

# 给力星 (<http://www.powerxing.com/>)

追逐内心的平和

首页 (<http://www.powerxing.com/>)

■ 笔记 (<http://www.powerxing.com/notes/>)

搜藏 (<http://www.powerxing.com/articles/>)

代码 (<http://www.powerxing.com/snippets/>)

音乐 (<http://www.powerxing.com/music/>)

关于 (<http://www.powerxing.com/about/>)

搜索

## Hadoop安装教程\_单机/伪分布式配置\_CentOS6.4/Hadoop2.6.0

📅 2015-12-17 (updated: 2016-01-06) 👁 922

本 Hadoop 教程由厦门大学数据库实验室 (<http://dblab.xmu.edu.cn>)出品，转载请注明。本教程适合于在 CentOS 6.x 系统中安装原生 Hadoop 2，适用于Hadoop 2.7.1, Hadoop 2.6.0 等版本，主要参考了官方安装教程 (<http://hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-common/SingleCluster.html>)，步骤详细，辅以适当说明，相信按照步骤来，都能顺利在 **CentOS** 中安装并运行 **Hadoop**。

## 环境

本教程使用 **CentOS 6.4 32位** 作为系统环境，请自行安装系统（可参考使用VirtualBox安装CentOS (<http://dblab.xmu.edu.cn/blog/164/>)）。如果用的是 Ubuntu 系统，请查看相应的Ubuntu安装Hadoop教程 (<http://www.powerxing.com/install-hadoop/>)。

本教程基于原生 Hadoop 2，在 **Hadoop 2.6.0 (stable)** 版本下验证通过，可适合任何 Hadoop 2.x.y 版本，例如 Hadoop 2.7.1, Hadoop 2.4.1等。

## Hadoop版本

Hadoop 有两个主要版本，Hadoop 1.x.y 和 Hadoop 2.x.y 系列，比较老的教材上用的可能是 0.20 这样的版本。Hadoop 2.x 版本在不断更新，本教程均可适用。如果需安装 0.20, 1.2.1这样的版本，本教程也可以作为参考，主要差别在于配置项，配置请参考官网教程或其他教程。

新版是兼容旧版的，书上旧版本的代码应该能够正常运行（我自己没验证，欢迎验证反馈）。

装好了 CentOS 系统之后，在安装 Hadoop 前还需要做一些必备工作。

## 创建hadoop用户

如果你安装 CentOS 的时候不是用的“hadoop”用户，那么需要增加一个名为 hadoop 的用户。

首先点击左上角的“应用程序”->“系统工具”->“终端”，首先在终端中输入 `su`，按回车，输入 root 密码以 root 用户登录，接着执行命令创建新用户 hadoop:

```
$ su                # 上述提到的以 root 用户登录
$ useradd -m hadoop -s /bin/bash # 创建新用户hadoop
```

如下图所示，这条命令创建了可以登陆的 hadoop 用户，并使用 /bin/bash 作为shell。



## CentOS创建hadoop用户

接着使用如下命令修改密码，按提示输入两次密码，可简单的设为“hadoop”（密码随意指定，若提示“无效的密码，过于简单”则再次输入确认就行）：

```
$ passwd hadoop
```

可为 hadoop 用户增加管理员权限，方便部署，避免一些对新手来说比较棘手的权限问题，执行：

```
$ visudo
```

如下图，找到 `root ALL=(ALL) ALL` 这行（应该在第98行，可以先按一下键盘上的 `ESC` 键，然后输入 `:98`（按一下冒号，接着输入98，再按回车键），可以直接跳到第98行），然后在这行下面增加一行内容：`hadoop ALL=(ALL) ALL`（当中的间隔为tab），如下图所示：



为hadoop增加sudo权限

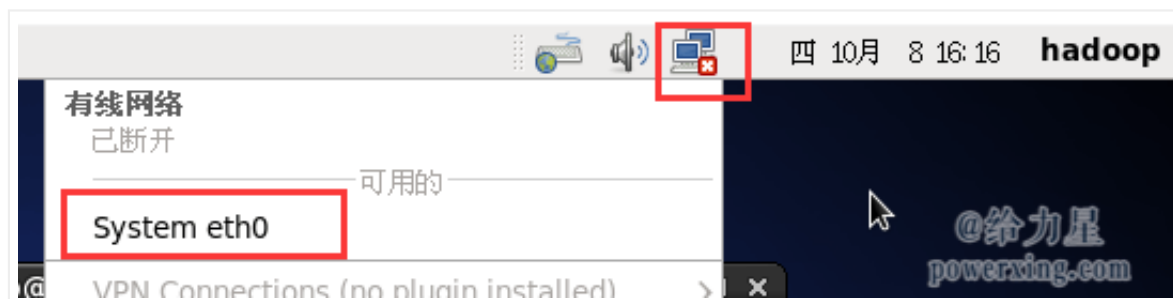
添加好内容后，先按一下键盘上的 `ESC` 键，然后输入 `:wq`（输入冒号还有wq，这是vi/vim编辑器的保存方法），再按回车键保存退出就可以了。

最后注销当前用户(点击屏幕右上角的用户名，选择退出->注销)，在登陆界面使用刚创建的 hadoop 用户进行登陆。

## 准备工作

使用 hadoop 用户登录后，还需要安装几个软件才能安装 Hadoop。

CentOS 使用 yum 来安装软件，需要联网环境，首先应检查一下是否连上了网络。如下图所示，桌面右上角的网络图标若显示红叉，则表明还未联网，应点击选择可用网络。



检查是否联网

连接网络后，需要安装 SSH 和 Java。

## 安装SSH、配置SSH无密码登陆

集群、单节点模式都需要用到 SSH 登陆（类似于远程登陆，你可以登录某台 Linux 主机，并且在上面运行命令），一般情况下，CentOS 默认已安装了 SSH client、SSH server，打开终端执行如下命令进行检验：

```
$ rpm -qa | grep ssh
```

如果返回的结果如下图所示，包含了 SSH client 跟 SSH server，则不需要再安装。



检查是否安装了SSH

若需要安装，则可以通过 yum 进行安装（安装过程中会让你输入 [y/N]，输入 y 即可）：

```
$ sudo yum install openssh-clients
$ sudo yum install openssh-server
```

接着执行如下命令测试一下 SSH 是否可用：

```
$ ssh localhost
```

此时会有如下提示(SSH首次登陆提示), 输入 **yes**。然后按提示输入密码 **hadoop**, 这样就登陆到本机了。

```
[hadoop@dblab ~]$ ssh localhost
The authenticity of host 'localhost (::1)' can't be established.
RSA key fingerprint is 99:90:ab:cf:61:75:1c:99:58:fb:d4:72:67:3c:4b:63.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'localhost' (RSA) to the list of known hosts.
hadoop@localhost's password:
```

测试SSH是否可用

但这样登陆是需要每次输入密码的, 我们需要配置成SSH无密码登陆比较方便。

首先输入 **exit** 退出刚才的 **ssh**, 就回到了我们原先的终端窗口, 然后利用 **ssh-keygen** 生成密钥, 并将密钥加入到授权中:

```
$ exit # 退出刚才的 ssh localhost
$ cd ~/.ssh/ # 若没有该目录, 请先执行一次ssh loc
alhost
$ ssh-keygen -t rsa # 会有提示, 都按回车就可以
$ cat id_rsa.pub >> authorized_keys # 加入授权
$ chmod 600 ./authorized_keys # 修改文件权限
```

## ~的含义

在 Linux 系统中, ~ 代表的是用户的主文件夹, 即 “/home/用户名” 这个目录, 如你的用户名为 **hadoop**, 则 ~ 就代表 “/home/hadoop/”。此外, 命令中的 # 后面的文字是注释。

此时再用 **ssh localhost** 命令, 无需输入密码就可以直接登陆了, 如下图所示。

```
hadoop@dblab:~/.ssh
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
[hadoop@dblab .ssh]$ ssh localhost
Last login: Thu Oct  8 19:02:06 2015 from localhost
[hadoop@dblab ~]$
[hadoop@dblab ~]$ exit
logout
Connection to localhost closed.
```

## 安装Java环境

Java 环境可选择 Oracle 的 JDK，或是 OpenJDK，现在一般 Linux 系统默认安装的基本是 OpenJDK，如 CentOS 6.4 就默认安装了 OpenJDK 1.7。按

<http://wiki.apache.org/hadoop/HadoopJavaVersions>

(<http://wiki.apache.org/hadoop/HadoopJavaVersions>) 中说的，Hadoop 在 OpenJDK 1.7 下运行是没问题的。需要注意的是，CentOS 6.4 中默认安装的只是 Java JRE，而不是 JDK，为了开发方便，我们还是需要通过 yum 进行安装 JDK，安装过程中会让输入 [y/N]，输入 y 即可：

```
$ sudo yum install java-1.7.0-openjdk java-1.7.0-openjdk-devel
```

通过上述命令安装 OpenJDK，默认安装位置为 /usr/lib/jvm/java-1.7.0-openjdk（该路径可以通过执行 `rpm -ql java-1.7.0-openjdk-devel | grep '/bin/javac'` 命令确定，执行后会输出一个路径，除去路径末尾的 “/bin/javac”，剩下的就是正确的路径了）。

OpenJDK 安装后就可以直接使用 java、javac 等命令了。

接着需要配置一下 JAVA\_HOME 环境变量，为方便，我们在 ~/.bashrc 中进行设置（扩展阅读：设置Linux环境变量的方法和区别 (<http://www.powerxing.com/linux-environment-variable/>)）：

```
$ vim ~/.bashrc
```

在文件最后面添加如下单独一行（指向 JDK 的安装位置），并保存：

```
export JAVA_HOME=/usr/lib/jvm/java-1.7.0-openjdk
```

如下图所示：



```
hadoop@dblab:~  
窗口菜单 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)  
# .bashrc  
  
# Source global definitions  
if [ -f /etc/bashrc ]; then  
    . /etc/bashrc  
fi  
  
# User specific aliases and functions  
export JAVA_HOME=/usr/lib/jvm/java-1.7.0-openjdk  
~
```

@给力星  
powerxing.com

设置JAVA\_HOME环境变量


接着还需要让该环境变量生效，执行如下代码：

```
$ source ~/.bashrc    # 使变量设置生效
```

设置好后我们来检验一下是否设置正确：

```
$ echo $JAVA_HOME      # 检验变量值  
$ java -version  
$ $JAVA_HOME/bin/java -version  # 与直接执行 java -version 一样
```

如果设置正确的话，`$JAVA_HOME/bin/java -version` 会输出 java 的版本信息，且和 `java -version` 的输出结果一样，如下图所示：



```
hadoop@dblab:~  
文件(E) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)  
[hadoop@dblab ~]$ vim ~/.bashrc  
[hadoop@dblab ~]$ source ~/.bashrc  
[hadoop@dblab ~]$ echo $JAVA_HOME  
/usr/lib/jvm/java-1.7.0-openjdk  
[hadoop@dblab ~]$ java -version  
java version "1.7.0_91"  
OpenJDK Runtime Environment (rhel-2.6.2.2.el6_7-i386 u91-b00)  
OpenJDK Client VM (build 24.91-b01, mixed mode, sharing)  
[hadoop@dblab ~]$ $JAVA_HOME/bin/java -version  
java version "1.7.0_91"  
OpenJDK Runtime Environment (rhel-2.6.2.2.el6_7-i386 u91-b00)  
OpenJDK Client VM (build 24.91-b01, mixed mode, sharing)  
[hadoop@dblab ~]$
```

@给力星  
powerxing.com

成功设置JAVA\_HOME环境变量

这样，Hadoop 所需的 Java 运行环境就安装好了。



# 安装 Hadoop 2

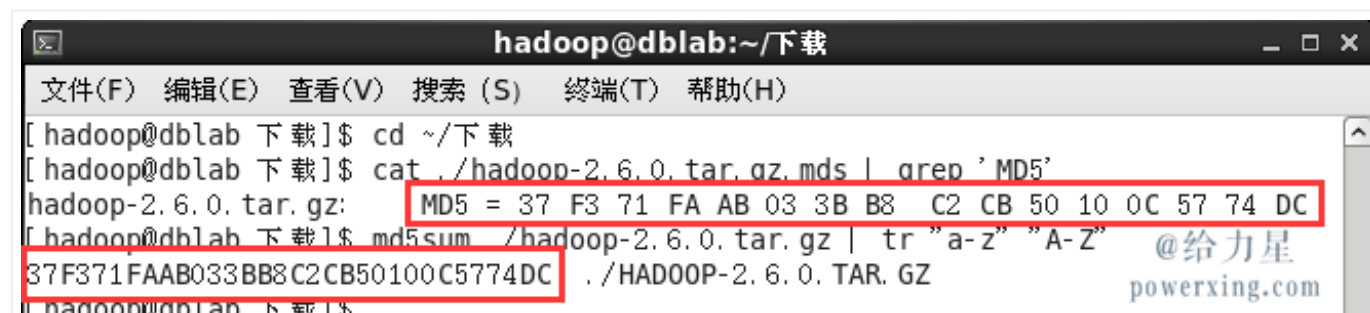
Hadoop 2 可以通过 <http://mirror.bit.edu.cn/apache/hadoop/common/> (<http://mirror.bit.edu.cn/apache/hadoop/common/>) 或者 <http://mirrors.cnnic.cn/apache/hadoop/common/> (<http://mirrors.cnnic.cn/apache/hadoop/common/>) 下载，本教程选择的是 2.6.0 版本，下载时请下载 **hadoop-2.x.y.tar.gz** 这个格式的文件，这是编译好的，另一个包含 src 的则是 Hadoop 源代码，需要进行编译才可使用。

下载时强烈建议也下载 **hadoop-2.x.y.tar.gz.mds** 这个文件，该文件包含了检验值可用于检查 **hadoop-2.x.y.tar.gz** 的完整性，否则若文件发生了损坏或下载不完整，Hadoop 将无法正常运行。

本文涉及的文件均通过浏览器下载，默认保存在“下载”目录中（若不是请自行更改 tar 命令的相应目录）。另外，如果你用的不是 2.6.0 版本，则将所有命令中出现的 2.6.0 更改为你所使用的版本。

```
$ cat ~/下载/hadoop-2.6.0.tar.gz.mds | grep 'MD5' # 列出md5检验值
$ # head -n 6 ~/下载/hadoop-2.7.1.tar.gz.mds # 2.7.1版本格式变了，
  可以用这种方式输出
$ md5sum ~/下载/hadoop-2.6.0.tar.gz | tr "a-z" "A-Z" # 计算md5
  值，并转化为大写，方便比较
```

若文件不完整则这两个值一般差别很大，可以简单对比下前几个字符跟后几个字符是否相等即可，如下图所示，如果两个值不一样，请务必重新下载。



```
hadoop@dblab:~/下载
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
[hadoop@dblab 下载]$ cd ~/下载
[hadoop@dblab 下载]$ cat ./hadoop-2.6.0.tar.gz.mds | grep 'MD5'
hadoop-2.6.0.tar.gz: MD5 = 37 F3 71 FA AB 03 3B B8 C2 CB 50 10 0C 57 74 DC
[hadoop@dblab 下载]$ md5sum ./hadoop-2.6.0.tar.gz | tr "a-z" "A-Z"
37F371FAAB033BB8C2CB50100C5774DC ./HADOOP-2.6.0.TAR.GZ
```

检验文件完整性

我们选择将 Hadoop 安装至 `/usr/local/` 中：



```
$ sudo tar -zxf ~/下载/hadoop-2.6.0.tar.gz -C /usr/local # 解
压到/usr/local中
$ cd /usr/local/
$ sudo mv ./hadoop-2.6.0/ ./hadoop # 将文件夹名改为had
oop
$ sudo chown -R hadoop:hadoop ./hadoop # 修改文件权限
```

Hadoop 解压后即可使用。输入如下命令来检查 Hadoop 是否可用，成功则会显示 Hadoop 版本信息：

```
$ cd /usr/local/hadoop
$ ./bin/hadoop version
```

## 相对路径与绝对路径的区别

请务必注意命令中的相对路径与绝对路径，本文后续出现的

`./bin/...`，`./etc/...` 等包含 `./` 的路径，均为相对路径，以 `/usr/local/hadoop` 为当前目录。例如在 `/usr/local/hadoop` 目录中执行 `./bin/hadoop version` 等同于执行 `/usr/local/hadoop/bin/hadoop version`。可以将相对路径改成绝对路径来执行，但如果你是在主文件夹 `~` 中执行 `./bin/hadoop version`，执行的会是 `/home/hadoop/bin/hadoop version`，就不是我们所想要的了。

# Hadoop单机配置(非分布式)

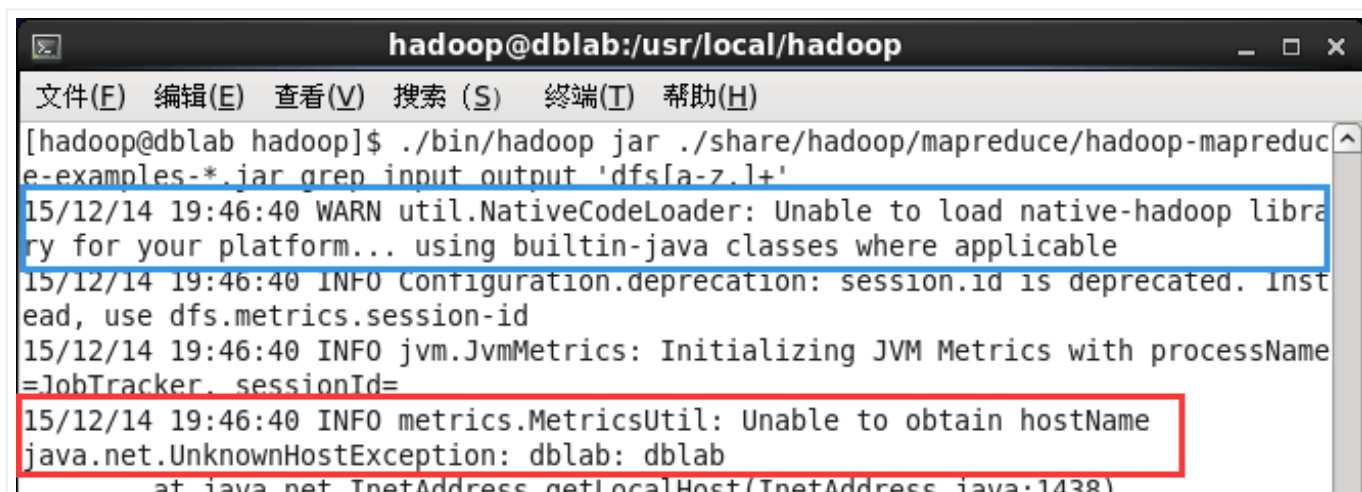
Hadoop 默认模式为非分布式模式，无需进行其他配置即可运行。非分布式即单 Java 进程，方便进行调试。

现在我们可以执行例子来感受下 Hadoop 的运行。Hadoop 附带了丰富的例子（运行 `./bin/hadoop jar ./share/hadoop/mapreduce/hadoop-mapreduce-examples-2.6.0.jar` 可以看到所有例子），包括 `wordcount`、`terasort`、`join`、`grep` 等。

在此我们选择运行 `grep` 例子，我们将 `input` 文件夹中的所有文件作为输入，筛选当中符合正则表达式 `dfs[a-z.]+` 的单词并统计出现的次数，最后输出结果到 `output` 文件夹中。

```
$ cd /usr/local/hadoop
$ mkdir ./input
$ cp ./etc/hadoop/*.xml ./input # 将配置文件作为输入文件
$ ./bin/hadoop jar ./share/hadoop/mapreduce/hadoop-mapreduce-examples-*.jar grep ./input ./output 'dfs[a-z.]+'
$ cat ./output/* # 查看运行结果
```

若运行出错，如出现如下图提示：

A terminal window titled 'hadoop@dblab:usr/local/hadoop' showing the execution of a Hadoop command. The command is './bin/hadoop jar ./share/hadoop/mapreduce/hadoop-mapreduce-examples-\*.jar grep input output 'dfs[a-z.]+'. The output shows a warning: 'WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable'. Below this, there is an information message about session.id being deprecated. At the bottom, there is an error: 'INFO metrics.MetricsUtil: Unable to obtain hostName java.net.UnknownHostException: dblab: dblab'. The error message is highlighted with a red box. The warning message is highlighted with a blue box.

```
hadoop@dblab:usr/local/hadoop
文件(E) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
[hadoop@dblab hadoop]$ ./bin/hadoop jar ./share/hadoop/mapreduce/hadoop-mapreduce-examples-*.jar grep input output 'dfs[a-z.]+'
15/12/14 19:46:40 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
15/12/14 19:46:40 INFO Configuration.deprecation: session.id is deprecated. Instead, use dfs.metrics.session-id
15/12/14 19:46:40 INFO jvm.JvmMetrics: Initializing JVM Metrics with processName=JobTracker, sessionId=
15/12/14 19:46:40 INFO metrics.MetricsUtil: Unable to obtain hostName
java.net.UnknownHostException: dblab: dblab
    at java.net.InetAddress.getLocalHost(InetAddress.java:1438)
```

运行Hadoop实例时可能会报错

若出现提示 “WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable”，该 WARN 提示可以忽略，不会影响 Hadoop 正常运行（可通过编译 Hadoop 源码解决，解决方法请自行搜索）。

若出现提示 “INFO metrics.MetricsUtil: Unable to obtain hostName java.net.UnknownHostException”，这需要执行如下命令修改 hosts 文件，为你的主机名增加IP映射：

```
$ sudo vim /etc/hosts
```

主机名在终端窗口标题里可以看到，或执行命令 `hostname` 查看，如下图所示，在最后面增加一行 “127.0.0.1 dblab”：

A terminal window titled 'hadoop@dblab:~' showing the contents of the /etc/hosts file. The file contains three lines: '127.0.0.1 localhost localhost.localdomain localhost4 localhost4.localdomain4', '::1 localhost localhost.localdomain localhost6 localhost6.localdomain6', and '127.0.0.1 dblab'. The last line is highlighted with a red box. A red arrow points from the terminal title 'hadoop@dblab' to the 'dblab' entry in the hosts file. The terminal title is also highlighted with a red box.

```
hadoop@dblab:~
文件(E) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
127.0.0.1 localhost localhost.localdomain localhost4 localhost4.localdomain4
::1 localhost localhost.localdomain localhost6 localhost6.localdomain6
127.0.0.1 dblab
```

## 设置主机名的IP映射

保存文件后，重新运行 **hadoop** 实例，若执行成功的话会输出很多作业的相关信息，最后的输出信息如下图所示。作业的结果会输出在指定的 **output** 文件夹中，通过命令 **cat ./output/\*** 查看结果，符合正则的单词 **dfsadmin** 出现了1次：



```
hadoop@dblab:/usr/local/hadoop
文件(E) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
IO_ERROR=0
WRONG_LENGTH=0
WRONG_MAP=0
WRONG_REDUCE=0
File Input Format Counters
  Bytes Read=123
File Output Format Counters
  Bytes Written=23
[hadoop@dblab hadoop]$ cat ./output/*
1 dfsadmin
```

Hadoop例子输出结果

注意，Hadoop 默认不会覆盖结果文件，因此再次运行上面实例会提示出错，需要先将 **./output** 删除。

```
$ rm -r ./output
```

## Hadoop伪分布式配置

Hadoop 可以在单节点上以伪分布式的方式运行，Hadoop 进程以分离的 Java 进程来运行，节点既作为 NameNode 也作为 DataNode，同时，读取的是 HDFS 中的文件。

在设置 Hadoop 伪分布式配置前，我们还需要设置 HADOOP 环境变量，执行如下命令在 **~/.bashrc** 中设置：

```
$ gedit ~/.bashrc
```

这次我们选择用 **gedit** 而不是 **vim** 来编辑。**gedit** 是文本编辑器，类似于 Windows 中的记事本，会比较方便。保存后记得关掉整个 **gedit** 程序，否则会占用终端。在文件最后面增加如下内容：

```
# Hadoop Environment Variables
export HADOOP_HOME=/usr/local/hadoop
export HADOOP_INSTALL=$HADOOP_HOME
export HADOOP_MAPRED_HOME=$HADOOP_HOME
export HADOOP_COMMON_HOME=$HADOOP_HOME
export HADOOP_HDFS_HOME=$HADOOP_HOME
export YARN_HOME=$HADOOP_HOME
export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/lib/native
export PATH=$PATH:$HADOOP_HOME/sbin:$HADOOP_HOME/bin
```

保存后，不要忘记执行如下命令使配置生效：

```
$ source ~/.bashrc
```

这些变量在启动 Hadoop 进程时需要用到，不设置的话可能会报错（这些变量也可以通过修改 `./etc/hadoop/hadoop-env.sh` 实现）。

Hadoop 的配置文件位于 `/usr/local/hadoop/etc/hadoop/` 中，伪分布式需要修改2个配置文件 **core-site.xml** 和 **hdfs-site.xml**。Hadoop的配置文件是 xml 格式，每个配置以声明 property 的 name 和 value 的方式来实现。

修改配置文件 **core-site.xml** (通过 gedit 编辑会比较方便: `gedit ./etc/hadoop/core-site.xml`)，将当中的

```
<configuration>
</configuration>
```

修改为下面配置：

```
<configuration>
  <property>
    <name>hadoop.tmp.dir</name>
    <value>file:/usr/local/hadoop/tmp</value>
    <description>Abase for other temporary directories.</de
scription>
  </property>
  <property>
    <name>fs.defaultFS</name>
    <value>hdfs://localhost:9000</value>
  </property>
</configuration>
```

同样的，修改配置文件 **hdfs-site.xml**：

```
<configuration>
  <property>
    <name>dfs.replication</name>
    <value>1</value>
  </property>
  <property>
    <name>dfs.namenode.name.dir</name>
    <value>file:/usr/local/hadoop/tmp/dfs/name</value>
  </property>
  <property>
    <name>dfs.datanode.data.dir</name>
    <value>file:/usr/local/hadoop/tmp/dfs/data</value>
  </property>
</configuration>
```

配置完成后，执行 NameNode 的格式化：

```
$ ./bin/hdfs namenode -format
```

成功的话，会看到 “successfully formatted” 和 “Exitting with status 0” 的提示，若为 “Exitting with status 1” 则是出错。

A terminal window titled 'hadoop@dblab:/usr/local/hadoop' showing the output of the 'hadoop fs -format' command. The output includes log messages from 'common.Storage', 'namenode.NNStorageRetentionManager', and 'util.ExitUtil'. Three lines are highlighted with red boxes: 'successfully formatted.', 'Exiting with status 0', and 'SHUTDOWN\_MSG: Shutting down NameNode at dblab/127.0.0.1'. A watermark '@给力星 powerxing.com' is visible in the bottom right corner.

```
hadoop@dblab:/usr/local/hadoop
文件(E) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
15/12/14 20:15:25 INFO common.Storage: Storage directory /usr/local/hadoop/tmp/d
fs/name has been successfully formatted.
15/12/14 20:15:25 INFO namenode.NNStorageRetentionManager: Going to retain 1 ima
ges with txid >= 0
15/12/14 20:15:25 INFO util.ExitUtil: Exiting with status 0
15/12/14 20:15:25 INFO namenode.NameNode: SHUTDOWN_MSG:
/*****
SHUTDOWN_MSG: Shutting down NameNode at dblab/127.0.0.1
*****/
@给力星
powerxing.com
```

执行NameNode格式化

接着开启 NameNode 和 DataNode 守护进程:

```
$ ./sbin/start-dfs.sh
```

若出现如下 SSH 的提示 “Are you sure you want to continue connecting”, 输入 yes 即可。

A terminal window titled 'hadoop@dblab:/usr/local/hadoop' showing the output of the 'start-dfs.sh' script. The output includes messages for starting namenodes, datanodes, and secondary namenodes. A red box highlights the SSH warning message: 'The authenticity of host '0.0.0.0 (0.0.0.0)' can't be established. RSA key fingerprint is bc:9c:bb:78:f9:47:71:11:d8:3b:4d:98:8d:71:7e:be.' followed by the prompt 'Are you sure you want to continue connecting (yes/no)?' and the user input 'yes'. A watermark '@给力星 powerxing.com' is visible in the bottom right corner.

```
hadoop@dblab:/usr/local/hadoop
文件(E) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
Starting namenodes on [localhost]
localhost: starting namenode, logging to /usr/local/hadoop/logs/hadoop-hadoop-na
menode-dblab.out
localhost: starting datanode, logging to /usr/local/hadoop/logs/hadoop-hadoop-da
tanode-dblab.out
Starting secondary namenodes [0.0.0.0]
The authenticity of host '0.0.0.0 (0.0.0.0)' can't be established.
RSA key fingerprint is bc:9c:bb:78:f9:47:71:11:d8:3b:4d:98:8d:71:7e:be.
Are you sure you want to continue connecting (yes/no)? yes
@给力星
powerxing.com
```

首次启动Hadoop时的SSH提示

启动时可能会有 WARN 提示 “WARN util.NativeCodeLoader...” 如前面提到的, 这个提示不会影响正常使用。

启动完成后, 可以通过命令 `jps` 来判断是否成功启动, 若成功启动则会列出如下进程: “NameNode”、“DataNode”和 SecondaryNameNode (如果 SecondaryNameNode 没有启动, 请运行 `sbin/stop-dfs.sh` 关闭进程, 然后再次尝试启动尝试)。如果没有 NameNode 或 DataNode, 那就是配置不成功, 请仔细检查之前步骤, 或通过查看启动日志排查原因。



A terminal window titled 'hadoop@dblab:/usr/local/hadoop' with a menu bar containing '文件(E)', '编辑(E)', '查看(V)', '搜索(S)', '终端(T)', and '帮助(H)'. The terminal output shows the command '[hadoop@dblab hadoop]\$ jps' and its results: '13437 NameNode', '13833 Jps', '13721 SecondaryNameNode', and '13563 DataNode'. A watermark '@给力星 powerxing.com' is visible in the bottom right corner of the terminal window.

```
hadoop@dblab:/usr/local/hadoop
文件(E) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
[hadoop@dblab hadoop]$ jps
13437 NameNode
13833 Jps
13721 SecondaryNameNode
13563 DataNode
[hadoop@dblab hadoop]$
```

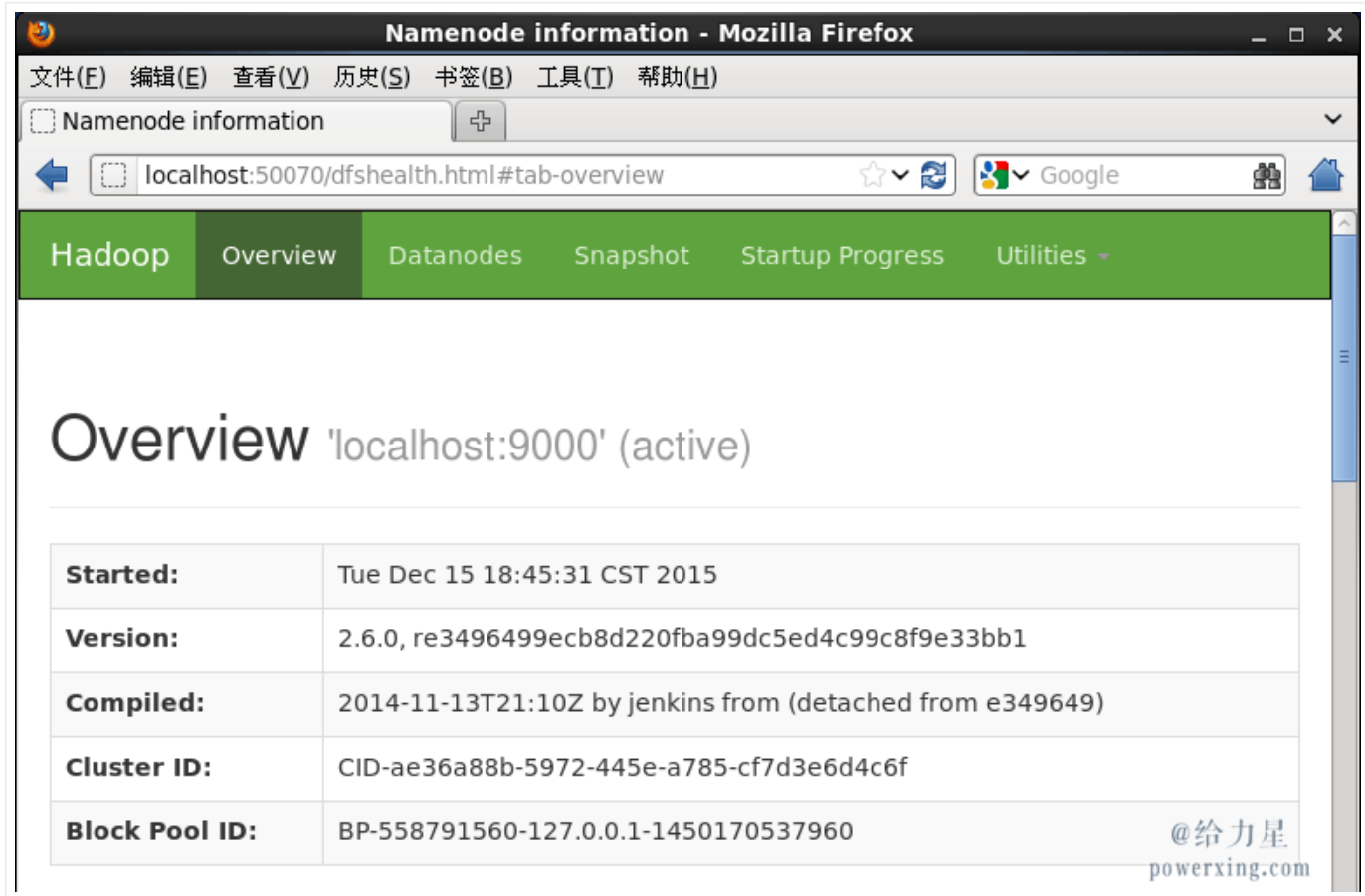
通过jps查看启动的Hadoop进程

## 通过查看启动日志分析启动失败原因

有时 Hadoop 无法正确启动，如 NameNode 进程没有顺利启动，这时可以查看启动日志来排查原因，注意几点：

- 启动时会提示形如 “dblab: starting namenode, logging to /usr/local/hadoop/logs/hadoop-hadoop-namenode-dblab.out”，其中 dblab 对应你的主机名，但启动的日志信息是记录在 /usr/local/hadoop/logs/hadoop-hadoop-namenode-dblab.log 中，所以应该查看这个后缀为 **.log** 的文件；
- 每一次的启动日志都是追加在日志文件之后，所以得拉到最后面看，看下记录的时间就知道了。
- 一般出错的提示在最后面，也就是写着 Fatal、Error 或者 Java Exception 的地方。
- 可以在网上搜索一下出错信息，看能否找到一些相关的解决方法。

成功启动后，可以访问 Web 界面 <http://localhost:50070> (<http://localhost:50070>) 查看 NameNode 和 Datanode 信息，还可以在线查看 HDFS 中的文件。



Hadoop的Web界面

## 运行Hadoop伪分布式实例

上面的单机模式，grep 例子读取的是本地数据，伪分布式读取的则是 HDFS 上的数据。要使用 HDFS，首先需要在 HDFS 中创建用户目录：

```
$ ./bin/hdfs dfs -mkdir -p /user/hadoop
```

接着将 `./etc/hadoop` 中的 xml 文件作为输入文件复制到分布式文件系统中，即将 `/usr/local/hadoop/etc/hadoop` 复制到分布式文件系统 `/user/hadoop/input` 中。我们使用的是 `hadoop` 用户，并且已创建相应的用户目录 `/user/hadoop`，因此在命令中就可以使用相对路径如 `input`，其对应的绝对路径就是 `/user/hadoop/input`：

```
$ ./bin/hdfs dfs -mkdir input
$ ./bin/hdfs dfs -put ./etc/hadoop/*.xml input
```

复制完成后，可以通过如下命令查看 HDFS 中的文件列表：

```
$ ./bin/hdfs dfs -ls input
```

伪分布式运行 MapReduce 作业的方式跟单机模式相同，区别在于伪分布式读取的是HDFS中的文件（可以将单机步骤中创建的本地 input 文件夹，输出结果 output 文件夹都删掉来验证这一点）。

```
$ ./bin/hadoop jar ./share/hadoop/mapreduce/hadoop-mapreduce-examples-*.jar grep input output 'dfs[a-z.]+'
```

查看运行结果的命令（查看的是位于 HDFS 中的输出结果）：

```
$ ./bin/hdfs dfs -cat output/*
```

结果如下，注意到刚才我们已经更改了配置文件，所以运行结果不同。

A terminal window titled 'hadoop@dblab:usr/local/hadoop' showing the execution of 'cat ./output/\*'. The output lists four configuration parameters: 'dfsadmin', 'dfs.replication', 'dfs.namenode.name.dir', and 'dfs.datanode.data.dir', each preceded by a '1'. A watermark '@给力星 powerxing.com' is visible on the right side of the terminal output.

```
hadoop@dblab:usr/local/hadoop
文件(E) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
[hadoop@dblab hadoop]$ cat ./output/*
1      dfsadmin
1      dfs.replication
1      dfs.namenode.name.dir
1      dfs.datanode.data.dir
[hadoop@dblab hadoop]$
```

Hadoop伪分布式运行grep的结果

我们也可以将运行结果取回到本地：

```
$ rm -r ./output      # 先删除本地的 output 文件夹（如果存在）
$ ./bin/hdfs dfs -get output ./output      # 将 HDFS 上的 output
文件夹拷贝到本机
$ cat ./output/*
```

Hadoop 运行程序时，输出目录不能存在，否则会提示错误

“org.apache.hadoop.mapred.FileAlreadyExistsException: Output directory

hdfs://localhost:9000/user/hadoop/output already exists”，因此若要再次执行，需要执行如下命令删除 output 文件夹：

```
$ ./bin/hdfs dfs -rm -r output      # 删除 output 文件夹
```

## 运行程序时，输出目录不能存在

运行 Hadoop 程序时，为了防止覆盖结果，程序指定的输出目录（如 output）不能存在，否则会提示错误，因此运行前需要先删除输出目录。在实际开发应用程序时，可考虑在程序中加上如下代码，能在每次运行时自动删除输出目录，避免繁琐的命令行操作：

```
1. Configuration conf = new Configuration();
2. Job job = new Job(conf);
3.
4. /* 删除输出目录 */
5. Path outputPath = new Path(args[1]);
6. outputPath.getFileSystem(conf).delete(outputPath, true);
```

若要关闭 Hadoop，则运行

```
$ ./sbin/stop-dfs.sh
```

## 注意

下次启动 hadoop 时，无需进行 NameNode 的初始化，只需要运行  
./sbin/start-dfs.sh 就可以！

# 启动YARN

（伪分布式不启动 YARN 也可以，一般不会影响程序执行）

有的读者可能会疑惑，怎么启动 Hadoop 后，见不到书上所说的 JobTracker 和 TaskTracker，这是因为新版的 Hadoop 使用了新的 MapReduce 框架（MapReduce V2，也称为 YARN，Yet Another Resource Negotiator）。

YARN 是从 MapReduce 中分离出来的，负责资源管理与任务调度。YARN 运行于 MapReduce 之上，提供了高可用性、高扩展性，YARN 的更多介绍在此不展开，有兴趣的可查阅相关资料。

上述通过 `./sbin/start-dfs.sh` 启动 Hadoop，仅仅是启动了 MapReduce 环境，我们可以启动 YARN，让 YARN 来负责资源管理与任务调度。

首先修改配置文件 **mapred-site.xml**，这边需要先进行重命名：

```
$ mv ./etc/hadoop/mapred-site.xml.template ./etc/hadoop/mapred-site.xml
```

然后再进行编辑，同样使用 `gedit` 编辑会比较方便些 `gedit ./etc/hadoop/mapred-site.xml`：

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

接着修改配置文件 **yarn-site.xml**：

```
<configuration>
  <property>
    <name>yarn.nodemanager.aux-services</name>
    <value>mapreduce_shuffle</value>
  </property>
</configuration>
```

然后就可以启动 YARN 了（需要先执行过 `./sbin/start-dfs.sh`）：

```
$ ./sbin/start-yarn.sh          $ 启动YARN
$ ./sbin/mr-jobhistory-daemon.sh start historyserver # 开启历史
服务器，才能在Web中查看任务运行情况
```

开启后通过 `jps` 查看，可以看到多了 NodeManager 和 ResourceManager 两个后台进程，如下图所示。

```
hadoop@dblab:/usr/local/hadoop
文件(E) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
[hadoop@dblab hadoop]$ ./sbin/start-yarn.sh
starting yarn daemons
starting resourcemanager, logging to /usr/local/hadoop/logs/yarn-hadoop-resource
manager-dblab.out
localhost: starting nodemanager, logging to /usr/local/hadoop/logs/yarn-hadoop-n
odemanager-dblab.out
[hadoop@dblab hadoop]$ jps
16262 NodeManager
14320 NameNode
16169 ResourceManager
16289 Jps
14446 DataNode
14606 SecondaryNameNode
[hadoop@dblab hadoop]$
```

启动yarn的输出信息

启动成功后多了 ResourceManager 和 NodeManager

@给力星  
powerxing.com

### 开启YARN

启动 YARN 之后，运行实例的方法还是一样的，仅仅是资源管理方式、任务调度不同。观察日志信息可以发现，不启用 YARN 时，是“mapred.LocalJobRunner”在跑任务，启用 YARN 之后，是“mapred.YARNRunner”在跑任务。启动 YARN 有个好处是可以通过 Web 界面查看任务的运行情况：<http://localhost:8088/cluster> (<http://localhost:8088/cluster>)，如下图所示。

cluster

hadoop

### All Applications

Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	VCores Used	VCores Total	VCores Reserved	Active Nodes	Decommissioned Nodes	Lost Nodes
2	0	0	2	0	0 B	8 GB	0 B	0	8	0	1	0	0

Show 20 entries

ID	User	Name	Application Type	Queue	StartTime	FinishTime	State	FinalStatus	Progress	Tracking UI
application_1450255106485_0002	hadoop	grep-sort	MAPREDUCE	default	Wed, 16 Dec 2015	Wed, 16 Dec 2015	FINISHED	SUCCEEDED		History

MapReduce Job job\_1450255106485\_0002 - Mozilla Firefox

Firefox

MapReduce Job job\_145025510...

Logged in as: dt.who

### MapReduce Job job\_1450255106485\_0002

Application

Job

Overview

Counters

Configuration

Map tasks

Reduce tasks

Tools

Job Overview

Job Name: grep-sort

User Name: hadoop

Queue: default

State: SUCCEEDED

Uberized: false

Submitted: Wed Dec 16 16:41:39 CST 2015

Started: Wed Dec 16 16:41:51 CST 2015

Finished: Wed Dec 16 16:42:06 CST 2015

Elapsed: 15sec

Diagnostics:

Average Map Time: 4sec

Average Shuffle Time: 4sec

@给力星  
powerxing.com

开启YARN后可以查看任务运行信息



但 YARN 主要是为集群提供更好的资源管理与任务调度，然而这在单机上体现不出价值，反而会使程序跑得稍慢些。因此在单机上是否开启 YARN 就看实际情况了。

## 不启动 YARN 需重命名 `mapred-site.xml`

如果不想启动 YARN，务必把配置文件 `mapred-site.xml` 重命名，改成 `mapred-site.xml.template`，需要用时改回来就行。否则在该配置文件存在，而未开启 YARN 的情况下，运行程序会提示 “Retrying connect to server: 0.0.0.0/0.0.0.0:8032” 的错误，这也是为何该配置文件初始文件名为 `mapred-site.xml.template`。

同样的，关闭 YARN 的脚本如下：

```
$ ./sbin/stop-yarn.sh
$ ./sbin/mr-jobhistory-daemon.sh stop historyserver
```

自此，你已经掌握 Hadoop 的配置和基本使用了。

## 附加教程：配置 PATH 环境变量

在这里额外讲一下 PATH 这个环境变量（可执行 `echo $PATH` 查看，当中包含了多个目录）。例如我们在主文件夹 `~` 中执行 `ls` 这个命令时，实际执行的是 `/bin/ls` 这个程序，而不是 `~/ls` 这个程序。系统是根据 PATH 这个环境变量中包含的目录位置，逐一进行查找，直至在这些目录位置下找到匹配的程序（若没有匹配的则提示该命令不存在）。

上面的教程中，我们都是先进入到 `/usr/local/hadoop` 目录中，再执行 `./sbin/hadoop`，实际上等同于运行 `/usr/local/hadoop/sbin/hadoop`。我们可以将 Hadoop 命令的相关目录加入到 PATH 环境变量中，这样就可以直接通过 `start-dfs.sh` 开启 Hadoop，也可以直接通过 `hdfs` 访问 HDFS 的内容，方便平时的操作。

在前面我们设置 HADOOP 环境变量时，我们已经顺便设置了 PATH 变量（即 “`export PATH=$PATH:$HADOOP_HOME/sbin:$HADOOP_HOME/bin`”），那么以后我们在任意目录中都可以直接通过执行 `start-dfs.sh` 来启动 Hadoop 或者执行 `hdfs dfs -ls input` 查看 HDFS 文件了，读者不妨现在就执行 `hdfs dfs -ls input` 试试看。

## 安装 Hadoop 集群

在平时的学习中，我们使用伪分布式就足够了。如果需要安装 Hadoop 集群，请查看 Hadoop集群安装配置教程 (<http://www.powerxing.com/install-hadoop-cluster/>)。

## 相关教程

- 使用Eclipse编译运行MapReduce程序 (<http://www.powerxing.com/hadoop-build-project-using-eclipse/>): 用文本编辑器写 Java 程序是不靠谱的，还是用 Eclipse 比较方便。
- 使用命令行编译打包运行自己的MapReduce程序 (<http://www.powerxing.com/hadoop-build-project-by-shell/>): 有时候需要直接通过命令来编译 MapReduce 程序。

## 参考资料

- Hadoop: Setting up a Single Node Cluster (<http://hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-common/SingleCluster.html>)
- How to Setup Hadoop 2.7.1 on CentOS, Ubuntu & LinuxMint (<http://tecadmin.net/setup-hadoop-single-node-cluster-on-centos-redhat/>)
- Yarn简单介绍及内存配置 (<http://blog.chinaunix.net/uid-28311809-id-4383551.html>)

---

 <http://www.powerxing.com/install-hadoop-in-centos/>

(<http://www.powerxing.com/install-hadoop-in-centos/>)

 笔记 (<http://www.powerxing.com/notes/>) 

CentOS (<http://www.powerxing.com/tag/centos/>) ,

Hadoop (<http://www.powerxing.com/tag/hadoop/>)

## 相关文章

➤ **Hadoop安装教程\_单机/伪分布式配置\_Hadoop2.6.0/Ubuntu14.04**

(<http://www.powerxing.com/install-hadoop/>)

➤ **Hadoop集群安装配置教程\_Hadoop2.6.0\_Ubuntu/CentOS**

(<http://www.powerxing.com/install-hadoop-cluster/>)

➤ **使用命令行编译打包运行自己的MapReduce程序 Hadoop2.4.1**

(<http://www.powerxing.com/hadoop-build-project-by-shell/>)

➤ **使用Eclipse编译运行MapReduce程序 Hadoop2.6.0\_Ubuntu/CentOS**

(<http://www.powerxing.com/hadoop-build-project-using-eclipse/>)

➤ **Hadoop-in-Practice第四章MapReduce-Join代码运行问题**

(<http://www.powerxing.com/hadoop-in-practice-joins/>)

➤ **Hadoop安装配置简略教程** (<http://www.powerxing.com/install-hadoop-simplify/>)

.....

0条评论

最新 最早 最热

还没有评论，沙发等你来抢

社交帐号登录:    微信    微博    QQ    人人    [更多»](#)



说点什么吧...

发布

多说 (<http://duoshuo.com>)

© 2014 给力星 (<http://www.powerxing.com>)