

Hadoop3.1.3安装教程_单机/伪分布式配置

_Hadoop3.1.3/Ubuntu18.04(16.04)_厦大数据库实验室

博客

已剪辑自: <https://dblab.xmu.edu.cn/blog/2441/>



[点击这里观看厦门大学林子雨老师主讲《大数据技术原理与应用》授课视频](#)

【相关文章推荐】[《大数据软件安装和基础编程实践指南》](#)，详细指导VirtualBox、Ubuntu、Hadoop、HDFS、HBase、Hive、MapReduce、Spark、Flink的安装和基础编程

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。

本教程由[厦门大学数据库实验室](#) / [林子雨](#)出品，转载请注明。本教程适合于原生 Hadoop3.1.3，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。

为了方便学习本教程，请读者们利用Linux系统中自带的firefox浏览器打开本指南进行学习。

Hadoop安装文件，可以到[Hadoop官网](#)下载hadoop-3.1.3.tar.gz。

也可以直接[点击这里从百度云盘下载软件](#)（提取码：ziyu），进入百度网盘后，进入“软件”目录，找到hadoop-3.1.3.tar.gz文件下载到本地。

环境

本教程使用 **Ubuntu 18.04 64位** 作为系统环境（或者Ubuntu 14.04, Ubuntu16.04 也行，32位、64位均可），请自行安装系统（可参考[使用VirtualBox安装Ubuntu](#)）。

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：

1. `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。

sudo命令：本文中会大量使用到sudo命令。sudo是ubuntu中一种权限管理机制，管理员可以授权给一些普通用户去执行一些需要root权限执行的操作。当使用sudo命令时，就需要输入您当前用户的密码。

密码：在Linux的终端中输入密码，终端是不会显示任何你当前输入的密码，也不会提示你已经输入了多少字符密码。而在windows系统中，输入密码一般都会以“*”表示你输入的密码字符

输入法中英文切换：ubuntu中终端输入的命令一般都是使用英文输入。linux中英文的切换方式是使用键盘“shift”键来切换，也可以点击顶部菜单的输入法按钮进行切换。ubuntu自带的Sunpinyin中文输入法已经足够读者使用。

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：

1. `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：

1. `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），返回登陆界面。在登陆界面中选择刚创建的 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：

1. `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。从软件源下载某些软件的过程中，可能由于网络方面的原因出现没法下载的情况，那么建议更改软件源。在学习Hadoop过程中，即使出现“Hash校验和不符”的提示，也不会影响Hadoop的安装。

```
W: 无法下载 bzip2:/var/lib/apt/lists/partial/cn.archive.ubuntu.com_ubuntu_dists_
trusty-updates_universe_i18n_Translation-en Hash 校验和不符合

W: 无法下载 bzip2:/var/lib/apt/lists/partial/cn.archive.ubuntu.com_ubuntu_dists_
trusty-backports_universe_binary-amd64_Packages Hash 校验和不符合

E: Some index files failed to download. They have been ignored, or old ones used
instead.
hadoop@star-lab:~$
```

首先点击左侧任务栏的【系统设置】（齿轮图标），选择【软件和更新】



点击“下载自”右侧的方框，选择【其他节点】



在列表中选【mirrors.aliyun.com】，并点击右下角的【选择服务器】，会要求输入用户密码，输入即可。



接着点击关闭。



此时会提示列表信息过时，点击【重新载入】，



最后耐心等待更新缓存即可。更新完成会自动关闭【软件和更新】这个窗口。如果还是提示错误，请选择其他服务器节点如 mirrors.163.com 再次进行尝试。更新成功后，再次执行 `sudo apt-get update` 就正常了。

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）：

1. `sudo apt-get install vim`

安装软件时若需要确认，在提示处输入 `y` 即可。

```
hadoop@DBLab-XMU:~$ sudo apt-get install vim
正在读取软件包列表... 完成
正在分析软件包的依赖关系树
正在读取状态信息... 完成
将会安装下列额外的软件包：
  vim-runtime
建议安装的软件包：
  ctags vim-doc vim-scripts
下列【新】软件包将被安装：
  vim vim-runtime
升级了 0 个软件包，新安装了 2 个软件包，要卸载 0 个软件包，有 366 个软件包未被升级。
需要下载 0 B/5,844 kB 的软件包。
解压缩后会消耗掉 28.0 MB 的额外空间。
您希望继续执行吗？ [Y/n]
```

vim的常用模式有分为命令模式，插入模式，可视模式，正常模式。本教程中，只需要用到正常模式和插入模式。二者间的切换即可以帮助你完成本指南的学习。

1. 正常模式

正常模式主要用来浏览文本内容。一开始打开vim都是正常模式。在任何模式下按下Esc键就可以返回正常模式

2. 插入编辑模式

插入编辑模式则用来向文本中添加内容的。在正常模式下，输入i键即可进入插入编辑模式

3. 退出vim

如果有利用vim修改任何的文本，一定要记得保存。Esc键退回到正常模式中，然后输入:wq即可保存文本并退出vim

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：

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

安装后，可以使用如下命令登陆本机：

1. `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-XMU:~$ ssh localhost
The authenticity of host 'localhost (127.0.0.1)' can't be established.
ECDSA key fingerprint is a9:28:e0:4e:89:40:a4:cd:75:8f:0b:8b:57:79:67:86.
Are you sure you want to continue connecting (yes/no)? yes
```

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：

1. exit # 退出刚才的 ssh localhost
2. cd ~/.ssh/ # 若没有该目录，请先执行一次ssh
localhost
3. ssh-keygen -t rsa # 会有提示，都按回车就可以
4. 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，只需要输入前面命令即可。

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-XMU:~/.ssh$ ssh localhost
Welcome to Ubuntu 14.04.2 LTS (GNU/Linux 3.13.0-49-generic x86_64)

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

40 packages can be updated.
40 updates are security updates.

Last login: Thu Apr 30 21:20:50 2015 from localhost
hadoop@DBLab-XMU:~$
```

安装Java环境

手动安装，推荐采用本方式

Hadoop3.1.3需要JDK版本在1.8及以上。需要按照下面步骤来自己手动安装JDK1.8。

我们已经把JDK1.8的安装包jdk-8u162-linux-x64.tar.gz放在了百度云盘，[可以点击这里到百度云盘下载JDK1.8安装包](#)（提取码：ziyu）。请把压缩格式的文件jdk-8u162-linux-x64.tar.gz下载到本地电脑，假设保存在“/home/linziyu/Downloads/”目录下。

在Linux命令行界面中，执行如下Shell命令（注意：当前登录用户名是hadoop）：

1. cd /usr/lib
2. sudo mkdir jvm #创建/usr/lib/jvm目录用来存放JDK文件
3. cd ~ #进入hadoop用户的主目录
4. cd Downloads #注意区分大小写字母，刚才已经通过FTP软件把JDK安装包jdk-8u162-linux-x64.tar.gz上传到该目录下
5. sudo tar -zxvf ./jdk-8u162-linux-x64.tar.gz -C /usr/lib/jvm #把JDK文件解压到/usr/lib/jvm目录下

上面使用了解压缩命令tar，如果对Linux命令不熟悉，可以参考[常用的Linux命令用法](#)。

JDK文件解压缩以后，可以执行如下命令到/usr/lib/jvm目录查看一下：

1. cd /usr/lib/jvm
2. ls

可以看到，在/usr/lib/jvm目录下有个jdk1.8.0_162目录。

下面继续执行如下命令，设置环境变量：

1. cd ~
2. vim ~/.bashrc

上面命令使用vim编辑器（[查看vim编辑器使用方法](#)）打开了hadoop这个用户的环境变量配置文件，请在这个文件的开头位置，添加如下几行内容：

```
export JAVA_HOME=/usr/lib/jvm/jdk1.8.0_162
export JRE_HOME=${JAVA_HOME}/jre
export CLASSPATH=.:${JAVA_HOME}/lib:${JRE_HOME}/lib
export PATH=${JAVA_HOME}/bin:$PATH
```

保存.bashrc文件并退出vim编辑器。然后，继续执行如下命令让.bashrc文件的配置立即生效：

1. `source ~/.bashrc`

这时，可以使用如下命令查看是否安装成功：

1. `java -version`

如果能够在屏幕上返回如下信息，则说明安装成功：

```
hadoop@ubuntu:~$ java -version
java version "1.8.0_162"
Java(TM) SE Runtime Environment (build 1.8.0_162-b12)
Java HotSpot(TM) 64-Bit Server VM (build 25.162-b12, mixed mode)
```

至此，就成功安装了Java环境。下面就可以进入Hadoop的安装。

安装 Hadoop3.1.3

Hadoop安装文件，可以到[Hadoop官网](#)下载hadoop-3.1.3.tar.gz。

也可以直接[点击这里从百度云盘下载软件](#)（提取码：1nwl），进入百度网盘后，进入“软件”目录，找到hadoop-3.1.3.tar.gz文件，下载到本地。

我们选择将 Hadoop 安装至 /usr/local/ 中：

1. `sudo tar -zxf ~/下载/hadoop-3.1.3.tar.gz -C /usr/local` # 解压到/usr/local中
2. `cd /usr/local/`
3. `sudo mv ./hadoop-3.1.3/ ./hadoop` # 将文件夹名改为hadoop
4. `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：

1. `cd /usr/local/hadoop`
2. `./bin/hadoop version`

相对路径与绝对路径：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，就不是我们所需要的了。

Hadoop单机配置(非分布式)

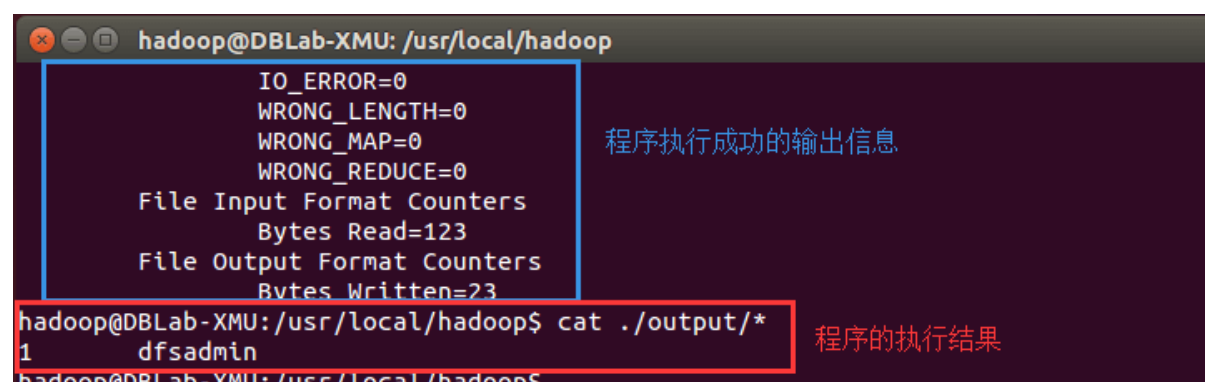
Hadoop 默认模式为非分布式模式（本地模式），无需进行其他配置即可运行。非分布式即单 Java 进程，方便进行调试。

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-3.1.3.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。

1. `cd /usr/local/hadoop`
2. `mkdir ./input`
3. `cp ./etc/hadoop/*.xml ./input` # 将配置文件作为输入文件
4. `./bin/hadoop jar ./share/hadoop/mapreduce/hadoop-mapreduce-examples-3.1.3.jar grep ./input ./output 'dfs[a-z.]'`
5. `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@DBLab-XMU: /usr/local/hadoop
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-XMU: /usr/local/hadoop$ cat ./output/*
1 dfsadmin
hadoop@DBLab-XMU: /usr/local/hadoop$
```

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。

1. `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

1. `<configuration>`
2. `</configuration>`

修改为下面配置：

```
1. <configuration>
2.   <property>
3.     <name>hadoop.tmp.dir</name>
4.     <value>file:/usr/local/hadoop/tmp</value>
5.     <description>Abase for other temporary
   directories.</description>
6.   </property>
7.   <property>
8.     <name>fs.defaultFS</name>
9.     <value>hdfs://localhost:9000</value>
10.  </property>
11. </configuration>
```

同样的，修改配置文件 **hdfs-site.xml**：

```
1. <configuration>
2.   <property>
3.     <name>dfs.replication</name>
4.     <value>1</value>
5.   </property>
6.   <property>
7.     <name>dfs.namenode.name.dir</name>
8.     <value>file:/usr/local/hadoop/tmp/dfs/name</value>
9.   </property>
10.  <property>
11.    <name>dfs.datanode.data.dir</name>
12.    <value>file:/usr/local/hadoop/tmp/dfs/data</value>
13.  </property>
14. </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：

```
1. cd /usr/local/hadoop
2. ./bin/hdfs namenode -format
```

成功的话，会看到 “successfully formatted” 的提示，具体返回信息类似如下：

```
2020-01-08 15:31:31,560 INFO namenode.NameNode: STARTUP_MSG:
/*****
```

```

STARTUP_MSG: Starting NameNode
STARTUP_MSG:  host = hadoop/127.0.1.1
STARTUP_MSG:  args = [-format]
STARTUP_MSG:  version = 3.1.3
*****/

.....
2020-01-08 15:31:35,677 INFO common.Storage: Storage directory
/usr/local/hadoop/tmp/dfs/name **has been successfully formatted**.
2020-01-08 15:31:35,700 INFO namenode.FSImageFormatProtobuf: Saving image
file /usr/local/hadoop/tmp/dfs/name/current/fsimage.ckpt_
00000000000000000000 using no compression
2020-01-08 15:31:35,770 INFO namenode.FSImageFormatProtobuf: Image file
/usr/local/hadoop/tmp/dfs/name/current/fsimage.ckpt_00000000000000000000 of
size 393 bytes saved in 0 seconds .
2020-01-08 15:31:35,810 INFO namenode.NNStorageRetentionManager: Going to
retain 1 images with txid >= 0
2020-01-08 15:31:35,816 INFO namenode.FSImage: FSImageSaver clean
checkpoint: txid = 0 when meet shutdown.
2020-01-08 15:31:35,816 INFO namenode.NameNode: SHUTDOWN_MSG:
/*****
SHUTDOWN_MSG: Shutting down NameNode at hadoop/127.0.1.1
*****/

```

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。如果已经按照前面教程在 .bashrc 文件中设置了 JAVA_HOME，还是出现 **Error: JAVA_HOME is not set and could not be found.** 的错误，那么，请到 hadoop 的安装目录修改配置文件 “/usr/local/hadoop/etc/hadoop/hadoop-env.sh”，在里面找到 “export JAVA_HOME=\${JAVA_HOME}” 这行，然后，把它修改成 JAVA 安装路径的具体地址，比如，“export JAVA_HOME=/usr/lib/jvm/default-java”，然后，再次启动 Hadoop。

接着开启 NameNode 和 DataNode 守护进程。

1. cd /usr/local/hadoop
2. ./sbin/start-dfs.sh #start-dfs.sh 是个完整的可执行文件，中间没有空格

若出现如下 SSH 提示，输入 yes 即可。

```

hadoop@DBLab-XMU:/usr/local/hadoop$ sbin/start-dfs.sh
Starting namenodes on [localhost]
localhost: starting namenode, logging to /usr/local/hadoop/logs/hadoop-hadoop-na
localhost: starting datanode, logging to /usr/local/hadoop/logs/hadoop-hadoop-da
Starting secondary namenodes [0.0.0.0]
The authenticity of host '0.0.0.0 (0.0.0.0)' can't be established.
ECDSA key fingerprint is a9:28:e0:4e:89:40:a4:cd:75:8f:0b:8b:57:79:67:86.
Are you sure you want to continue connecting (yes/no)? yes

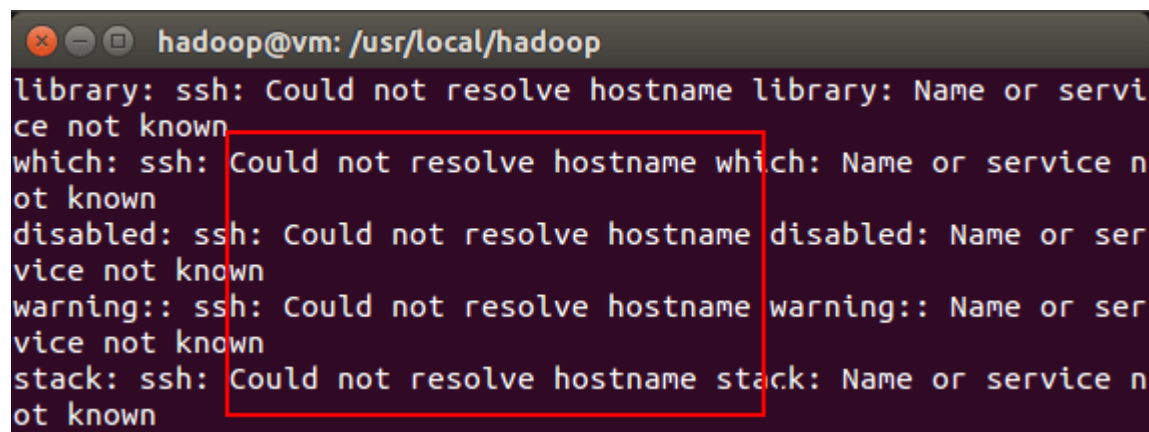
```

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 WARN 提示可以忽略，并不会影响正常使用。

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：

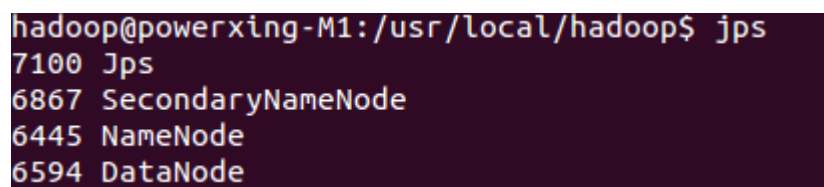
A terminal window titled 'hadoop@vm: /usr/local/hadoop' showing a series of repeated warnings: 'library: ssh: Could not resolve hostname library: Name or service not known', 'which: ssh: Could not resolve hostname which: Name or service not known', 'disabled: ssh: Could not resolve hostname disabled: Name or service not known', 'warning:: ssh: Could not resolve hostname warning:: Name or service not known', and 'stack: ssh: Could not resolve hostname stack: Name or service not known'. A red rectangle highlights the middle three lines of warnings.

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）：

1. `export HADOOP_HOME=/usr/local/hadoop`
2. `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。

A terminal window showing the output of the `jps` command. The output lists four processes: '7100 Jps', '6867 SecondaryNameNode', '6445 NameNode', and '6594 DataNode'.

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）：

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

成功启动后，可以访问 Web 界面 <http://localhost:9870> 查看 NameNode 和 Datanode 信息，还可以在线查看 HDFS 中的文件。

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：

1. ./bin/hdfs dfs -mkdir -p /user/hadoop

注意：教材《大数据技术原理与应用》的命令是以“./bin/hadoop dfs”开头的Shell命令方式，实际上有三种shell命令方式。

1. hadoop fs
2. hadoop dfs
3. hdfs dfs

hadoop fs适用于任何不同的文件系统，比如本地文件系统和HDFS文件系统

hadoop dfs只能适用于HDFS文件系统

hdfs dfs跟hadoop dfs的命令作用一样，也只能适用于HDFS文件系统

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上的 /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：

1. ./bin/hdfs dfs -mkdir input
2. ./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：

1. ./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）。

1. ./bin/hadoop jar ./share/hadoop/mapreduce/hadoop-mapreduce-examples-3.1.3.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）：

1. ./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。

```
File Input Format Counters
  Bytes Read=219
File Output Format Counters
  Bytes Written=77
hadoop@DBLab-XMU:/usr/local/hadoop$ bin/hdfs dfs -cat output/*
1      dfsadmin
1      dfs.replication
1      dfs.namenode.name.dir
1      dfs.datanode.data.dir
hadoop@DBLab-XMU:/usr/local/hadoop$
```

我们也可以将运行结果取回到本地：

1. `rm -r ./output` # 先删除本地的 `output` 文件夹（如果存在）
2. `./bin/hdfs dfs -get output ./output` # 将 HDFS 上的 `output` 文件夹拷贝到本机
3. `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：

1. `./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

1. `./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！

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（Hadoop3.1.3）](#)。