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

2014-08-09 (updated: 2017-03-02) 436533 355

当开始着手实践 Hadoop 时，安装 Hadoop 往往会成为新手的一道门槛。尽管安装其实很简单，书上有写到，官方网站也有 Hadoop 安装配置教程，但由于对 Linux 环境不熟悉，书上跟官网上简略的安装步骤新手往往 Hold 不住。加上网上不少教程也甚是坑，导致新手折腾老几天愣是没装好，很是打击学习热情。

本教程适合于原生 Hadoop 2，包括 Hadoop 2.6.0, Hadoop 2.7.1 等版本，主要参考了官方安装教程，步骤详细，辅以适当说明，**相信按照步骤来，都能顺利安装并运行Hadoop**。另外有Hadoop安装配置简略版方便有基础的读者快速完成安装。此外，希望读者们能多去了解一些 Linux 的知识，以后出现问题时才能自行解决。本教程由给力星出品，转载请注明。

环境

本教程使用 **Ubuntu 14.04 64位** 作为系统环境（Ubuntu 12.04 也行，32位、64位均可），请自行安装系统。

如果用的是 CentOS/RedHat 系统，请查看相应的CentOS安装Hadoop教程_单机伪分布式配置。

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 2.4.1。

> **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 Ubuntu 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 Ubuntu 的时候不是用的 “hadoop” 用户，那么需要增加一个名为 hadoop 的用户。

首先按 **ctrl+alt+t** 打开终端窗口，输入如下命令创建新用户：

Shell 命令

```
sudo useradd -m hadoop -s /bin/bash
```

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

> **Ubuntu终端复制粘贴快捷键**
> 在Ubuntu终端窗口中，复制粘贴的快捷键需要加上 shift，即粘贴是 ctrl+shift+v。

接着使用如下命令设置密码，可简单设置为 hadoop，按提示输入两次密码：

Shell 命令

```
sudo 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：

Shell 命令

```
sudo adduser hadoop sudo
```

最后注销当前用户（点击屏幕右上角的齿轮，选择注销），在登陆界面使用刚创建的 hadoop 用户进行登陆。

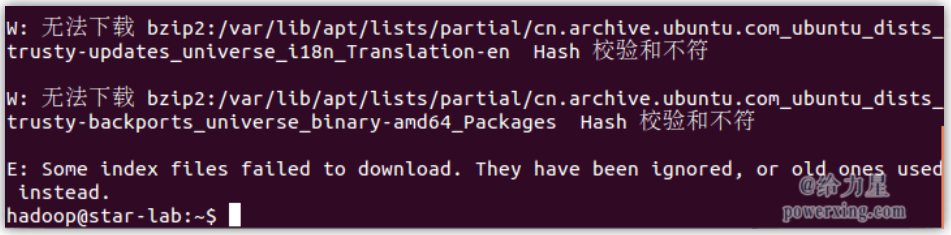
更新apt

用 hadoop 用户登录后，我们先更新一下 apt，后续我们使用 apt 安装软件，如果没更新可能有一些软件安装不了。按 ctrl+alt+t 打开终端窗口，执行如下命令：

Shell 命令

```
sudo apt-get update
```

若出现如下 “Hash校验和不符” 的提示，可通过更改软件源来解决。若没有该问题，则不需要更改。



Ubuntu更新软件源时遇到Hash校验和不符的问题

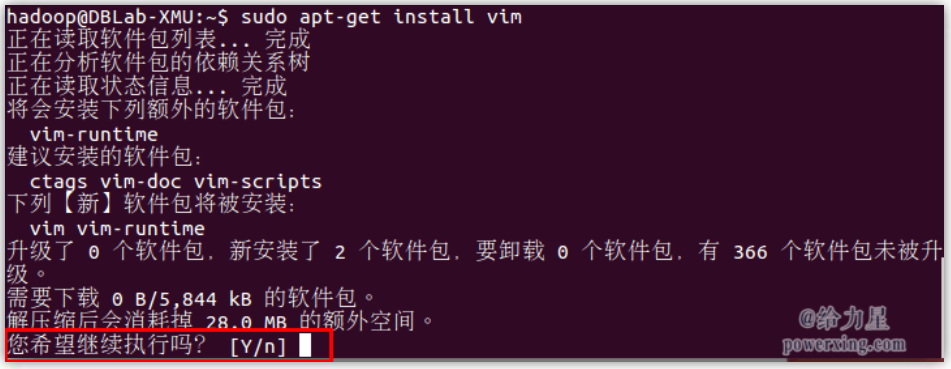
点击查看：如何更改软件源

后续需要更改一些配置文件，我比较喜欢用的是 vim（vi增强版，基本用法相同），建议安装一下（如果你实在还不会用 vi/vim 的，请将后面用到 vim 的地方改为 gedit，这样可以使用文本编辑器进行修改，并且每次文件更改完成后请关闭整个 gedit 程序，否则会占用终端）：

Shell 命令

```
sudo apt-get install vim
```

安装软件时若需要确认，在提示处输入 y 即可。



通过命令行安装软件

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），Ubuntu 默认已安装了 SSH client，此外还需要安装 SSH server：

Shell 命令

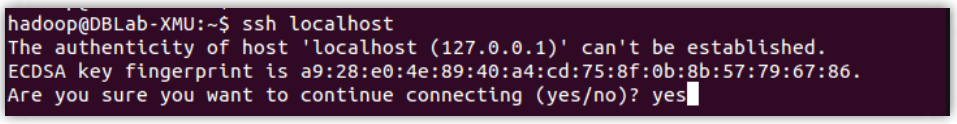
```
sudo apt-get install openssh-server
```

安装后，可以使用如下命令登陆本机：

Shell 命令

```
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。



SSH首次登陆提示

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：

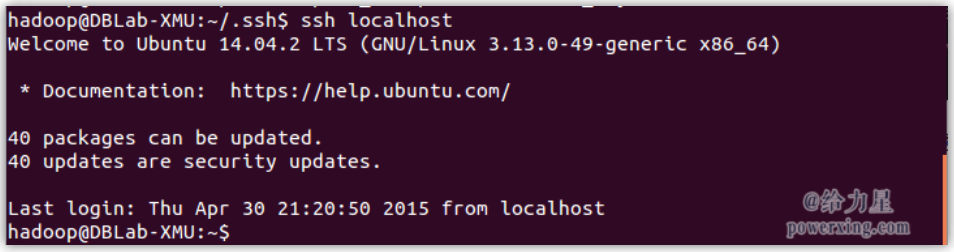
Shell 命令

```
exit                           # 退出刚才的 ssh localhost
cd ~/.ssh/                     # 若没有该目录，请先执行一次ssh local
host
```

```
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
```

> **~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。



SSH无密码登录

安装Java环境

Java环境可选择 Oracle 的 JDK，或是 OpenJDK，按http://wiki.apache.org/hadoop/HadoopJavaVersions中说的，新版本在 OpenJDK 1.7 下是没问题的。为图方便，这边直接通过命令安装 OpenJDK 7。

Shell 命令

```
sudo apt-get install openjdk-7-jre openjdk-7-jdk
```

> **JRE和JDK的区别**
> JRE（Java Runtime Environment，Java运行环境），是运行 Java 所需的环境。JDK（Java Development Kit，Java软件开发工具包）即包括 JRE，还包括开发 Java 程序所需的工具和类库。

安装好 OpenJDK 后，需要找到相应的安装路径，这个路径是用于配置 JAVA_HOME 环境变量的。执行如下命令：

Shell 命令

```
dpkg -L openjdk-7-jdk | grep '/bin/javac'
```

该命令会输出一个路径，除去路径末尾的 “/bin/javac”，剩下的就是正确的路径了。如输出路径为 /usr/lib/jvm/java-7-openjdk-amd64/bin/javac，则我们需要的路径为 /usr/lib/jvm/java-7-openjdk-amd64。

接着配置 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）：

Shell 命令

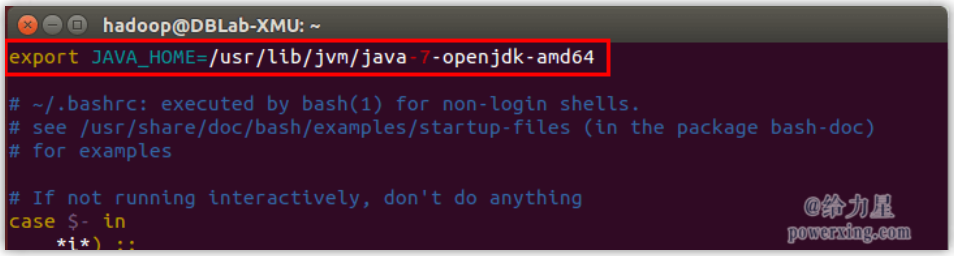
```
vim ~/.bashrc
```

在文件最前面添加如下单独一行（注意 = 号前后不能有空格），将“JDK安装路径”改为上述命令得到的路径，并保存：

Shell

```
1. export JAVA_HOME=JDK安装路径
```

如下图所示（该文件原本可能不存在，内容为空，这不影响）：



配置JAVA_HOME变量

接着还需要让该环境变量生效，执行如下代码：

Shell 命令

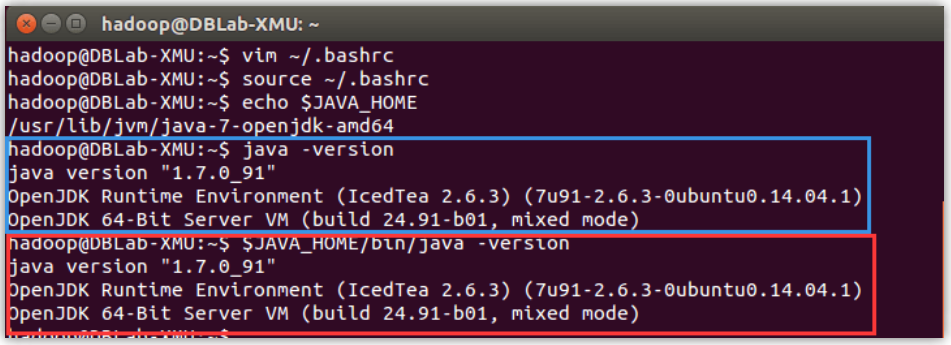
```
source ~/.bashrc    # 使变量设置生效
```

设置好后我们来检验一下是否设置正确：

Shell 命令

```
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：



成功配置JAVA_HOME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/ 或者 http://mirrors.cnnic.cn/apache/hadoop/common/ 下载，一般选择下载最新的稳定版本，即下载 “stable” 下的 **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。

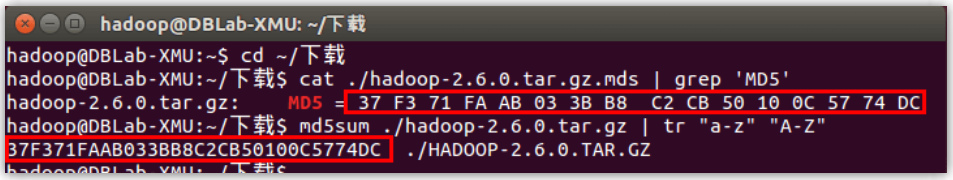
下载时强烈建议也下载 **hadoop-2.x.y.tar.gz.mds** 这个文件，该文件包含了检验值可用于检查 hadoop-2.x.y.tar.gz 的完整性，否则若文件发生了损坏或下载不完整，Hadoop 将无法正常运行。

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，本教程选择的是 2.6.0 版本，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。

Shell 命令

```
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。



检验文件完整性

我们选择将 Hadoop 安装至 /usr/local/ 中：

Shell 命令

```
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hadoop
sudo chown -R 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：

Shell 命令

```
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。

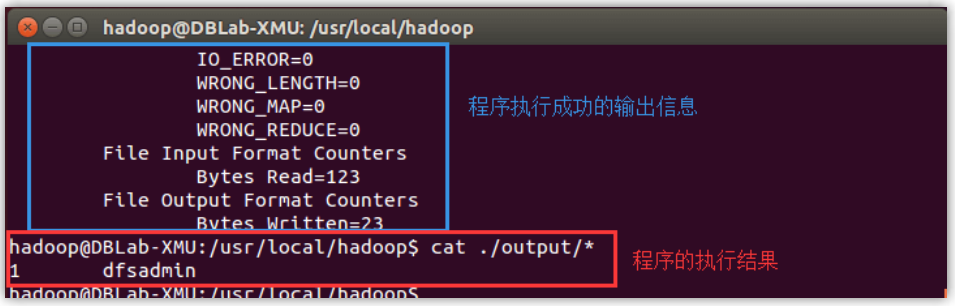
现在我们可以执行例子来感受下 Hadoop 的运行。Hadoop 附带了丰富的例子（运行 `./bin/hadoop jar ./share/hadoop/mapreduce/hadoop-mapreduce-examples-2.6.0.jar` 可以看到所有例子），包括 wordcount、terasort、join、grep 等。

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。

Shell 命令

```
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-example
s-*.jar grep ./input ./output 'dfs[a-z.]+'
cat ./output/*          # 查看运行结果
```

执行成功后如下所示，输出了作业的相关信息，输出的结果是符合正则的单词 dfsadmin 出现了1次



Hadoop单机模式运行grep的输出结果

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。

Shell 命令

```
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。

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

XML

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

修改为下面配置：

XML

```
<configuration>
        <property>
```

```
            <name>hadoop.tmp.dir</name>
            <value>file:/usr/local/hadoop/tmp</value>
            <description>Abase for other temporary directories.</description>
        </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**：

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>
```

Hadoop配置文件说明

Hadoop 的运行方式是由配置文件决定的（运行 Hadoop 时会读取配置文件），因此如果需要从伪分布式模式切换回非分布式模式，需要删除 core-site.xml 中的配置项。

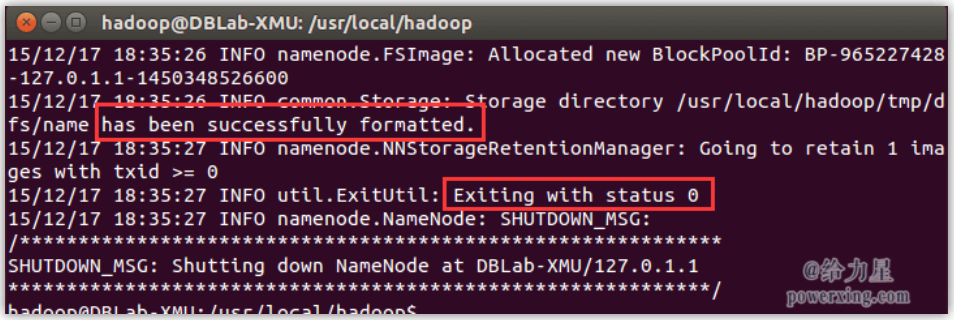
此外，伪分布式虽然只需要配置 fs.defaultFS 和 dfs.replication 就可以运行（官方教程如此），不过若没有配置 hadoop.tmp.dir 参数，则默认使用的临时目录为 /tmp/hadoo-hadoop，而这个目录在重启时有可能被系统清理掉，导致必须重新执行 format 才行。所以我们进行了设置，同时也指定 dfs.namenode.name.dir 和 dfs.datanode.data.dir，否则在接下来的步骤中可能会出错。

配置完成后，执行 NameNode 的格式化:

Shell 命令

```
./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。



执行namenode格式化

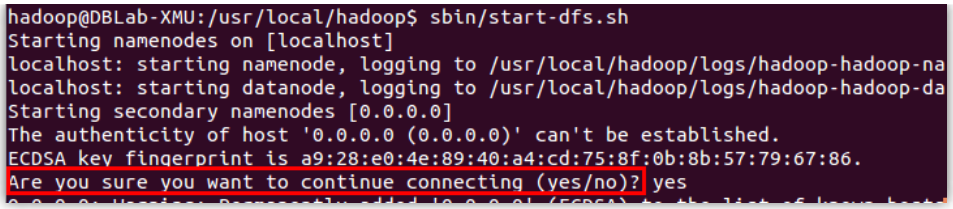
如果在这一步时提示 **Error: JAVA_HOME is not set and could not be found.** 的错误，则说明之前设置 JAVA_HOME 环境变量那边就没设置好，请按教程先设置好 JAVA_HOME 变量，否则后面的过程都是进行不下去的。

接着开启 NameNode 和 DataNode 守护进程。

Shell 命令

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

若出现如下SSH提示，输入yes即可。

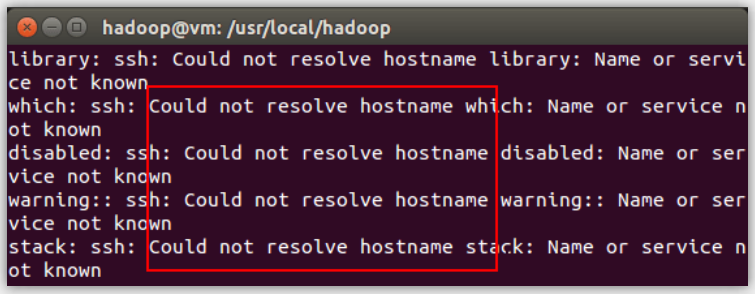


启动Hadoop时的SSH提示

启动时可能会出现如下 WARN 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，并不会影响正常使用（该 WARN 可以通过编译 Hadoop 源码解决）。

启动 Hadoop 时提示 Could not resolve hostname

如果启动 Hadoop 时遇到输出非常多“ssh: Could not resolve hostname xxx”的异常情况，如下图所示：



启动Hadoop时的异常提示

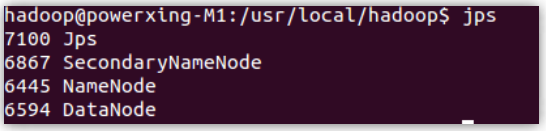
这个并不是 ssh 的问题，可通过设置 Hadoop 环境变量来解决。首先按键盘的 **ctrl + c** 中断启动，然后在 ~/.bashrc 中，增加如下两行内容（设置过程与 JAVA_HOME 变量一样，其中 HADOOP_HOME 为 Hadoop 的安装目录）：

Shell

```
export HADOOP_HOME=/usr/local/hadoop
export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/lib/native
```

保存后，务必执行 `source ~/.bashrc` 使变量设置生效，然后再次执行 `./sbin/start-dfs.sh` 启动 Hadoop。

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。



通过jps查看启动的Hadoop进程

Hadoop无法正常启动的解决方法

一般可以查看启动日志来排查原因，注意几点：

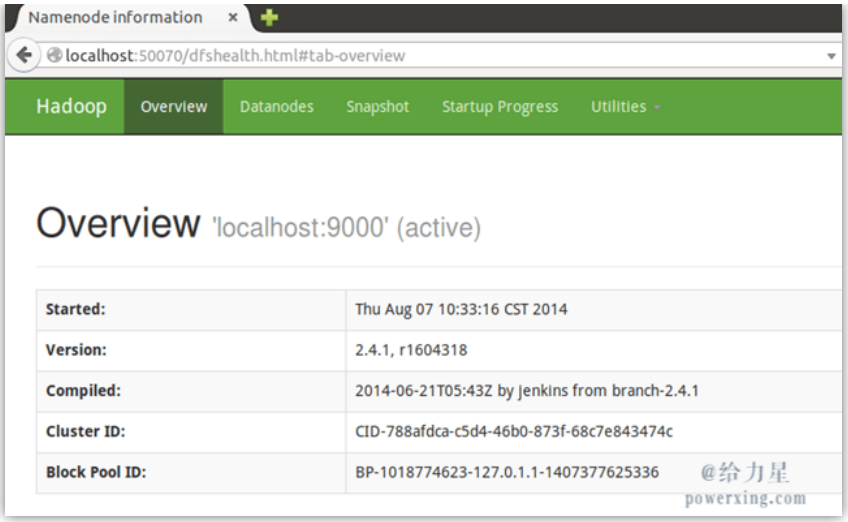
- 启动时会提示形如 “DBLab-XMU: starting namenode, logging to /usr/local/hadoop/logs/hadoop-hadoop-namenode-DBLab-XMU.out”，其中 DBLab-XMU 对应你的机器名，但其实启动日志信息是记录在 /usr/local/hadoop/logs/hadoop-hadoop-namenode-DBLab-XMU.log 中，所以应该查看这个后缀为 **.log** 的文件；
- 每一次的启动日志都是追加在日志文件之后，所以得拉到最后面看，对比下记录的时间就知道了。
- 一般出错的提示在最后面，通常是写着 Fatal、Error、Warning 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。

此外，若是 **DataNode 没有启动**，可尝试如下的方法（注意这会删除 HDFS 中原有的所有数据，如果原有的数据很重要请不要这样做）：

Shell 命令

```
# 针对 DataNode 没法启动的解决方法
./sbin/stop-dfs.sh   # 关闭
rm -r ./tmp     # 删除 tmp 文件，注意这会删除 HDFS 中原有的所有数据
./bin/hdfs namenode -format   # 重新格式化 NameNode
./sbin/start-dfs.sh  # 重启
```

成功启动后，可以访问 Web 界面 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：

Shell 命令

```
./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中的 /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:

Shell 命令

```
./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文件列表：

Shell 命令

```
./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）。

Shell 命令

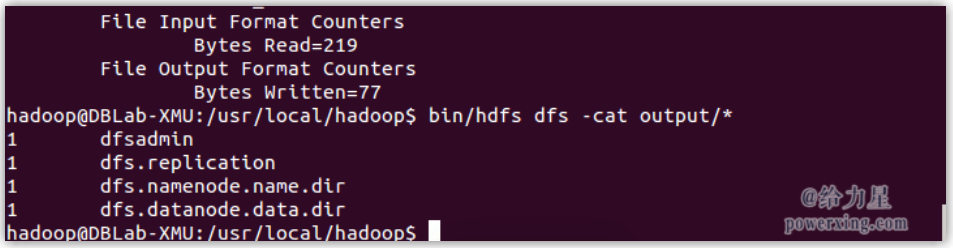
```
./bin/hadoop jar ./share/hadoop/mapreduce/hadoop-mapreduce-example
s-*.jar grep input output 'dfs[a-z.]+'
```

查看运行结果的命令（查看的是位于 HDFS 中的输出结果）：

Shell 命令

```
./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。



Hadoop伪分布式运行grep结果

我们也可以将运行结果取回到本地：

Shell 命令

```
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:

Shell 命令

```
./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：
>
> Java
>
> ```
> Configuration conf = new Configuration();
> Job job = new Job(conf);
>
> /* 删除输出目录 */
> Path outputPath = new Path(args[1]);
> outputPath.getFileSystem(conf).delete(outputPath, true);
> ```

若要关闭 Hadoop，则运行

Shell 命令

```
./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：

Shell 命令

```
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` :

XML

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

接着修改配置文件 **yarn-site.xml**：

XML

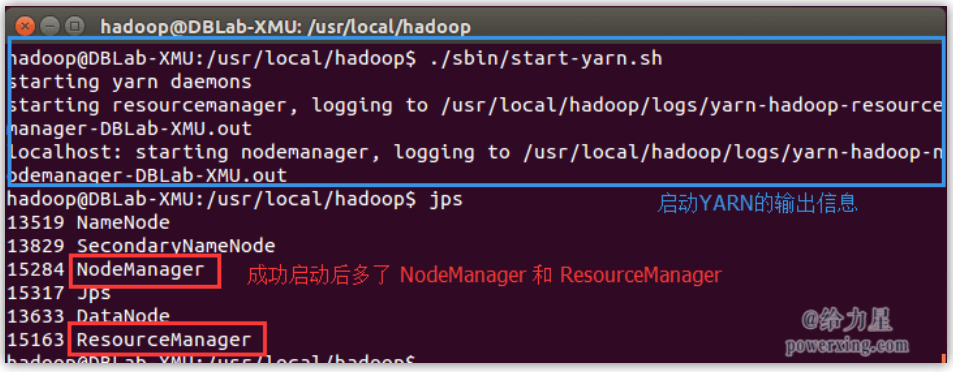
```xml
<configuration>
        <property>
             <name>yarn.nodemanager.aux-services</name>
             <value>mapreduce_shuffle</value>
            </property>
</configuration>
```

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

Shell 命令

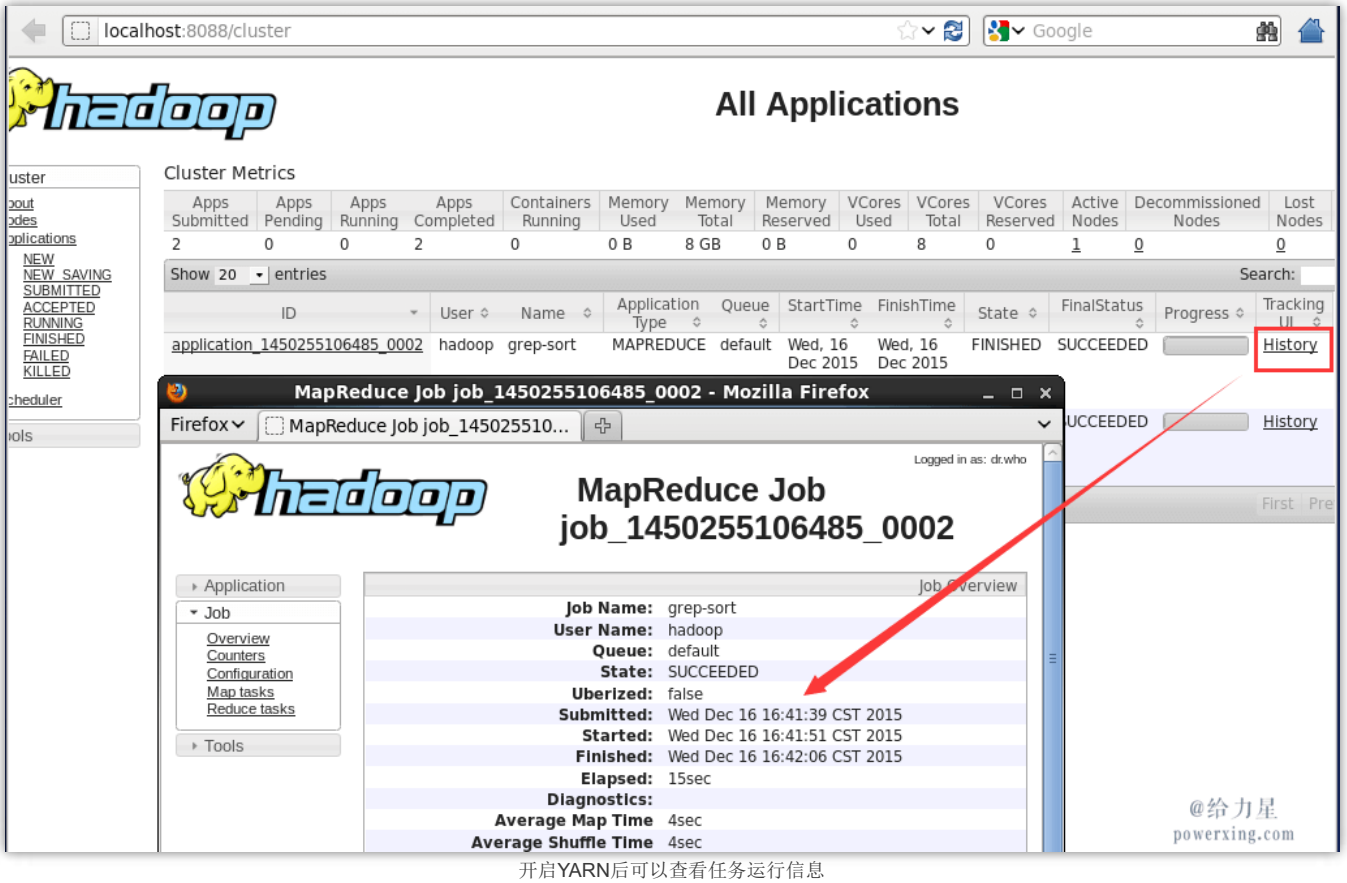
```
./sbin/start-yarn.sh      # 启动YARN
./sbin/mr-jobhistory-daemon.sh start historyserver  # 开启历史服务器，才能在Web中查看任务运行情况
```

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。



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，如下图所示。



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：

Shell 命令

```
./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。

同样我们选择在 ~/.bashrc 中进行设置（ `vim ~/.bashrc` ，与 JAVA_HOME 的设置相似），在文件最前面加入如下单独一行:

```
export PATH=$PATH:/usr/local/hadoop/sbin:/usr/local/hadoop/bin
```

添加后执行 `source ~/.bashrc` 使设置生效，生效后，在任意目录中，都可以直接使用 `hdfs` 等命令了，读者不妨现在就执行 `hdfs dfs -ls input` 查看 HDFS 文件试试看。

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。

相关教程

- 使用Eclipse编译运行MapReduce程序: 使用 Eclipse 可以方便的开发、运行 MapReduce 程序，还可以直接管理 HDFS 中的文件。
- 使用命令行编译打包运行自己的MapReduce程序: 有时候需要直接通过命令来编译、打包 MapReduce 程序。

参考资料

- http://hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-common/SingleCluster.html
- http://www.cnblogs.com/xia520pi/archive/2012/05/16/2503949.html
- http://www.micmiu.com/bigdata/hadoop/hadoop-2x-ubuntu-build/