

## 第 3 章 Hadoop 的安装与配置

### 3.1. 伪分布模式安装

hadoop 的安装分为本地模式、伪分布模式、集群模式。本地模式是运行在本地，只负责存储，没有计算功能，本书不讲述。伪分布模式是在一台机器上模拟分布式部署，方便学习和调试。集群模式是在多个机器上配置 hadoop，是真正的“分布式”。[本章讲述伪分布模式和集群的安装。](#)

#### 3.1.1. 解压缩 hadoop

使用 WinSCP 把压缩包软件 `hadoop-1.10.24.tar.gz` 从宿主机复制移动到 linux 的 `/usr/local` 目录下，如图 3-1。

```
[root@book0 Desktop]# pwd
/root/Desktop
[root@book0 Desktop]# ls
hadoop-1.0.4.tar.gz
[root@book0 Desktop]# mv hadoop-1.0.4.tar.gz /usr/local
[root@book0 Desktop]# cd /usr/local
[root@book0 local]# ls
bin  games  include  jdk-6u24-linux-i586.bin
etc  hadoop-1.0.4.tar.gz  jdk  lib
[root@book0 local]#
```

图 3-1

解压缩文件，并重命名为 hadoop，方便使用。[重命名后这时](#)，hadoop 目录的完整路径是“`/usr/local/hadoop`”。

```
#tar -xvzf hadoop-1.0.4.tar.gz
#mv hadoop-1.0.4 hadoop
```

设置环境变量 `HADOOP_HOME`，修改文件“`/etc/profile`”，如下图：

```
alias cdha='cd /usr/local/hadoop'
|
export JAVA_HOME=/usr/local/jdk
export HADOOP_HOME=/usr/local/hadoop

export PATH=.:$PATH:$JAVA_HOME/bin:$HADOOP_HOME/bin
Plain Text ▾ Tab Width: 8 ▾ Ln 81, Col 1
```

图 3-2

请读者与 jdk 设置时配置文件对照。这里我们设置了一个别名 cdha，可以快速转到 hadoop 的目录。

修改环境变量后，记得执行 source 命令哦。

现在让我们浏览一下 hadoop 的目录结构吧

```
[root@book0 hadoop]# ls -l
total 7520
drwxr-xr-x. 2 root root 4096 Jun 4 01:12 bin ← 各种运行命令
-rw-rw-r--. 1 root root 119875 Oct 3 2012 build.xml
drwxr-xr-x. 4 root root 4096 Oct 3 2012 c++
-rw-rw-r--. 1 root root 446999 Oct 3 2012 CHANGES.txt
drwxr-xr-x. 2 root root 4096 Jun 4 01:12 conf ← 各种配置文件
drwxr-xr-x. 10 root root 4096 Oct 3 2012 contrib
drwxr-xr-x. 7 root root 4096 Jun 4 01:12 docs ← api等
-rw-rw-r--. 1 root root 6840 Oct 3 2012 hadoop-ant-1.0.4.jar
-rw-rw-r--. 1 root root 410 Oct 3 2012 hadoop-client-1.0.4.jar
-rw-rw-r--. 1 root root 3928530 Oct 3 2012 hadoop-core-1.0.4.jar
-rw-rw-r--. 1 root root 142452 Oct 3 2012 hadoop-examples-1.0.4.jar
-rw-rw-r--. 1 root root 413 Oct 3 2012 hadoop-minicluster-1.0.4.jar
-rw-rw-r--. 1 root root 2656646 Oct 3 2012 hadoop-test-1.0.4.jar
-rw-rw-r--. 1 root root 287807 Oct 3 2012 hadoop-tools-1.0.4.jar
drwxr-xr-x. 2 root root 4096 Jun 4 01:12 ivy
-rw-rw-r--. 1 root root 10525 Oct 3 2012 ivy.xml
drwxr-xr-x. 5 root root 4096 Jun 4 01:12 lib
drwxr-xr-x. 2 root root 4096 Jun 4 01:12 libexec
-rw-rw-r--. 1 root root 13366 Oct 3 2012 LICENSE.txt
-rw-rw-r--. 1 root root 101 Oct 3 2012 NOTICE.txt
-rw-rw-r--. 1 root root 1366 Oct 3 2012 README.txt
drwxr-xr-x. 2 root root 4096 Jun 4 01:12 sbin
drwxr-xr-x. 3 root root 4096 Oct 3 2012 share
drwxr-xr-x. 16 root root 4096 Jun 4 01:12 src
drwxr-xr-x. 9 root root 4096 Oct 3 2012 webapps
[root@book0 hadoop]#
```

图 3-3

我们关注 bin 目录和 conf 目录。

### 3.1.2. 修改配置文件

hadoop 配置文件默认是本地模式，我们修改四个配置文件，这些文件都位于 \$HADOOP\_HOME/conf 目录下。

第一个是 hadoop 环境变量脚本文件 hadoop-env.sh。

修改第 9 行代码为

```
export JAVA_HOME=/usr/local/jdk
```

保存并关闭。这里设置的是 JAVA\_HOME，注意去掉前面的“#”。

第二个是 hadoop 核心配置文件 core-site.xml，结果如下

```
<configuration>
  <property>
    <name>hadoop.tmp.dir</name>
    <value>/home/hadoop/tmp</value>
    <description>hadoop 的运行临时文件的主目录</description>
  </property>
  <property>
    <name>fs.default.name</name>
    <value>hdfs://book0:9000</value>
    <description>HDFS 的访问路径</description>
  </property>
</configuration>
```

第三个是 hdfs 配置文件 hdfs-site.xml，结果如下

```
<configuration>
  <property>
    <name>dfs.replication</name>
    <value>1</value>
    <description>存储副本数</description>
  </property>
</configuration>
```

第四个是 MapReduce 配置文件 mapred-site.xml，结果如下

```
<configuration>
  <property>
    <name>mapred.job.tracker</name>
    <value>book0:9001</value>
    <description>JobTracker 的访问路径</description>
  </property>
</configuration>
```

这是安装伪分布模式的最小化配置。目前的任务是把 hadoop 跑起来，先不关注各配置项什么含义，这些配置项还在后文会有详细的解释。

### 3.1.3. 格式化文件系统

hdfs 是文件系统，所以在第一次使用之前需要进行格式化。执行命令 \$HADOOP\_HOME/bin/hadoop namenode -format。见图 3-4

```
[root@book0 conf]# hadoop namenode -format
Warning: $HADOOP_HOME is deprecated.

13/06/04 04:50:16 INFO namenode.NameNode: STARTUP_MSG:
/*****
STARTUP_MSG: Starting NameNode
STARTUP_MSG:   host = book0/192.168.1.100
STARTUP_MSG:   args = [-format]
STARTUP_MSG:   version = 1.0.4
STARTUP_MSG:   build = https://svn.apache.org/repos/asf/hadoop/common/branches/branch-1.0 -r 1393290; compiled by '
*****/
13/06/04 04:50:17 INFO util.GSet: VM type      = 32-bit
13/06/04 04:50:17 INFO util.GSet: 2% max memory = 19.33375 MB
13/06/04 04:50:17 INFO util.GSet: capacity    = 2^22 = 4194304 entries
13/06/04 04:50:17 INFO util.GSet: recommended=4194304, actual=4194304
13/06/04 04:50:17 INFO namenode.FSNamesystem: fsOwner=root
13/06/04 04:50:17 INFO namenode.FSNamesystem: supergroup=supergroup
13/06/04 04:50:17 INFO namenode.FSNamesystem: isPermissionEnabled=true
13/06/04 04:50:17 INFO namenode.FSNamesystem: dfs.block.invalidate.limit=100
13/06/04 04:50:17 INFO namenode.FSNamesystem: isAccessTokenEnabled=false accessKeyUpdateInterval=0 min(s), accessTo
13/06/04 04:50:17 INFO namenode.NameNode: Caching file names occurring more than 10 times
13/06/04 04:50:18 INFO common.Storage: Image file of size 110 saved in 0 seconds.
13/06/04 04:50:18 INFO common.Storage: Storage directory /home/hadoop/tmp/dfs/name has been successfully formatted.
13/06/04 04:50:18 INFO namenode.NameNode: SHUTDOWN_MSG:
/*****
SHUTDOWN_MSG: Shutting down NameNode at book0/192.168.1.100
*****/
[root@book0 conf]#
```

图 3-4

**注意：**只在第一次启动的时候格式化，不要每次启动都格式化。理解为我们新买了块移动硬盘，使用之前总要格式化硬盘。

如果真的有必要再次格式化，请先把“\$HADOOP\_HOME/tmp”目录下的文件全部删除。

读者可以自己观察目录“\$HADOOP\_HOME/tmp”在格式化前后的变化情况。

格式化操作很少有出现失败的情况。如果真出现了，请检查配置是否正确。

### 3.1.4. 启动

格式化完成后，开始启动 hadoop 程序。

启动 hadoop 的命令脚本都在\$HADOOP\_HOME/bin/下，下面的所有命令都不再带有完整路径名称。

这里讲述 hadoop 启动的三种方式：

**第一种，一次性全部启动：**

```
[root@book0 conf]# start-all.sh
Warning: $HADOOP_HOME is deprecated.

starting namenode, logging to /usr/local/hadoop/libexec/./
localhost: starting datanode, logging to /usr/local/hadoop/
localhost: starting secondarynamenode, logging to /usr/loca
starting jobtracker, logging to /usr/local/hadoop/libexec/.
localhost: starting tasktracker, logging to /usr/local/hadc
```

图 3-5

执行 start-all.sh 启动 hadoop，观察控制台的输出，见图 3-5,可以看到正在启动进程，分别是 namenode、datanode、secondarynamenode、jobtracker、tasktracker，一共 5 个，待执行完毕后，并不意味着这 5 个进程成功启动，上面仅仅表示系统正在启动进程而已。

我们使用 jdk 的命令 jps 查看进程是否已经正确启动。执行以下 jps，如果看到了这 5 个进程，见图 3-6,说明 hadoop 真的启动成功了。如果缺少一个或者多个，那就进入到“Hadoop 的常见启动错误”章节寻找原因了。

```
[root@book0 conf]# jps
8027 Jps
7651 DataNode
7777 SecondaryNameNode
7543 NameNode
7970 TaskTracker
7864 JobTracker
```

图 3-6

关闭 hadoop 的命令是 stop-all.sh。

上面的命令是最简单的，可以一次性把所有节点都启动、关闭。除此之外，还有其他命令，是分别启动的。

**第二种，分别启动 HDFS 和 MapReduce：**

```

[root@book0 conf]# start-dfs.sh
Warning: $HADOOP_HOME is deprecated.

starting namenode, logging to /usr/local/hadoop/
localhost: starting datanode, logging to /usr/lo
localhost: starting secondarynamenode, logging t
[root@book0 conf]# jps
11598 DataNode
11737 SecondaryNameNode
11794 Jps
11483 NameNode
[root@book0 conf]# █

```

图 3-7

执行命令 `start-dfs.sh`，是单独启动 hdfs，见图 3-7。执行完该命令后，通过 `jps` 能够看到 NameNode、DataNode、SecondaryNameNode 三个进程启动了，该命令适合于只执行 hdfs 存储不使用 MapReduce 做计算的场景。关闭的命令就是 `stop-dfs.sh` 了。

```

[root@book0 conf]# start-mapred.sh
Warning: $HADOOP_HOME is deprecated.

starting jobtracker, logging to /usr/local/
localhost: starting tasktracker, logging to
[root@book0 conf]# jps
12540 Jps
12368 JobTracker
12487 TaskTracker
[root@book0 conf]# █

```

图 3-8

执行命令 `start-mapred.sh`，可以单独启动 MapReduce 的两个进程。关闭的命令就是 `stop-mapred.sh` 了。当然，也可以先启动 MapReduce，再启动 HDFS。这说明，HDFS 和 MapReduce 的进程之间是互相独立的，没有依赖关系。

### 第三种，分别启动各个进程：

```

[root@book0 bin]# jps
14821 Jps
[root@book0 bin]# hadoop-daemon.sh start namenode
[root@book0 bin]# hadoop-daemon.sh start datanode
[root@book0 bin]# hadoop-daemon.sh start secondarynamenode
[root@book0 bin]# hadoop-daemon.sh start jobtracker
[root@book0 bin]# hadoop-daemon.sh start tasktracker
[root@book0 bin]# jps
14855 NameNode
14946 DataNode
15043 SecondaryNameNode
15196 TaskTracker
15115 JobTracker
15303 Jps

```

执行的命令是“`hadoop-daemon.sh start [进程名称]`”，这种启动方式适合于单独增加、删除节点的情况，在安装集群环境的时候会看到。

### 3.1.5. 运行简单的 MapReduce 计算

在\$HADOOP\_HOME下有个 jar 包, 叫 `hadoop-example-1.1.2.jar`, 见图 3-9, 这里面含有框架提供的很多例子. 我们现在学习一下如何运行其中的例子吧.

```
[root@hadoop0 hadoop]# pwd
/usr/local/hadoop
[root@hadoop0 hadoop]# ls
bin                hadoop-ant-1.1.2.jar      ivy.xml            内置的例子
build.xml          hadoop-client-1.1.2.jar  lib                share
c++               hadoop-core-1.1.2.jar    libexec            src
CHANGES.txt      hadoop-examples-1.1.2.jar LICENSE.txt         TestDFSIO_results.log
conf              hadoop-minicluster-1.1.2.jar logs                tmp
contrib           hadoop-test-1.1.2.jar   NOTICE.txt         webapps
dfs               hadoop-tools-1.1.2.jar  pids
docs              ivy                      README.txt
```

图 3-9

执行如下命令

```
hadoop jar hadoop-example-1.1.2.jar
```

可以看到图 3-10 的输出信息, 可以看到 18 个输出信息, 都是内置的例子程序. 我们选择最后一个“word count”来运行, 该程序的说明在最后, 是统计文件中单词的出现次数的.

```
[root@hadoop0 hadoop]# hadoop jar hadoop-examples-1.1.2.jar
Warning: $HADOOP_HOME is deprecated.

An example program must be given as the first argument. 查看jar中程序的命令
Valid program names are:
  aggregatewordcount: An Aggregate based map/reduce program that counts the word
s in the input files.
  aggregatewordhist: An Aggregate based map/reduce program that computes the his
togram of the words in the input files. 这是内置的各种程序
  dbcount: An example job that count the pageview counts from a database.
  grep: A map/reduce program that counts the matches of a regex in the input.
  join: A job that effects a join over sorted, equally partitioned datasets
  multifilewc: A job that counts words from several files.
  pentomino: A map/reduce tile laying program to find solutions to pentomino pro
blems.
  pi: A map/reduce program that estimates Pi using monte-carlo method.
  randomtextwriter: A map/reduce program that writes 10GB of random textual data
per node.
  randomwriter: A map/reduce program that writes 10GB of random data per node.
  secondarysort: An example defining a secondary sort to the reduce.
  sleep: A job that sleeps at each map and reduce task.
  sort: A map/reduce program that sorts the data written by the random writer.
  sudoku: A sudoku solver.
  teragen: Generate data for the terasort
  terasort: Run the terasort
  teravalidate: Checking results of terasort
  wordcount: A map/reduce program that counts the words in the input files.
```



图 3-10

如何执行这个程序哪?见图 3-11, 如下

```
[root@hadoop0 hadoop]# hadoop jar hadoop-examples-1.1.2.jar wordcount
Warning: $HADOOP_HOME is deprecated.


Usage: wordcount <in> <out>  表示后面跟输入路径和输出路径
```

图 3-11

根据用法, 我们需要补全 wordcount 的文件输入路径和文件输出路径. 我们首先上传一个文件到 hdfs 中, 见图 3-12. 那么输入文件的路径就是/README.txt 了。

```
[root@hadoop0 hadoop]# pwd
/usr/local/hadoop
[root@hadoop0 hadoop]# ls
bin                hadoop-examples-1.1.2.jar  NOTICE.txt
build.xml          hadoop-miniclust-1.1.2.jar README.txt
c++               hadoop-test-1.1.2.jar     sbin
CHANGES.txt      hadoop-tools-1.1.2.jar   share
conf              ivy                       src
contrib           ivy.xml                  TestDFSIO_results.log
docs              lib                       tmp
hadoop-ant-1.1.2.jar libexec                   webapps
hadoop-client-1.1.2.jar LICENSE.txt
hadoop-core-1.1.2.jar logs
[root@hadoop0 hadoop]# hadoop fs -put README.txt /
Warning: $HADOOP_HOME is deprecated.
```

图 3-12

这时候我们再运行 wordcount 例子, 命令如下

```
hadoop jar hadoop-examples-1.1.2.jar wordcount
“/” /README.txt /wordcountoutput
```

等到命令执行结束, 运行的结果就会存在输出路径的文件夹中, 文件名称叫做“part-r-00000”, 我们使用命令查看输出内容, 如图 3-13

```
[root@hadoop0 hadoop]# hadoop fs -text /wordcountoutput/part-r-00000
Warning: $HADOOP_HOME is deprecated.

(BIS), 1
(ECCN) 1
(TSU) 1
(see 1
5D002.C.1, 1
740.13) 1
<http://www.wassenaar.org/> 1
Administration 1
Apache 1
BEFORE 1
BIS 1
Bureau 1
Commerce, 1
Commodity 1
Control 1
Core 1
Department 1
```

图 3-13

以上只是显示了一部分。显示结果是按照字符的字段顺序排列的，每一行显示字符及出现次数。

如果读者能够成功运行，那么恭喜你！你看到的就是 hadoop 的 MapReduce 做的事情。通过一个简单的命令，就可以把文件中的单词统计一遍出现次数，还是很有意思的。参加工作后，大家会根据输入文件的内容和类型，写这样的算法程序，产生输出结果。

## 3.2. 分布模式安装

### 3.2.1. 集群的架构和拓扑图

所谓集群环境，通常是局域网中的机器，多台机器为了完成一件事件而互相协作。每台机器称作节点。

搭建集群之前，我们先规划一下集群各节点的功能，如下图：

主机名	机器 IP	用途	描述
hadoop0	192.168.1.240/24	namenode/secondaryNamenode/jobTracker	64 位 rhel6, jdk6IntelC6002*(Xeon E5-2620 2GHz)/64GB/12TB
hadoop1	192.168.1.241/24	datanode/taskTracker	64 位 rhel6, jdk6IntelC6002*(Xeon E5-2620 2GHz)/64GB/12TB

图 3-14

读者在练习的时候，对于主机名和机器 ip 可以自己设定。不过，需要注意的是用途一定要保持一致，方便下面的学习。

### 3.2.2. 安装步骤

可以在原来的伪分布基础上安装。

请在各节点检查以下配置：

- 1) 防火墙是否永久关闭
- 2) 静态 ip 是否设置
- 3) 主机名称是否设置



- 4) /etc/hosts 是否添加了所有节点的 ip 与主机名映射
- 5) 各节点的 ssh 是否可以免密码登录自己的主机名

以上检查如果读者没有通过，请参考前面的配置说明进行。

### 3.2.2.1. 集群间 SSH 免密码登录

搭建 hadoop 集群，需要保证各个节点之间必须是 ssh 免密码登录的。那么，我们在节点 hadoop0 上执行，以下命令

```
ssh-copy-id -I ~/.ssh/id_rsa.pub hadoop1
```

在节点 hadoop1 中执行以下命令

```
ssh-copy-id -I ~/.ssh/id_rsa.pub hadoop0
```

这样，就可以实现两个节点通过 ssh 互相登录时不需要密码了。

### 3.2.2.2. 配置/etc/hosts

把节点 192.168.1.241 的 hostname 修改为 hadoop1.

在节点 hadoop0 的/etc/hosts 文件中新增以下两行：

```
192.168.1.240 hadoop0
192.168.1.241 hadoop1
```

### 3.2.2.3. 在其他节点安装 jdk 和 hadoop

在节点 hadoop0 执行命令如下：

```
scp -rq /usr/local/jdk hadoop1:/usr/local
scp -rq /usr/local/hadoop hadoop1:/usr/local
scp -rq /etc/profile hadoop1:/etc
scp -rq /etc/hosts hadoop1:/etc
ssh hadoop1
source /etc/profile
exit
```

第一条命令的目的是把 jdk 文件夹复制到节点 hadoop1 的/usr/local/目录下；  
第二条命令的目的是把 hadoop 文件夹复制到节点 hadoop1 的/usr/local/目录下；  
第三条命令的目的是把/etc/profile 文件复制到节点 hadoop1 的/etc/目录下；  
第四条命令的目的是把/etc/hosts 文件复制到节点 hadoop1 的/etc/目录下；

#### 3.2.2.4. 配置集群核心文件

hadoop 的配置文件 `slaves`，位于 `conf` 目录下，里面存储着 `datanode` 和 `tasktracker` 运行的节点名称。现在修改该文件，把内容 `localhost` 删掉，增加 `hadoop1`。如下：

```
hadoop1
```

修改后，意味着在节点 `hadoop1` 运行 `datanode` 和 `tasktracker` 节点。

#### 3.2.2.5. 格式化文件系统

把新安装的 hadoop 系统，理解为买了一块新硬盘。我们新买的硬盘，只有格式化才可以使用。hadoop 的文件系统也是如此。

在节点 `hadoop0` 中执行命令：`hadoop namenode -format`。

格式化整个 hadoop 集群的文件系统。

#### 3.2.2.6. 启动集群

在节点 `hadoop0` 中执行命令：`start-all.sh`

#### 3.2.2.7. 验证

虽然我们执行命令“`start-all.sh`”，可以看到很多输出信息，但是并不意味着，hadoop 集群启动成功了，只是表示开始启动了，是否启动成功，需要我们自己判断一下。

在节点 `hadoop0` 中执行 `jps`，可以观察到 `NameNode`、`SecondaryNameNode`、`JobTracker` 三个 `java` 进程。

在节点 `hadoop1` 中执行 `jps`，可以观察到 `DataNode`、`TaskTracker` 两个 `java` 进程。

如果看到这些信息，表明确实启动成功了。

### 3.3. Hadoop 的常见启动错误

当使用 `hadoop` 发现有问题时，首先使用 `jps` 命令查看启动的节点是否正确，然后再去查看日志文件。

#### ● 设置主机名错误

看日志，会发现下面的错误

```
ERROR org.apache.hadoop.hdfs.server.namenode.NameNode: java.net.UnknownHostException:
Invalid hostname for server: master
```

这是由于主机名设置错误造成的，请检查配置文件中关于主机名的设置，是否正确。

首先使用 `hostname` 命令查看主机名是否正确；

然后使用 `more /etc/sysconfig/network` 命令查看主机名是否记录在文件中；

最后使用 `more /etc/hosts` 命令查看 ip 与主机名的映射是否设置。

## ● ip 设置错误

看日志，发现下面的错误

```
ERROR org.apache.hadoop.hdfs.server.namenode.NameNode: java.net.BindException: Problem binding to book0/192.168.1.100:9000 : Cannot assign requested address
```

这是由于 ip 地址设置错误引起的，请检查主机的 ip 设置与配置文件的 ip 设置是否一致。

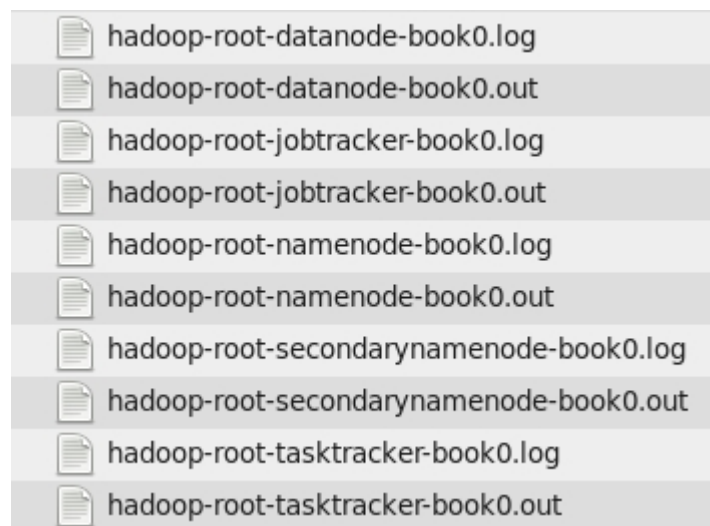
使用 `more /etc/hosts` 命令查看 ip 与主机名的映射是否设置。

## ● hostname 不能带下划线，不能数字开头

`hostname` 含有下划线，也会造成启动失败。

启动完成后，一定使用 `jps` 命令查看 5 个节点是否全部成功启动。如果哪个节点没有启动，请查看对应的 log 日志。日志的默认目录是 `$HADOOP_HOME/logs`。

以下是作者的日志截图



文件命名是有规律的，“`hadoop-[当前用户名]-[节点名称]-[主机名].log`”我们只查看 log 结尾的文件。如果是 namenode 节点没有启动，查看 `hadoop-root-namenode-book0.log` 文件。

## ● 多次执行 hadoop 格式化

现象：在 `Hadoop-root-datanode-master.log` 中有如下错误：

```
ERROR org.apache.hadoop.hdfs.server.datanode.DataNode: java.io.IOException: Incompatible namespaceIDs in
```

原因：每次 namenode format 会重新创建一个 namenodeId, 而 dfs.data.dir 参数配置的目录中包含的是上次 format 创建的 id, 和 dfs.name.dir 参数配置的目录中的 id 不一致。namenode format 清空了 namenode 下的数据, 但是没有清空 datanode 下的数据, 导致启动时失败, 所要做的就是每次 format 前, 清空 dfs.data.dir 参数配置的目录。重新格式化 hdfs 的命令。

## ● 防火墙没有关闭

从本地往 hdfs 文件系统上传文件, 出现如下错误:

```
[root@hadoop0 hadoop]# hadoop fs -put CHANGES.txt /wordcount
Warning: $HADOOP_HOME is deprecated.
```

```
13/07/16 14:59:39 WARN hdfs.DFSClient: DataStreamer Exception: org.apache.hadoop.ipc.RemoteException: java.io.IOException: File /wordcount/CHANGES.txt does not exist.
o 0 nodes, instead of 1
    at org.apache.hadoop.hdfs.server.namenode.FSNamesystem.getAdditionalBlock(FSNamesystem.java:1639)
    at org.apache.hadoop.hdfs.server.namenode.NameNode.addBlock(NameNode.java:736)
    at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
    at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:39)
    at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:25)
    at java.lang.reflect.Method.invoke(Method.java:597)
    at org.apache.hadoop.ipc.RPC$Server.call(RPC.java:578)
    at org.apache.hadoop.ipc.Server$Handler$1.run(Server.java:1393)
    at org.apache.hadoop.ipc.Server$Handler$1.run(Server.java:1389)
    at java.security.AccessController.doPrivileged(Native Method)
    at javax.security.auth.Subject.doAs(Subject.java:396)
    at org.apache.hadoop.security.UserGroupInformation.doAs(UserGroupInformation.java:1149)
    at org.apache.hadoop.ipc.Server$Handler.run(Server.java:1387)

    at org.apache.hadoop.ipc.Client.call(Client.java:1107)
    at org.apache.hadoop.ipc.RPC$Invoker.invoke(RPC.java:229)
    at $Proxy1.addBlock(Unknown Source)
```

这个问题最有可能就是防火墙没有关闭, 导致节点 DataNode 与节点 NameNode 通信失败。可以使用 service iptables status 命令关闭防火墙。

这样关闭后, 重启操作系统时, 防火墙有可能重启, 可以关闭自动重启功能。使用 chkconfig iptables off 关闭。

## ● 安全模式导致的错误

错误提示如下:

```
org.apache.hadoop.dfs.SafeModeException: Cannot delete ..., Name node is in
safe mode
```

在分布式文件系统启动的时候, 开始的时候会有安全模式, 当分布式文件系统处于安全模式的情况下, 文件系统中的内容不允许修改也不允许删除, 直到安全模式结束。安全模式

主要是为了系统启动的时候检查各个 DataNode 上数据块的有效性，同时根据策略必要的复制或者删除部分数据块。运行期通过命令也可以进入安全模式。在实践中，系统启动的时候去修改和删除文件也会有安全模式不允许修改的出错提示，只需要等待一会儿即可。

——如果着急的话，可以执行 `hadoop dfsadmin -safemode leave` 命令关闭安全模式。

### 3.4. 本章小结

