
基本实验步骤如下：

1. 修改各个节点的主机名，使其与该节点的角色名一致，如 master,slave1,slave2（这一步非必须，只是为了便于区分）：

```
sudo vi /etc/hostname          #编辑 /etc/hostname 文件从而修改主机名
sudo reboot                    #重启使新主机名生效
```

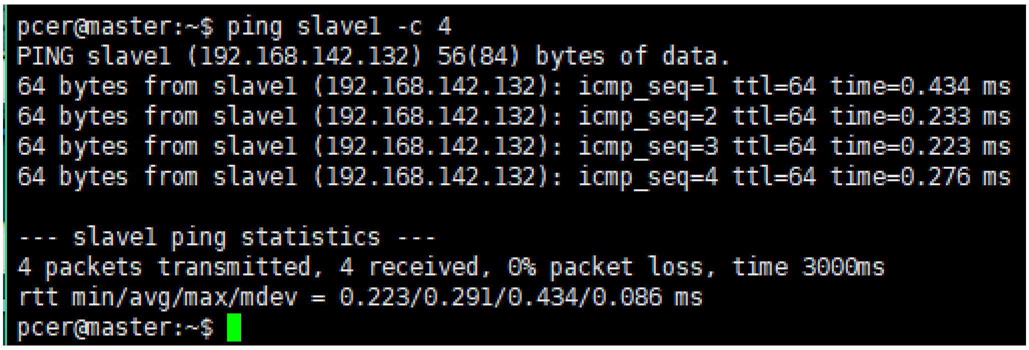
2. 修改各个 hosts 文件，在本地植入部分 DNS 映射，将对应的角色名与 IP 匹配起来，然后尝试用角色名相互 ping，相互能 ping 通证明配置成功：

```
sudo vi /etc/hosts             #编辑 /etc/hosts 文件，插入角色与 IP 映射
ping master -c 4               #尝试用角色名 ping 其它主机，一次 4 个包
```



```
127.0.0.1       localhost
127.0.1.1       pcer

192.168.142.131 master
192.168.142.132 slave1
192.168.142.133 slave2

# The following lines are desirable for IPv6 capable hosts
::1     localhost ip6-localhost ip6-loopback
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
```

这部分是我们要插入的，其它的为系统原有，不用管



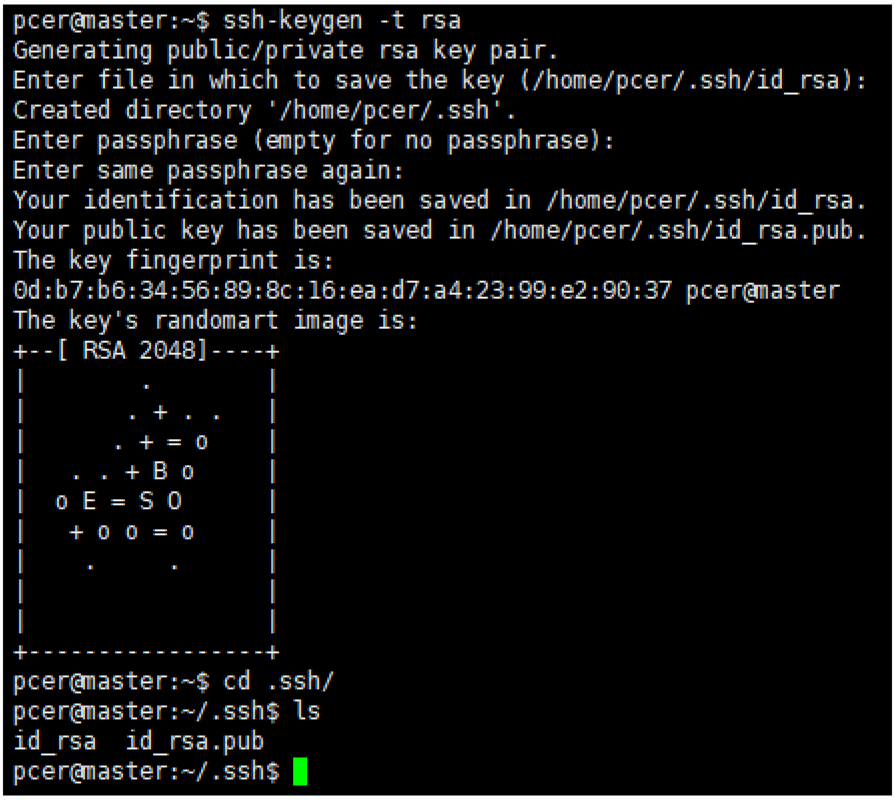
```
pcer@master:~$ ping slave1 -c 4
PING slave1 (192.168.142.132) 56(84) bytes of data.
64 bytes from slave1 (192.168.142.132): icmp_seq=1 ttl=64 time=0.434 ms
64 bytes from slave1 (192.168.142.132): icmp_seq=2 ttl=64 time=0.233 ms
64 bytes from slave1 (192.168.142.132): icmp_seq=3 ttl=64 time=0.223 ms
64 bytes from slave1 (192.168.142.132): icmp_seq=4 ttl=64 time=0.276 ms

--- slave1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3000ms
rtt min/avg/max/mdev = 0.223/0.291/0.434/0.086 ms
pcer@master:~$ 
```

说明：第 2 步保障了 Hadoop 可以通过角色名在局域网里找到各个节点，为了让 Hadoop 可以进一步读取、操作各个节点，需要赋予其登录的权限，意即让 Hadoop 拥有各个节点的普通用户账号，从而在需要操作各个节点时直接用对应的账号登录获取操作权限。SSH 协议可以为节点上的账户创建唯一的公私钥，然后利用这些公私钥实现无密码登录，从而让 Hadoop 直接绕开传统的账号密码登录过程，直接用公私钥访问节点。

3. 配置 SSH 无密码登录

   a) 生成各个节点的 SSH 公私钥：

```
cd ~/.ssh                    # 如果没有该目录，先执行一次 ssh localhost
rm ./id_rsa*                 # 删除之前生成的公匙（如果有）
ssh-keygen -t rsa            # 一直按回车就可以
```



```
pcer@master:~$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/pcer/.ssh/id_rsa):
Created directory '/home/pcer/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/pcer/.ssh/id_rsa.
Your public key has been saved in /home/pcer/.ssh/id_rsa.pub.
The key fingerprint is:
0d:b7:b6:34:56:89:8c:16:ea:d7:a4:23:99:e2:90:37 pcer@master
The key's randomart image is:
+--[ RSA 2048]----+
|        .        |
|       . + . .   |
|      . + = o    |
|   . . + B o     |
|  o E = S O      |
|   + o o = o     |
|    .     .      |
|                 |
|                 |
+-----------------+
pcer@master:~$ cd .ssh/
pcer@master:~/.ssh$ ls
id_rsa  id_rsa.pub
pcer@master:~/.ssh$ 
```

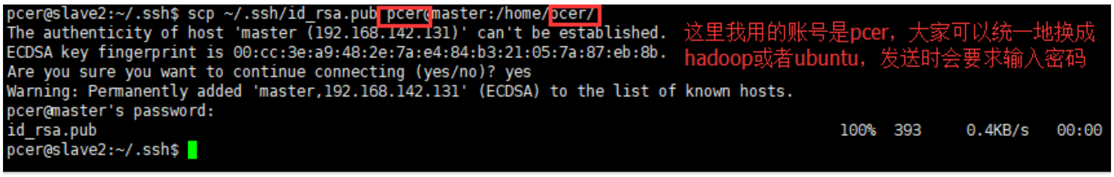
   b) 为了让每个节点都拥有其它节点的公钥，要先把所有公钥放进一个文件里，分 4 步走：

i. 在 master 上，将 master 的公钥复制到 authorized_keys 文件里：

```
cat ./id_rsa.pub >> ./authorized_keys    # cat 命令用于提取内容，>>输出重定向
```

ii. 将 slave1 的公钥文件发送给 master，此时的传送要输入密码：

```
scp ~/.ssh/id_rsa.pub hadoop@master:/home/hadoop/
```

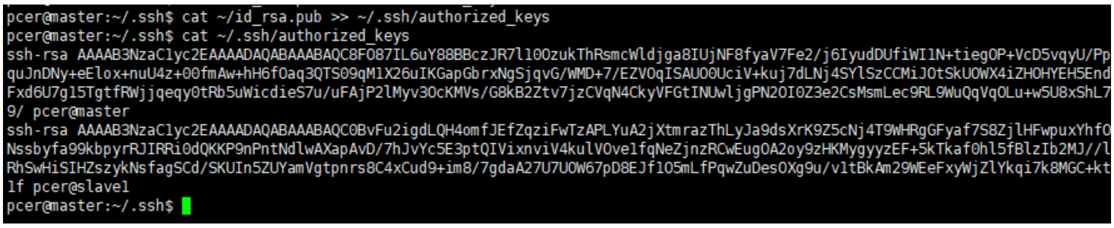


```
pcer@slave2:~/.ssh$ scp ~/.ssh/id_rsa.pub pcer@master:/home/pcer/
The authenticity of host 'master (192.168.142.131)' can't be established.
ECDSA key fingerprint is 00:cc:3e:a9:48:2e:7a:e4:84:b3:21:05:7a:87:eb:8b.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'master,192.168.142.131' (ECDSA) to the list of known hosts.
pcer@master's password:
id_rsa.pub                                                  100%  393     0.4KB/s   00:00
pcer@slave2:~/.ssh$
```

这里我用的账号是pcer，大家可以统一地换成hadoop或者ubuntu，发送时会要求输入密码

iii. Master 将接收到的 slave1 的公钥文件里的内容提取追加到 authorized_keys 文件里：

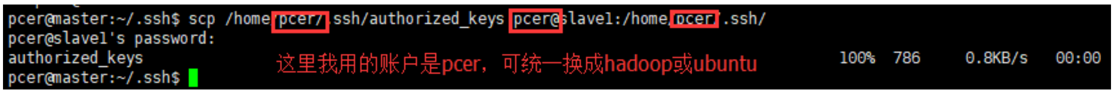
```
cat ~/id_rsa.pub >> ~/.ssh/authorized_keys
```



```
pcer@master:~/.ssh$ cat ~/id_rsa.pub >> ~/.ssh/authorized_keys
pcer@master:~/.ssh$ cat ~/.ssh/authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQC8FO87IL6uY88BBczJR7l10OzukThRsmcWldjga8IUjNF8fyaV7Fe2/j6IyudDUfiWI1N+tiegOP+VcD5vqyU/Pp
quJnDNy+eElox+nuU4z+00fmAw+hH6fOaq3QTS09qM1X26uIKGapGbrxNgSjqvG/WMD+7/EZVOqISAUO0UciV+kuj7dLNj4SYlSzCCMiJOtSkUOWX4iZHOHYEH5End
Fxd6U7gl5TgtfRWjjqeqy0tRb5uWicdieS7u/uFAjP2lMyv3OcKMVs/G8kB2Ztv7jzCVqN4CkyVFGtINUwljgPN2OI0Z3e2CsMsmLec9RL9WuQqVqOLu+w5U8xShL7
9/ pcer@master
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQC0BvFu2igdLQH4omfJEfZqziFwTzAPLYuA2jXtmrazThLyJa9dsXrK9Z5cNj4T9WHRgGFyaf7S8ZjlHFwpuxYhfO
Nssbyfa99kbpyrRJIRRi0dQKKP9nPntNdlwAXapAvD/7hJvYc5E3ptQIVixnviV4kulVOvelfqNeZjnzRCwEugOA2oy9zHKMygyyzEF+5kTkaf0hl5fBlzIb2MJ//l
RhSwHiSIHZszykNsfagSCd/SKUIn5ZUYamVgtpnrs8C4xCud9+im8/7gdaA27U7UOW67pD8EJf1O5mLfPqwZuDesOXg9u/vltBkAm29WEeFxyWjZlYkqi7k8MGC+kt
lf pcer@slave1
pcer@master:~/.ssh$
```

iv. 重复前两步，将 slave2 的公钥内容也放进 authorized_keys 文件，然后将 authorized_keys 文件分别发送到两个 slave 的~/.ssh/下：

```
scp ~/.ssh/authorized_keys hadoop@slave1:/home/hadoop/.ssh/
```

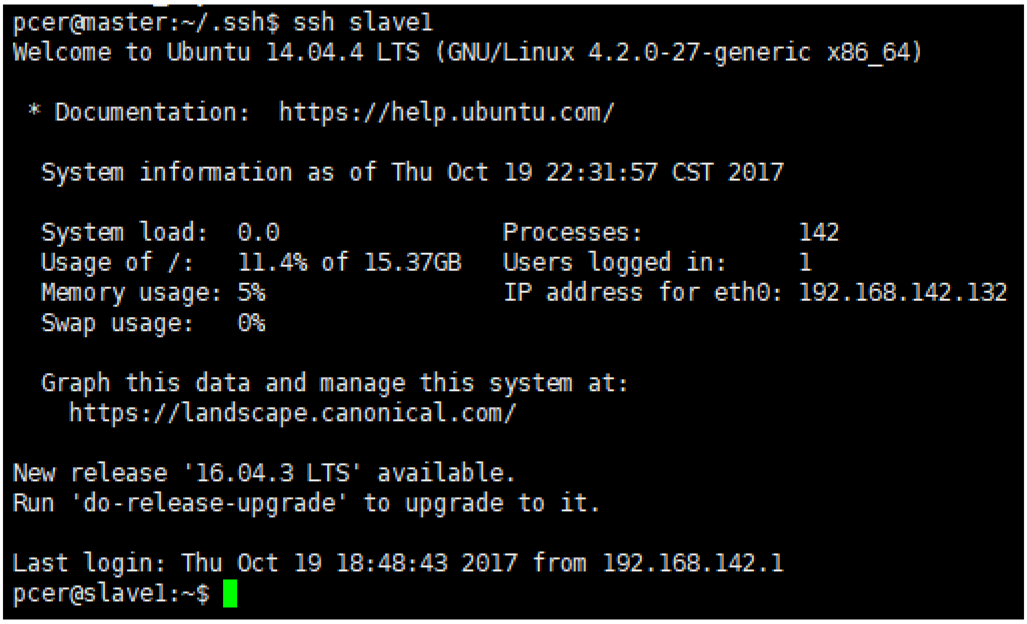


```
pcer@master:~/.ssh$ scp /home/pcer/.ssh/authorized_keys pcer@slave1:/home/pcer/.ssh/
pcer@slave1's password:
authorized_keys                                             100%  786     0.8KB/s   00:00
pcer@master:~/.ssh$
```

这里我用的账户是pcer，可统一换成hadoop或ubuntu

c) 每个节点尝试使用 ssh <角色名>的命令直接登录其它节点，直到每个节点都可以成功免密码登录其它节点，则免密码登录配置成功！ 如在 master 上输入：

```
ssh slave1
```



```
pcer@master:~/.ssh$ ssh slave1
Welcome to Ubuntu 14.04.4 LTS (GNU/Linux 4.2.0-27-generic x86_64)

 * Documentation:  https://help.ubuntu.com/

  System information as of Thu Oct 19 22:31:57 CST 2017

  System load:  0.0               Processes:           142
  Usage of /:   11.4% of 15.37GB   Users logged in:     1
  Memory usage: 5%                IP address for eth0: 192.168.142.132
  Swap usage:   0%

  Graph this data and manage this system at:
    https://landscape.canonical.com/

New release '16.04.3 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Thu Oct 19 18:48:43 2017 from 192.168.142.1
pcer@slave1:~$
```

说明：Hadoop 平台的运行需要 JDK 作为依托，Java 代码的执行同样也需要 JDK；如果是在天河二号上面的 OpenStack 云平台完成本次实验，则 JDK 和 Hadoop 的安装包都已经提前上传到/data/目录下了，直接从里面 mv 到家目录下操作就行；如果是在个人电脑上用 VMware 搭建的云平台，则需要在 Linux 系统中安装一个上传下载的程序包 lrzsz，命令为 sudo apt-get install lrzsz，安装完后，输入 rz 可以将外部文件传输到系统当前目录下，输入 sz <filename> 可以将文件传输到外部（如 sz test.cpp）；

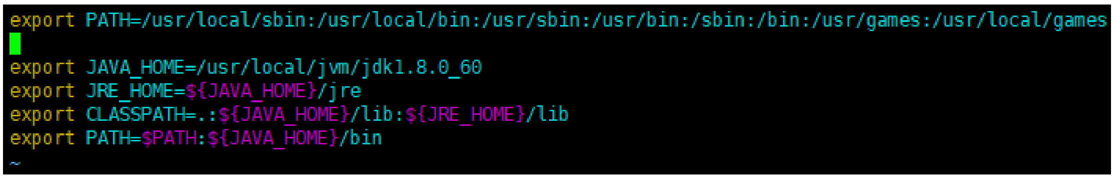
说明：第 4 步需要在三台机器上都各自做一遍

4. 安装 JDK；
   a) 将上传的 JDK 压缩包（jdk-8u60-linux-x64.tar）放到家目录（/home/hadoop/），解压并放到指定的文件夹：

```
sudo mkdir -p /usr/local/jvm
tar -zxvf jdk-8u60-linux-x64.tar.gz -C /usr/local/jvm
```

   b) 将当前的 PATH 环境变量提取保存到 setenv.sh，然后将其修改为初始化语句，增加 JAVA 的路径（我用的是 jdk1.7.0_80，大家相应地改成 jdk1.8.0_60）：

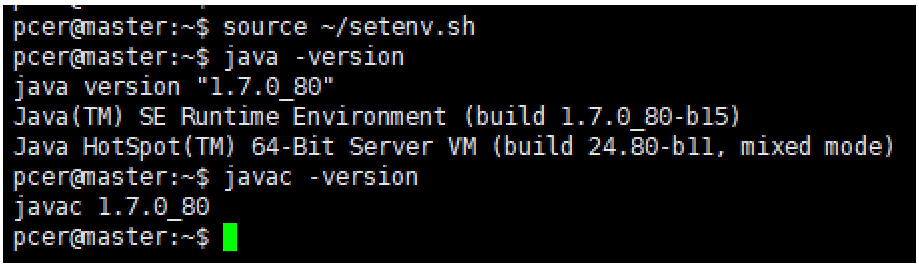
```
echo $PATH >> ~/setenv.sh
vi ~/setenv.sh
```



```
export PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games

export JAVA_HOME=/usr/local/jvm/jdk1.8.0_60
export JRE_HOME=${JAVA_HOME}/jre
export CLASSPATH=.:${JAVA_HOME}/lib:${JRE_HOME}/lib
export PATH=$PATH:${JAVA_HOME}/bin
~
```

   c) 执行 setenv.sh 脚本文件修改当前环境变量 PATH，然后尝试 java 和 javac 指令是否有效：

```
source ~/setenv.sh
java -version
javac -version
```



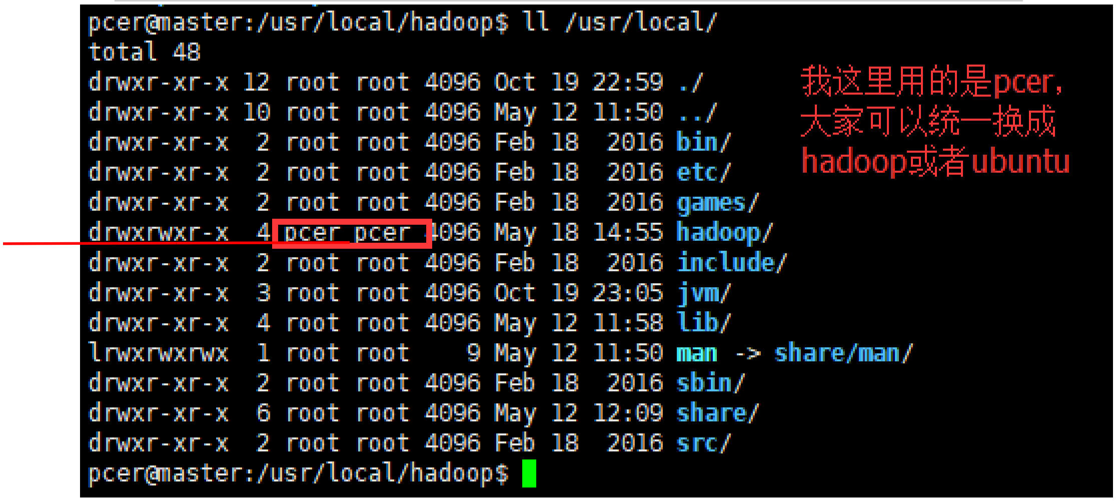
```
pcer@master:~$ source ~/setenv.sh
pcer@master:~$ java -version
java version "1.7.0_80"
Java(TM) SE Runtime Environment (build 1.7.0_80-b15)
Java HotSpot(TM) 64-Bit Server VM (build 24.80-b11, mixed mode)
pcer@master:~$ javac -version
javac 1.7.0_80
pcer@master:~$
```

说明：第 5 步需要在三台机器上都各自做一遍

5. 安装 Hadoop;
   a) 在各个节点上将 hadoop 解压到/usr/local/目录下，改变其所属用户和所属组（让 hadoop 软件用 hadoop 账号登录时对 hadoop 文件夹拥有最高权限）：

```
tar -zxvf hadoop-2.6.0.tar.gz -C /usr/local/
```

```
sudo mv /usr/loca/hadoop-2.6.0 /usr/local/hadoop     #mv 实现重命名
sudo chown -R hadoop:hadoop /usr/local/hadoop
```

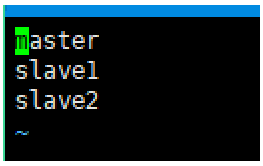


```
pcer@master:/usr/local/hadoop$ ll /usr/local/
total 48
drwxr-xr-x 12 root root 4096 Oct 19 22:59 ./
drwxr-xr-x 10 root root 4096 May 12 11:50 ../
drwxr-xr-x  2 root root 4096 Feb 18  2016 bin/
drwxr-xr-x  2 root root 4096 Feb 18  2016 etc/
drwxr-xr-x  2 root root 4096 Feb 18  2016 games/
drwxrwxr-x  4 pcer pcer 4096 May 18 14:55 hadoop/
drwxr-xr-x  2 root root 4096 Feb 18  2016 include/
drwxr-xr-x  3 root root 4096 Oct 19 23:05 jvm/
drwxr-xr-x  4 root root 4096 May 12 11:58 lib/
lrwxrwxrwx  1 root root    9 May 12 11:50 man -> share/man/
drwxr-xr-x  2 root root 4096 Feb 18  2016 sbin/
drwxr-xr-x  6 root root 4096 May 12 12:09 share/
drwxr-xr-x  2 root root 4096 Feb 18  2016 src/
pcer@master:/usr/local/hadoop$
```

我这里用的是pcer，大家可以统一换成hadoop或者ubuntu

b) 修改 slaves 文件，让 hadoop 知道自己可以聚合的节点名（保证与 hosts 里的角色名一致）：

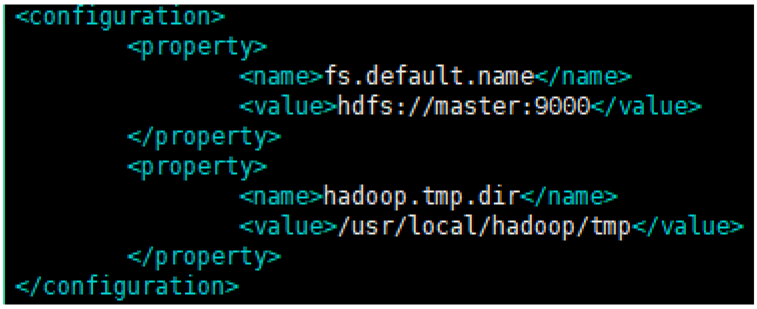
```
vi /usr/local/hadoop/etc/hadoop/slaves
```



```
master
slave1
slave2
~
```

c) 修改 core-site.xml 文件如下：

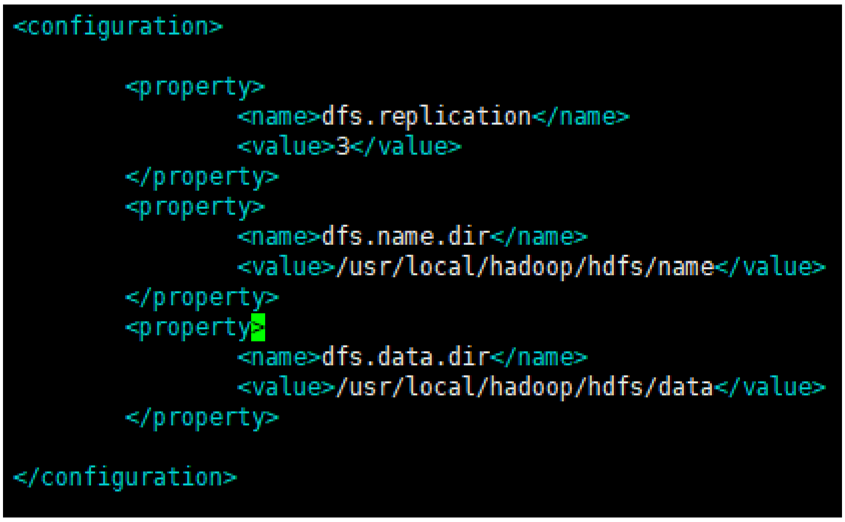
```
vi /usr/local/hadoop/etc/hadoop/core-site.xml
```



```
<configuration>
        <property>
                <name>fs.default.name</name>
                <value>hdfs://master:9000</value>
        </property>
        <property>
                <name>hadoop.tmp.dir</name>
                <value>/usr/local/hadoop/tmp</value>
        </property>
</configuration>
```

d) 修改 hdfs-site.xml 文件如下（启用所有节点作为 DataNode，故 replication=3）：

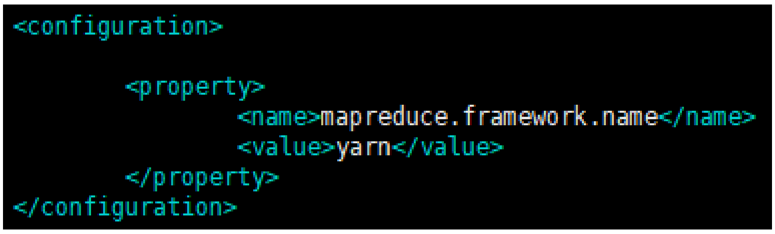
```
vi /usr/local/hadoop/etc/hadoop/hdfs-site.xml
```



```
<configuration>

        <property>
                <name>dfs.replication</name>
                <value>3</value>
        </property>
        <property>
                <name>dfs.name.dir</name>
                <value>/usr/local/hadoop/hdfs/name</value>
        </property>
        <property>
                <name>dfs.data.dir</name>
                <value>/usr/local/hadoop/hdfs/data</value>
        </property>

</configuration>
```

e) 修改 mapred-site.xml 文件如下：

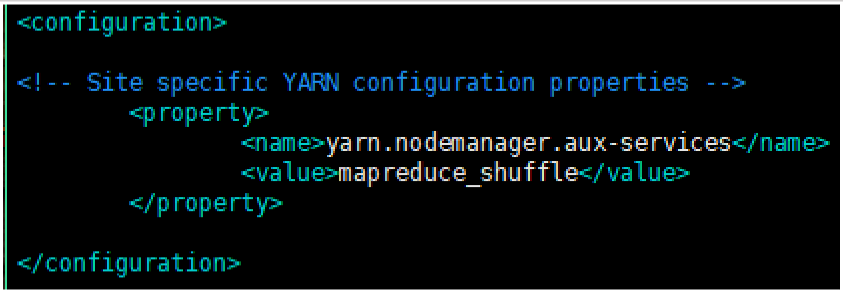
```
vi /usr/local/hadoop/etc/hadoop/mapred-site.xml
```



```
<configuration>

        <property>
                <name>mapreduce.framework.name</name>
                <value>yarn</value>
        </property>
</configuration>
```

f) 修改 yarn-site.xml 文件如下（启用 yarn 资源管理器）：

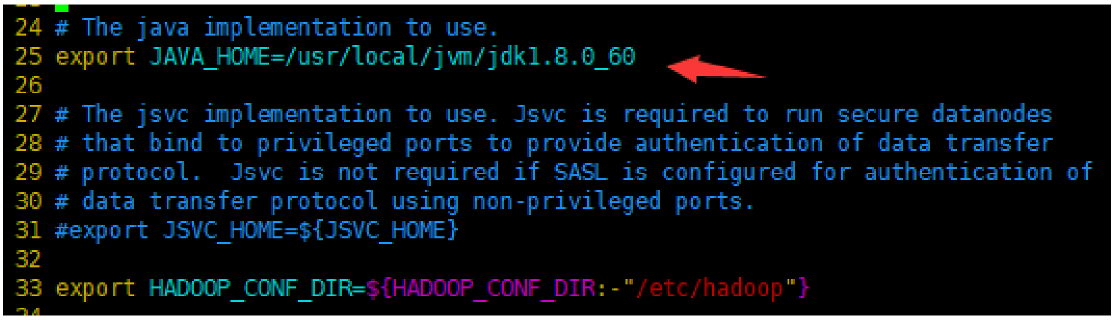
```
vi /usr/local/hadoop/etc/hadoop/yarn-site.xml
```



```
<configuration>

<!-- Site specific YARN configuration properties -->
        <property>
                <name>yarn.nodemanager.aux-services</name>
                <value>mapreduce_shuffle</value>
        </property>

</configuration>
```

g) 修改 hadoop-env.sh 文件，将 25 行 JAVA_HOME 的值换成 jdk 所在的路径：

```
vi /usr/local/hadoop/etc/hadoop/hadoop-env.sh
```



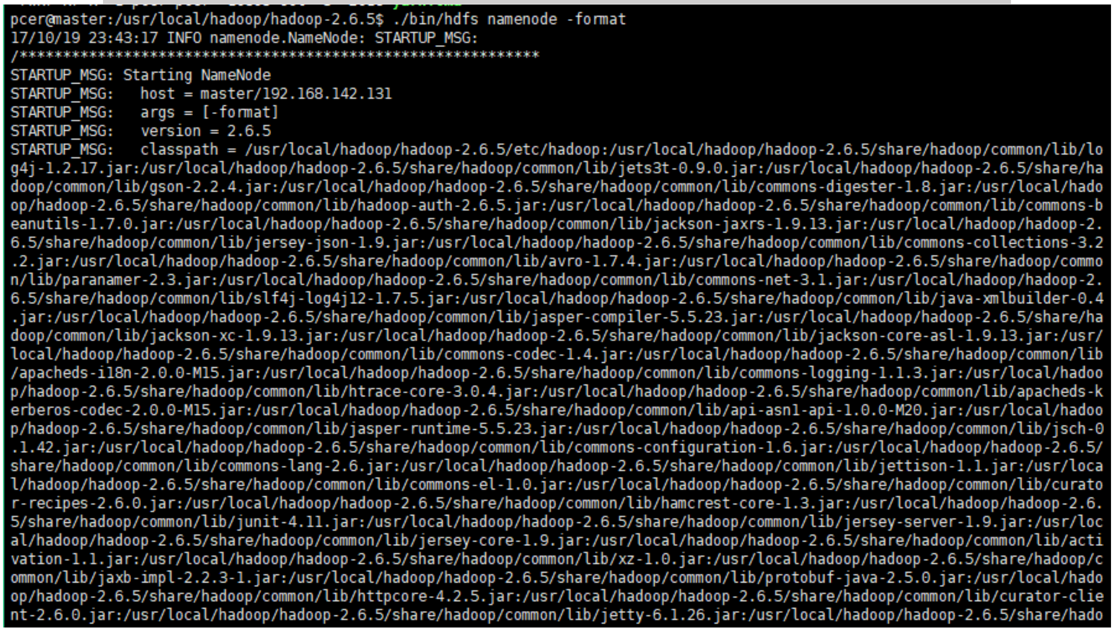
```
24 # The java implementation to use.
25 export JAVA_HOME=/usr/local/jvm/jdk1.8.0_60
26
27 # The jsvc implementation to use. Jsvc is required to run secure datanodes
28 # that bind to privileged ports to provide authentication of data transfer
29 # protocol.  Jsvc is not required if SASL is configured for authentication of
30 # data transfer protocol using non-privileged ports.
31 #export JSVC_HOME=${JSVC_HOME}
32
33 export HADOOP_CONF_DIR=${HADOOP_CONF_DIR:-"/etc/hadoop"}
```

说明：上述 Hadoop 的配置操作要在每个节点上做一次，确保每个环节都不出错，然后就可以尝试初始化 NameNode（聚合所有节点成为一个集群的服务），然后尝试启动各项服务。

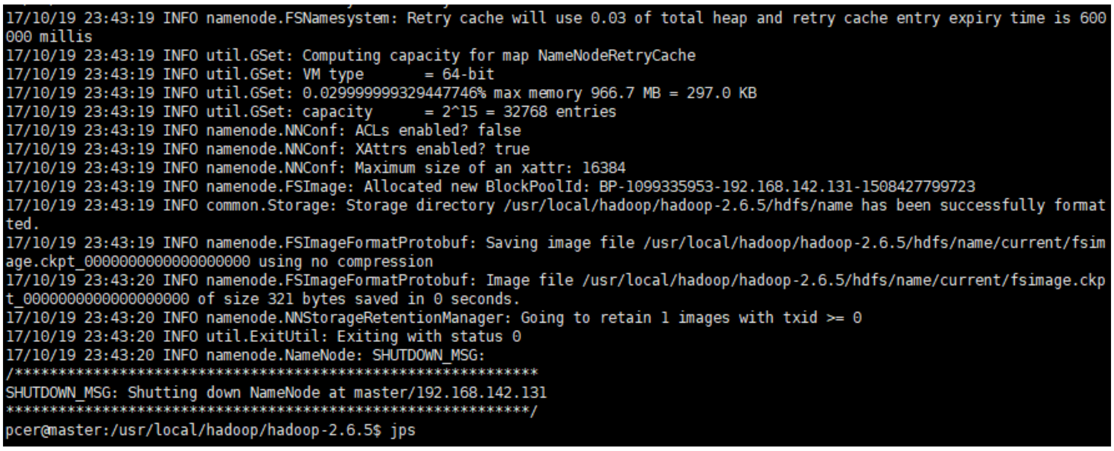
6. 启动及验证 hadoop；

a) 对 hadoop 进行 NameNode 的格式化：

```
/usr/local/hadoop/bin/hdfs namenode -format
```



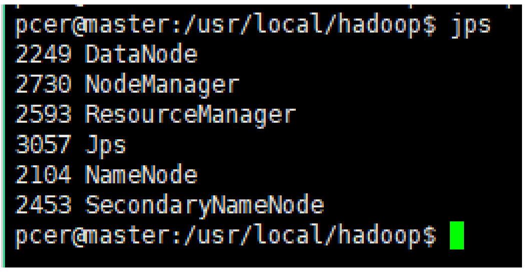
```
pcer@master:/usr/local/hadoop/hadoop-2.6.5$ ./bin/hdfs namenode -format
17/10/19 23:43:17 INFO namenode.NameNode: STARTUP_MSG:
/************************************************************
STARTUP_MSG: Starting NameNode
STARTUP_MSG:   host = master/192.168.142.131
STARTUP_MSG:   args = [-format]
STARTUP_MSG:   version = 2.6.5
STARTUP_MSG:   classpath = /usr/local/hadoop/hadoop-2.6.5/etc/hadoop:/usr/local/hadoop/hadoop-2.6.5/share/hadoop/common/lib/lo
g4j-1.2.17.jar:/usr/local/hadoop/hadoop-2.6.5/share/hadoop/common/lib/jets3t-0.9.0.jar:/usr/local/hadoop/hadoop-2.6.5/share/ha
doop/common/lib/gson-2.2.4.jar:/usr/local/hadoop/hadoop-2.6.5/share/hadoop/common/lib/commons-digester-1.8.jar:/usr/local/hado
op/hadoop-2.6.5/share/hadoop/common/lib/hadoop-auth-2.6.5.jar:/usr/local/hadoop/hadoop-2.6.5/share/hadoop/common/lib/commons-b
eanutils-1.7.0.jar:/usr/local/hadoop/hadoop-2.6.5/share/hadoop/common/lib/jackson-jaxrs-1.9.13.jar:/usr/local/hadoop/hadoop-2.
6.5/share/hadoop/common/lib/jersey-json-1.9.jar:/usr/local/hadoop/hadoop-2.6.5/share/hadoop/common/lib/commons-collections-3.2
.2.jar:/usr/local/hadoop/hadoop-2.6.5/share/hadoop/common/lib/avro-1.7.4.jar:/usr/local/hadoop/hadoop-2.6.5/share/hadoop/commo
n/lib/paranamer-2.3.jar:/usr/local/hadoop/hadoop-2.6.5/share/hadoop/common/lib/commons-net-3.1.jar:/usr/local/hadoop/hadoop-2.
6.5/share/hadoop/common/lib/slf4j-log4j12-1.7.5.jar:/usr/local/hadoop/hadoop-2.6.5/share/hadoop/common/lib/java-xmlbuilder-0.4
.jar:/usr/local/hadoop/hadoop-2.6.5/share/hadoop/common/lib/jasper-compiler-5.5.23.jar:/usr/local/hadoop/hadoop-2.6.5/share/ha
doop/common/lib/jackson-xc-1.9.13.jar:/usr/local/hadoop/hadoop-2.6.5/share/hadoop/common/lib/jackson-core-asl-1.9.13.jar:/usr/
local/hadoop/hadoop-2.6.5/share/hadoop/common/lib/commons-codec-1.4.jar:/usr/local/hadoop/hadoop-2.6.5/share/hadoop/common/lib
/apacheds-i18n-2.0.0-M15.jar:/usr/local/hadoop/hadoop-2.6.5/share/hadoop/common/lib/commons-logging-1.1.3.jar:/usr/local/hadoo
p/hadoop-2.6.5/share/hadoop/common/lib/htrace-core-3.0.4.jar:/usr/local/hadoop/hadoop-2.6.5/share/hadoop/common/lib/apacheds-k
erberos-codec-2.0.0-M15.jar:/usr/local/hadoop/hadoop-2.6.5/share/hadoop/common/lib/api-asn1-api-1.0.0-M20.jar:/usr/local/hadoo
p/hadoop-2.6.5/share/hadoop/common/lib/jasper-runtime-5.5.23.jar:/usr/local/hadoop/hadoop-2.6.5/share/hadoop/common/lib/jsch-0
.1.42.jar:/usr/local/hadoop/hadoop-2.6.5/share/hadoop/common/lib/commons-configuration-1.6.jar:/usr/local/hadoop/hadoop-2.6.5/
share/hadoop/common/lib/commons-lang-2.6.jar:/usr/local/hadoop/hadoop-2.6.5/share/hadoop/common/lib/jettison-1.1.jar:/usr/loca
l/hadoop/hadoop-2.6.5/share/hadoop/common/lib/commons-el-1.0.jar:/usr/local/hadoop/hadoop-2.6.5/share/hadoop/common/lib/curato
r-recipes-2.6.0.jar:/usr/local/hadoop/hadoop-2.6.5/share/hadoop/common/lib/hamcrest-core-1.3.jar:/usr/local/hadoop/hadoop-2.6.
5/share/hadoop/common/lib/junit-4.11.jar:/usr/local/hadoop/hadoop-2.6.5/share/hadoop/common/lib/jersey-server-1.9.jar:/usr/loc
al/hadoop/hadoop-2.6.5/share/hadoop/common/lib/jersey-core-1.9.jar:/usr/local/hadoop/hadoop-2.6.5/share/hadoop/common/lib/acti
vation-1.1.jar:/usr/local/hadoop/hadoop-2.6.5/share/hadoop/common/lib/xz-1.0.jar:/usr/local/hadoop/hadoop-2.6.5/share/hadoop/c
ommon/lib/jaxb-impl-2.2.3-1.jar:/usr/local/hadoop/hadoop-2.6.5/share/hadoop/common/lib/protobuf-java-2.5.0.jar:/usr/local/hado
op/hadoop-2.6.5/share/hadoop/common/lib/httpcore-4.2.5.jar:/usr/local/hadoop/hadoop-2.6.5/share/hadoop/common/lib/curator-clie
nt-2.6.0.jar:/usr/local/hadoop/hadoop-2.6.5/share/hadoop/common/lib/jetty-6.1.26.jar:/usr/local/hadoop/hadoop-2.6.5/share/hado
```



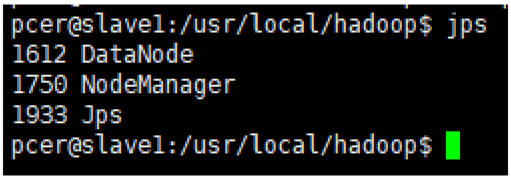
```
17/10/19 23:43:19 INFO namenode.FSNamesystem: Retry cache will use 0.03 of total heap and retry cache entry expiry time is 600
000 millis
17/10/19 23:43:19 INFO util.GSet: Computing capacity for map NameNodeRetryCache
17/10/19 23:43:19 INFO util.GSet: VM type       = 64-bit
17/10/19 23:43:19 INFO util.GSet: 0.029999999329447746% max memory 966.7 MB = 297.0 KB
17/10/19 23:43:19 INFO util.GSet: capacity      = 2^15 = 32768 entries
17/10/19 23:43:19 INFO namenode.NNConf: ACLs enabled? false
17/10/19 23:43:19 INFO namenode.NNConf: XAttrs enabled? true
17/10/19 23:43:19 INFO namenode.NNConf: Maximum size of an xattr: 16384
17/10/19 23:43:19 INFO namenode.FSImage: Allocated new BlockPoolId: BP-1099335953-192.168.142.131-1508427799723
17/10/19 23:43:19 INFO common.Storage: Storage directory /usr/local/hadoop/hadoop-2.6.5/hdfs/name has been successfully format
ted.
17/10/19 23:43:19 INFO namenode.FSImageFormatProtobuf: Saving image file /usr/local/hadoop/hadoop-2.6.5/hdfs/name/current/fsim
age.ckpt_0000000000000000000 using no compression
17/10/19 23:43:20 INFO namenode.FSImageFormatProtobuf: Image file /usr/local/hadoop/hadoop-2.6.5/hdfs/name/current/fsimage.ckp
t_0000000000000000000 of size 321 bytes saved in 0 seconds.
17/10/19 23:43:20 INFO namenode.NNStorageRetentionManager: Going to retain 1 images with txid >= 0
17/10/19 23:43:20 INFO util.ExitUtil: Exiting with status 0
17/10/19 23:43:20 INFO namenode.NameNode: SHUTDOWN_MSG:
/************************************************************
SHUTDOWN_MSG: Shutting down NameNode at master/192.168.142.131
************************************************************/
pcer@master:/usr/local/hadoop/hadoop-2.6.5$ jps
```

b) 启动 hdfs 和 yarn，并在各个节点上输入 jps 查看启动的服务：

```
/usr/local/hadoop/sbin/start-dfs.sh
/usr/local/hadoop/sbin/start-yarn.sh
jps        # 每个节点都查看一次
```



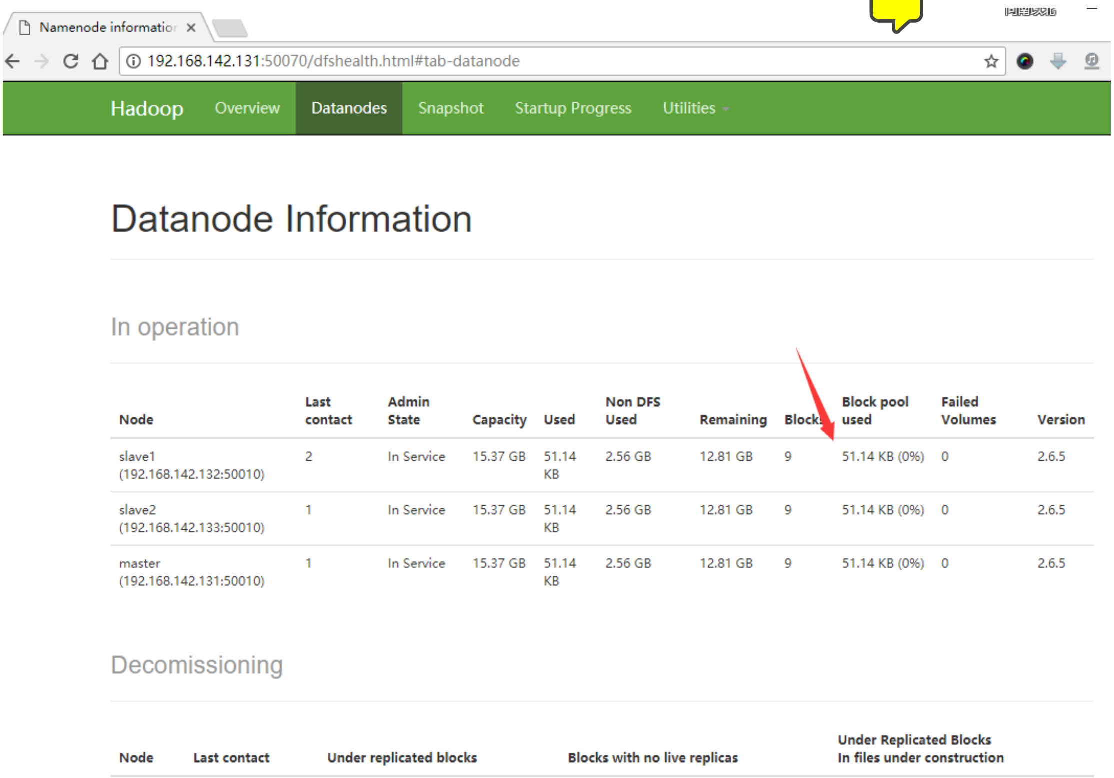
```
pcer@master:/usr/local/hadoop$ jps
2249 DataNode
2730 NodeManager
2593 ResourceManager
3057 Jps
2104 NameNode
2453 SecondaryNameNode
pcer@master:/usr/local/hadoop$
```



```
pcer@slave1:/usr/local/hadoop$ jps
1612 DataNode
1750 NodeManager
1933 Jps
pcer@slave1:/usr/local/hadoop$
```

c)　尝试在 hdfs 上创建输入文件夹 input, 并把 etc/hadoop 下的所有文本文件放进去：

```
/usr/loca/hadoop/bin/hdfs dfs -mkdir /input
/usr/loca/hadoop/bin/hdfs dfs -put /usr/local/hadoop/*.xml /input
```



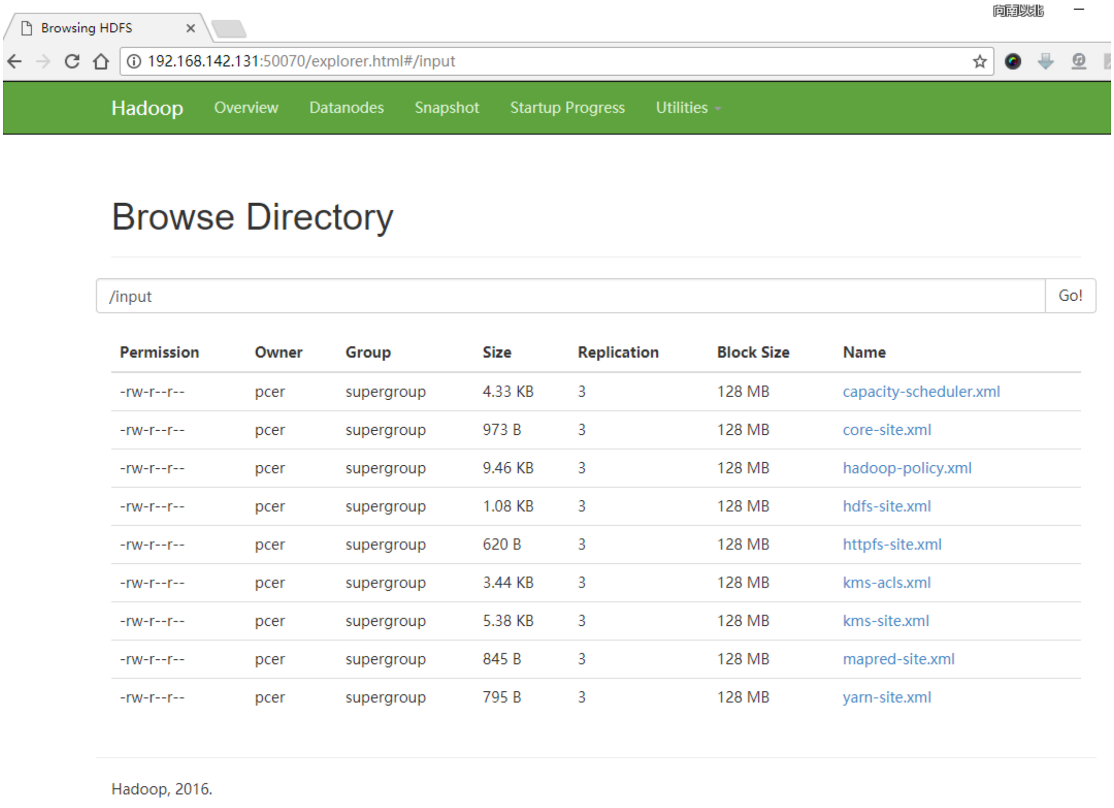
Datanode Information

In operation

Node	Last contact	Admin State	Capacity	Used	Non DFS Used	Remaining	Blocks	Block pool used	Failed Volumes	Version
slave1 (192.168.142.132:50010)	2	In Service	15.37 GB	51.14 KB	2.56 GB	12.81 GB	9	51.14 KB (0%)	0	2.6.5
slave2 (192.168.142.133:50010)	1	In Service	15.37 GB	51.14 KB	2.56 GB	12.81 GB	9	51.14 KB (0%)	0	2.6.5
master (192.168.142.131:50010)	1	In Service	15.37 GB	51.14 KB	2.56 GB	12.81 GB	9	51.14 KB (0%)	0	2.6.5

Decomissioning

Node	Last contact	Under replicated blocks	Blocks with no live replicas	Under Replicated Blocks In files under construction



Browse Directory

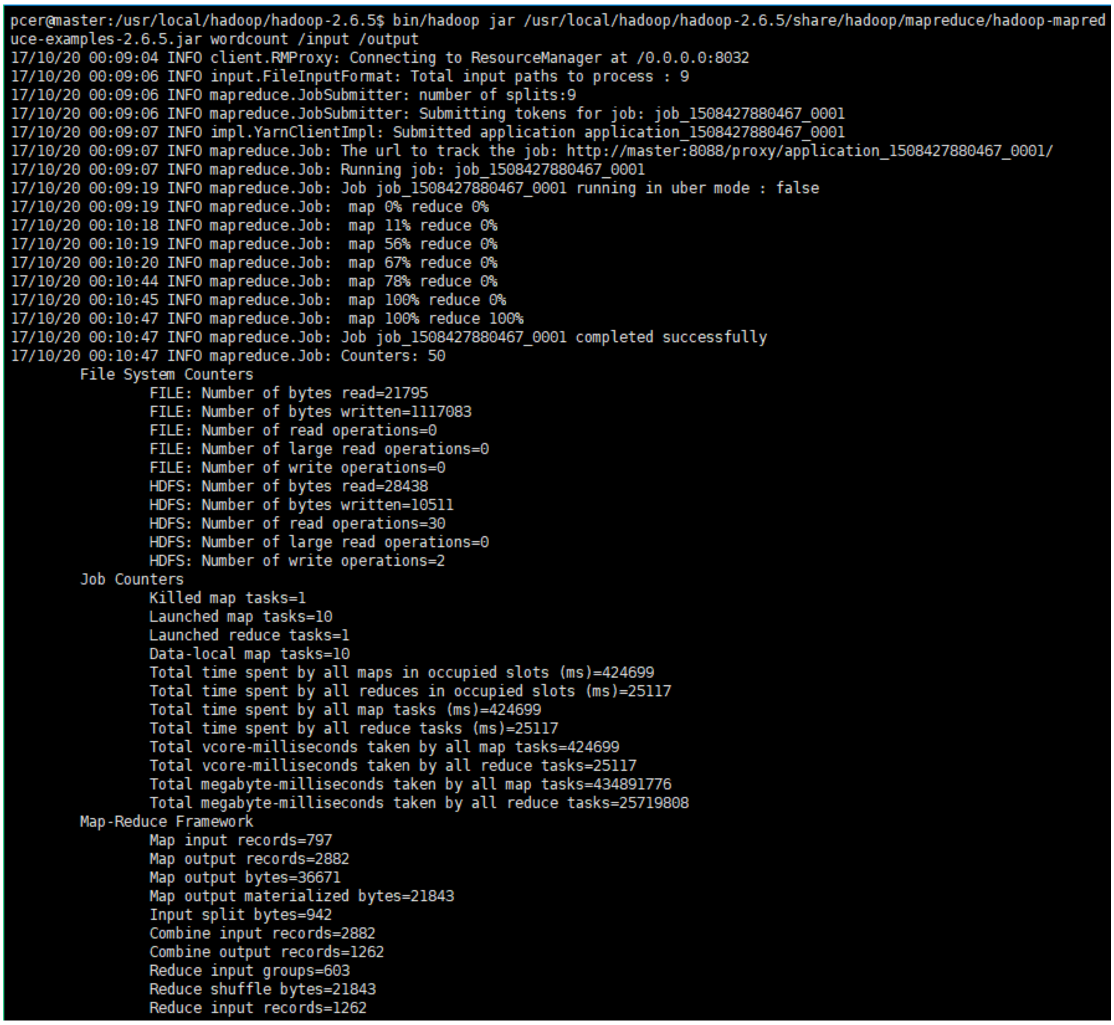
Permission	Owner	Group	Size	Replication	Block Size	Name
-rw-r--r--	pcer	supergroup	4.33 KB	3	128 MB	capacity-scheduler.xml
-rw-r--r--	pcer	supergroup	973 B	3	128 MB	core-site.xml
-rw-r--r--	pcer	supergroup	9.46 KB	3	128 MB	hadoop-policy.xml
-rw-r--r--	pcer	supergroup	1.08 KB	3	128 MB	hdfs-site.xml
-rw-r--r--	pcer	supergroup	620 B	3	128 MB	httpfs-site.xml
-rw-r--r--	pcer	supergroup	3.44 KB	3	128 MB	kms-acls.xml
-rw-r--r--	pcer	supergroup	5.38 KB	3	128 MB	kms-site.xml
-rw-r--r--	pcer	supergroup	845 B	3	128 MB	mapred-site.xml
-rw-r--r--	pcer	supergroup	795 B	3	128 MB	yarn-site.xml

Hadoop, 2016.

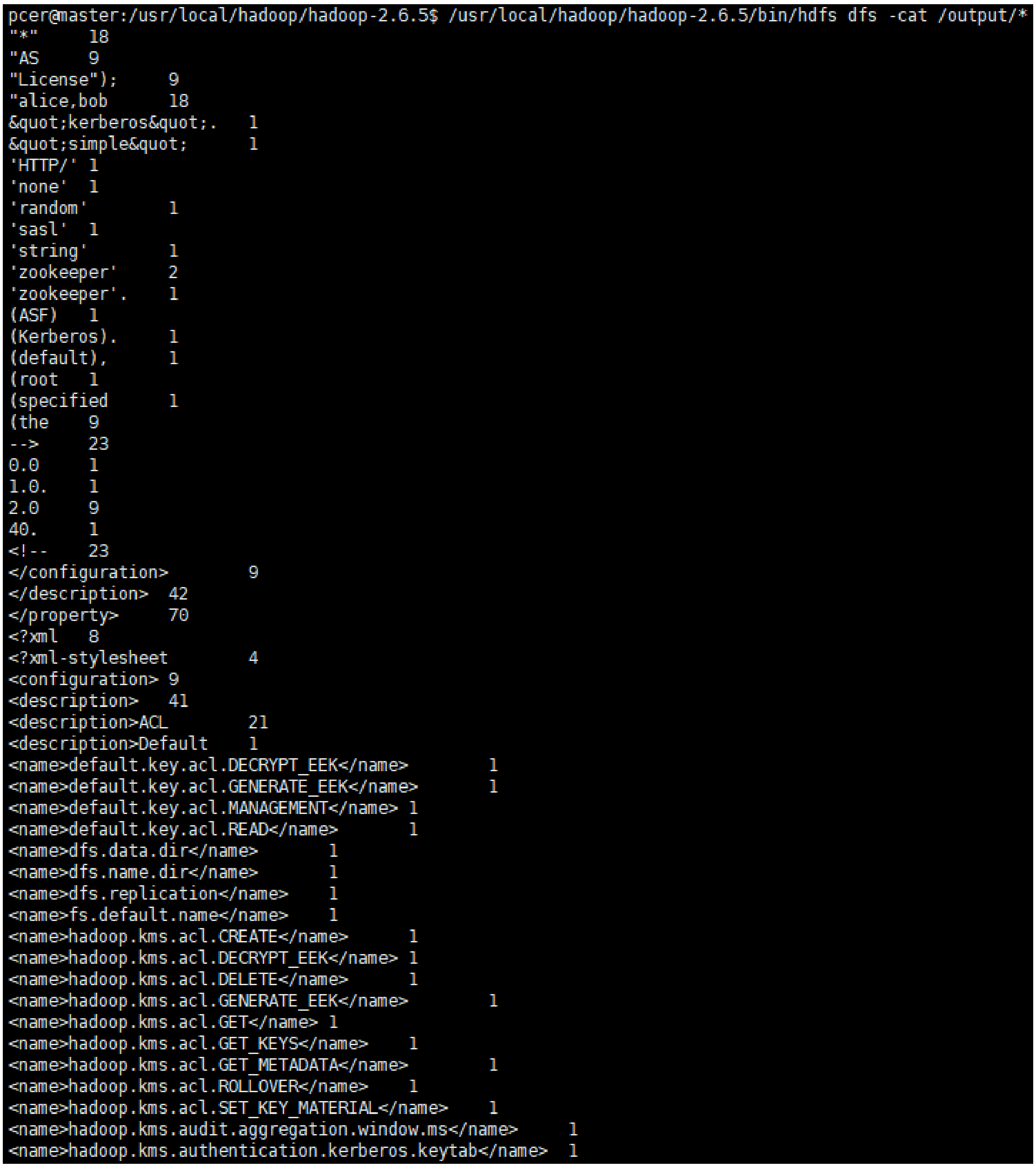
可以在外部浏览器输入 master 的 IP 地址和 50070 端口查看 hdfs 上的文件

d) 尝试用 hadoop 启动自带的 WordCount 样例代码，统计上面文本文件中每个单词出现的频数：

```
/usr/loca/hadoop/bin/hadoop jar
  /usr/local/hadoop/share/hadoop/mapreduce/hadoop-mapreduce-
  examples-2.6.5.jar wordcount /input /output
/usr/loca/hadoop/bin/hdfs dfs -cat /output/*
```



```
pcer@master:/usr/local/hadoop/hadoop-2.6.5$ bin/hadoop jar /usr/local/hadoop/hadoop-2.6.5/share/hadoop/mapreduce/hadoop-mapred
uce-examples-2.6.5.jar wordcount /input /output
17/10/20 00:09:04 INFO client.RMProxy: Connecting to ResourceManager at /0.0.0.0:8032
17/10/20 00:09:06 INFO input.FileInputFormat: Total input paths to process : 9
17/10/20 00:09:06 INFO mapreduce.JobSubmitter: number of splits:9
17/10/20 00:09:06 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1508427880467_0001
17/10/20 00:09:07 INFO impl.YarnClientImpl: Submitted application application_1508427880467_0001
17/10/20 00:09:07 INFO mapreduce.Job: The url to track the job: http://master:8088/proxy/application_1508427880467_0001/
17/10/20 00:09:07 INFO mapreduce.Job: Running job: job_1508427880467_0001
17/10/20 00:09:19 INFO mapreduce.Job: Job job_1508427880467_0001 running in uber mode : false
17/10/20 00:09:19 INFO mapreduce.Job:  map 0% reduce 0%
17/10/20 00:10:18 INFO mapreduce.Job:  map 11% reduce 0%
17/10/20 00:10:19 INFO mapreduce.Job:  map 56% reduce 0%
17/10/20 00:10:20 INFO mapreduce.Job:  map 67% reduce 0%
17/10/20 00:10:44 INFO mapreduce.Job:  map 78% reduce 0%
17/10/20 00:10:45 INFO mapreduce.Job:  map 100% reduce 0%
17/10/20 00:10:47 INFO mapreduce.Job:  map 100% reduce 100%
17/10/20 00:10:47 INFO mapreduce.Job: Job job_1508427880467_0001 completed successfully
17/10/20 00:10:47 INFO mapreduce.Job: Counters: 50
        File System Counters
                FILE: Number of bytes read=21795
                FILE: Number of bytes written=1117083
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=28438
                HDFS: Number of bytes written=10511
                HDFS: Number of read operations=30
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=2
        Job Counters
                Killed map tasks=1
                Launched map tasks=10
                Launched reduce tasks=1
                Data-local map tasks=10
                Total time spent by all maps in occupied slots (ms)=424699
                Total time spent by all reduces in occupied slots (ms)=25117
                Total time spent by all map tasks (ms)=424699
                Total time spent by all reduce tasks (ms)=25117
                Total vcore-milliseconds taken by all map tasks=424699
                Total vcore-milliseconds taken by all reduce tasks=25117
                Total megabyte-milliseconds taken by all map tasks=434891776
                Total megabyte-milliseconds taken by all reduce tasks=25719808
        Map-Reduce Framework
                Map input records=797
                Map output records=2882
                Map output bytes=36671
                Map output materialized bytes=21843
                Input split bytes=942
                Combine input records=2882
                Combine output records=1262
                Reduce input groups=603
                Reduce shuffle bytes=21843
                Reduce input records=1262
```



```
pcer@master:/usr/local/hadoop/hadoop-2.6.5$ /usr/local/hadoop/hadoop-2.6.5/bin/hdfs dfs -cat /output/*
"*"     18
"AS     9
"License");     9
"alice,bob      18
&quot;kerberos&quot;.   1
&quot;simple&quot;      1
'HTTP/' 1
'none'  1
'random'        1
'sasl'  1
'string'        1
'zookeeper'     2
'zookeeper'.    1
(ASF)   1
(Kerberos).     1
(default),      1
(root   1
(specified      1
(the    9
-->     23
0.0     1
1.0.    1
2.0     9
40.     1
<!--    23
</configuration>        9
</description>  42
</property>     70
<?xml   8
<?xml-stylesheet        4
<configuration> 9
<description>   41
<description>ACL        21
<description>Default    1
<name>default.key.acl.DECRYPT_EEK</name>        1
<name>default.key.acl.GENERATE_EEK</name>       1
<name>default.key.acl.MANAGEMENT</name> 1
<name>default.key.acl.READ</name>       1
<name>dfs.data.dir</name>       1
<name>dfs.name.dir</name>       1
<name>dfs.replication</name>    1
<name>fs.default.name</name>    1
<name>hadoop.kms.acl.CREATE</name>      1
<name>hadoop.kms.acl.DECRYPT_EEK</name> 1
<name>hadoop.kms.acl.DELETE</name>      1
<name>hadoop.kms.acl.GENERATE_EEK</name>        1
<name>hadoop.kms.acl.GET</name> 1
<name>hadoop.kms.acl.GET_KEYS</name>    1
<name>hadoop.kms.acl.GET_METADATA</name>        1
<name>hadoop.kms.acl.ROLLOVER</name>    1
<name>hadoop.kms.acl.SET_KEY_MATERIAL</name>    1
<name>hadoop.kms.audit.aggregation.window.ms</name>     1
<name>hadoop.kms.authentication.kerberos.keytab</name>  1
```

- 上述结果一切正常则可交由 TA 验收；
- 示意图片里出现了 pcer 和 hadoop-2.6.5 皆为原来使用的用户名和安装的路径，将 pcer 切换成 hadoop 以及无视 hadoop-2.6.5 即可；
- 请注意理解每一步的具体用意，尝试去理解每一份配置文件的作用；
- 在 lab1 或 lab2 的基础上完成本次实验，后面的实验将基于本次实验搭建的平台进行，所以请注意保留本次实验结果；
- 有事问 TA；