# Hadoop和Spark安装配置

## 实验介绍

#### 1. 实验内容

* hadoop单机模式安装、Spark安装
* 测试安装

#### 2. 实验环境

* jdk-8u261-linux-x64.tar.gz
* Hadoop3.1.4
* spark-3.0.1-bin-hadoop3.2.tgz

#### 3. 适合人群

本课程难度为一般，属于初级级别课程，适合具有linux基础的用户。

## 实验要求

* linux 环境已就绪（本示例采用的是 ubuntu-20.04.1环境）。

## 实验步骤

## 一、Hadoop安装配置

需要先添加用来运行Hadoop进程的用户组hadoop及用户hadoop。

#### 1. 添加用户及用户组

创建用户和用户组hadoop

$ sudo mkdir -p /hadoop

$ sudo groupadd hadoop

$ sudo useradd -g hadoop -G hadoop -d /hadoop hadoop

$ sudo chown -R hadoop:hadoop /hadoop

$ sudo usermod -s /bin/bash hadoop

按照提示输入hadoop用户的密码，例如密码设定为 hadoop。注意输入密码的时候是不显示的。

$ sudo passwd hadoop

Enter new UNIX password: hadoop

Retype new UNIX password: hadoop

passwd: password updated successfully

#### 

#### 2. 添加sudo权限

将hadoop用户添加进sudo用户组

$ sudo usermod -G sudo hadoop

#### 3. 安装及配置依赖的软件包

#### (1) hadoop环境需要预安装openssh-server、java等，这些软件包在实验环境中如果没有，需要手工安装。

安装并启动openssh-verver：

$ sudo apt-get install openssh-server -y

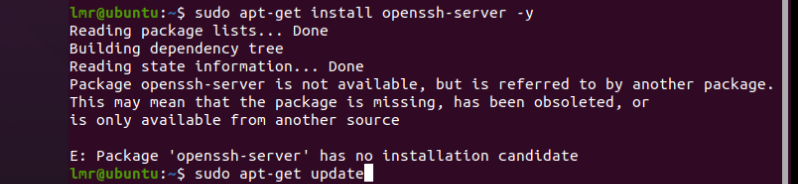
$ sudo service ssh start

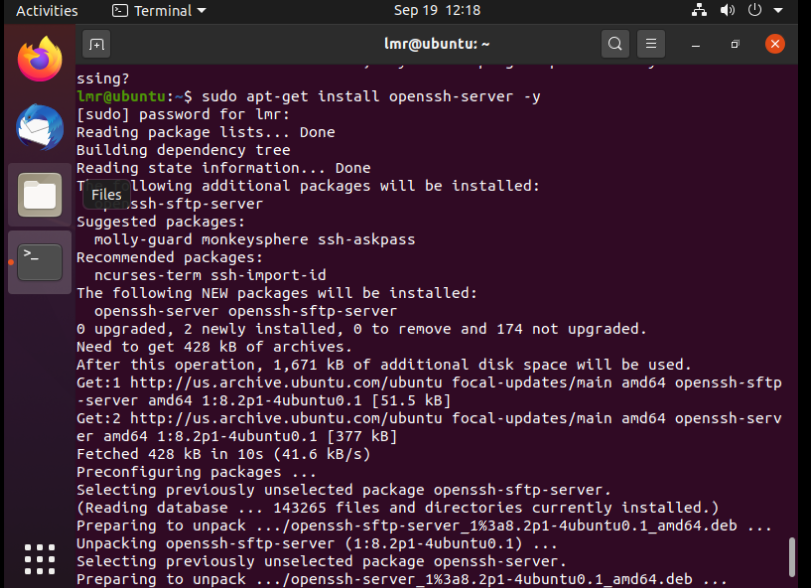
如果提示：Package ‘openssh-server’ has no installation candidate.

请执行以下命令

$ sudo apt-get update

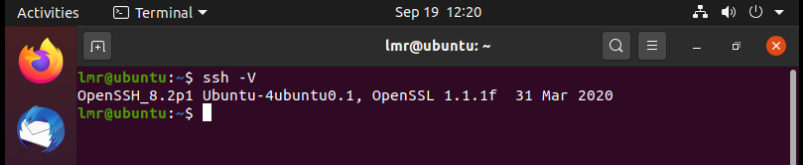
$ sudo apt-get upgrade





验证环境执行下列指令：

$ ssh -V



#### (2) 配置ssh免密码登录

切换到hadoop用户，需要输入添加hadoop用户时配置的密码。后续步骤都将在hadoop用户的环境中执行。

$ su - hadoop

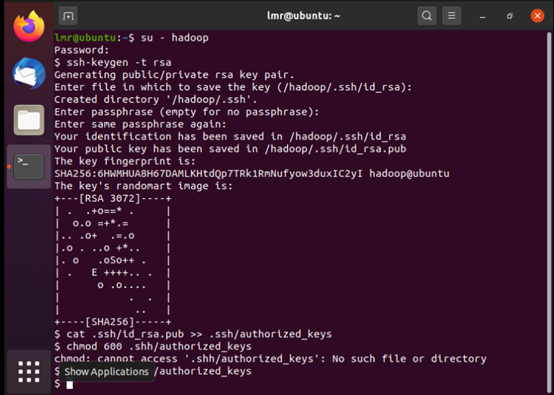
配置ssh环境免密码登录。

在/hadoop目录下执行

$ ssh-keygen -t rsa #一路回车

$ cat .ssh/id\_rsa.pub >> .ssh/authorized\_keys

$ chmod 755 .ssh/authorized\_keys



验证登录本机是否还需要密码，第一次需要输入yes，以后不需要密码就可以登录。

$ ssh localhost

The authenticity of host 'localhost (::1)' can't be established.

ECDSA key fingerprint is 33:5d:12:e4:d5:59:8b:a3:a3:46:45:fd:16:f7:51:c8.

Are you sure you want to continue connecting (yes/no)? yes

Warning: Permanently added 'localhost' (ECDSA) to the list of known hosts.

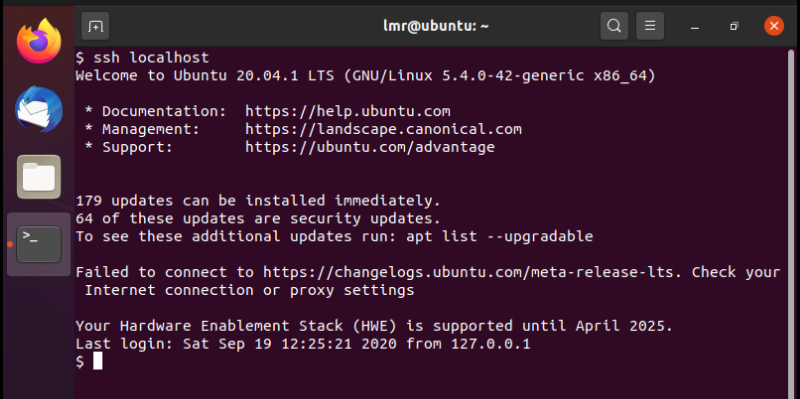
The programs included with the Debian GNU/Linux system are free software;

the exact distribution terms for each program are described in the

individual files in /usr/share/doc/\*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent

permitted by applicable law.



#### 4.下载并安装JAVA和Hadoop

## JAVA 安装包下载地址：

## [https://www.oracle.com/java/technologies/javase/javase-jdk8-downloads.html#license-lightbox](https://www.oracle.com/java/technologies/javase/javase-jdk8-downloads.html" \l "license-lightbox)

## Hadoop安装包下载地址：

## <https://www.apache.org/dyn/closer.cgi/hadoop/common/hadoop-3.1.4/hadoop-3.1.4.tar.gz>

#### 可以先下载好，存放在共享文件夹下：

#### 

在hadoop用户登录的环境中进行下列操作：

#### (1) 复制压缩包到/hadoop目录下

$ cp /mnt/hgfs/Share/hadoop-3.1.4.tar.gz /hadoop/

$ cp /mnt/hgfs/Share/jdk-8u261-linux-x64.tar.gz /hadoop/

#### (2) 解压并安装

$ cd /hadoop

$ tar -zxvf hadoop-3.1.4.tar.gz

$ chmod 755 hadoop

$ tar -zxvf jdk-8u261-linux-x64.tar.gz

#### (3) 配置Hadoop环境变量

$ vi /hadoop/.bash\_profile

增加以下内容：

*#HADOOP START*

*export JAVA\_HOME=/hadoop/jdk1.8.0\_261*

*export HADOOP\_HOME=/hadoop/hadoop-3.1.4*

*export PATH=$PATH:$HADOOP\_HOME/bin:$HADOOP\_HOME/sbin:$JAVA\_HOME/bin*

*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 HADOOP\_OPTS="-Djava.library.path=$HADOOP\_HOME/lib"*

*#HADOOP END*

配置成功后，激活新加的环境变量

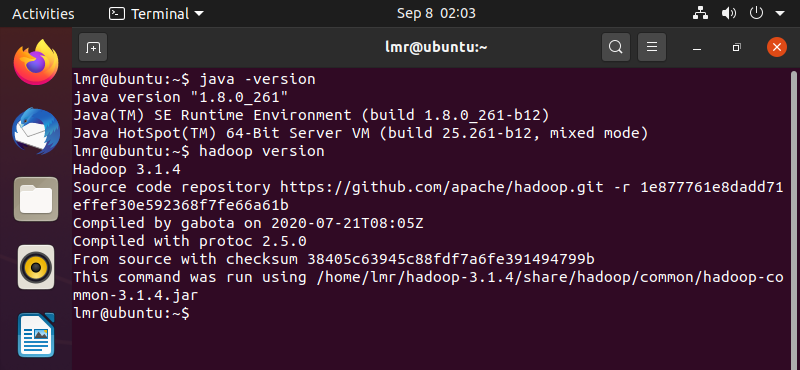
$ bash

$ source ~/.bash\_profile

#### (4)查看JAVA和Hadoop版本

$ java –version

$ hadoop version



至此，Hadoop单机模式安装完成，可以通过下述步骤的测试来验证安装是否成功。

## 实验结果

创建输入的数据，暂时采用/etc/protocols文件作为测试

$ mkdir /hadoop/hadoop\_project

$ cd /hadoop/hadoop\_project

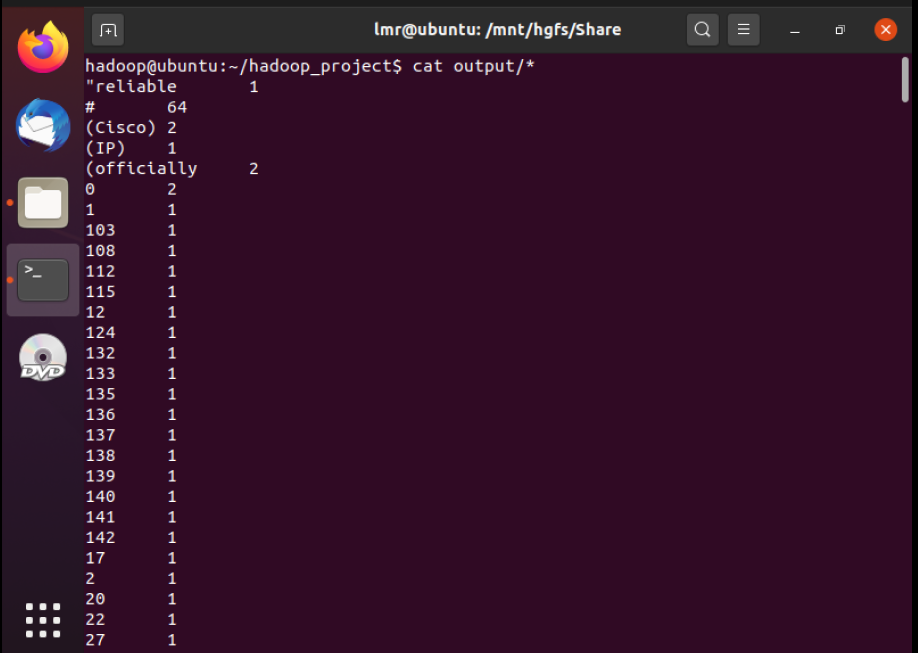
$ cp /etc/protocols input

执行Hadoop WordCount应用（词频统计）

$ /hadoop/hadoop-3.1.4/bin/hadoop jar /hadoop/hadoop-3.1.4/share/hadoop/mapreduce/sources/hadoop-mapreduce-examples-3.1.4-sources.jar org.apache.hadoop.examples.WordCount input output

查看生成的单词统计数据

$ cat output/\*



## 练习题

请使用hadoop的wordcount对日志文件/var/log/dpkg.log进行词频统计。 将你执行的命令，和输出的结果粘贴到下面的文本框中。

作业练习：

###输入你的作业代码###

###作业代码结束###

## 二、Spark安装配置

#### 1.hadoop伪分布式配置

hadoop的配置文件存放在/hadoop/hadoop-3.1.4/etc/hadoop下，要修改该目录下的文件core-site.xml和hdfs-site.xml来达到实现伪分布式配置。

修改core-site.xml，将<configure></configure>修改为：

<configuration>

<property>

<name>hadoop.tmp.dir</name>

<value>file:/hadoop/hadoop-3.1.4/tmp</value>

<description>Abase for other temporary directories.</description>

</property>

<property>

<name>fs.defaultFS</name>

<value>hdfs://localhost:9000</value>

</property>

</configuration>

#### 修改hdfs-site.xml，将<configure></configure>修改为：

<configuration>

<property>

<name>dfs.replication</name>

<value>1</value>

</property>

<property>

<name>dfs.namenode.name.dir</name>

<value>file:/hadoop/hadoop-3.1.4/tmp/dfs/name</value>

</property>

<property>

<name>dfs.datanode.data.dir</name>

<value>file:/hadoop//hadoop-3.1.4/tmp/dfs/data</value>

</property>

</configuration>

#### 配置完成后在/hadoop/hadoop-3.1.4下使用命令

#### $ ./bin/hdfs namenode -format  实现namenode的格式化。

#### 2.启动hadoop（namenode节点）（start-all.sh在sbin里面）

启动命令为：

$ start-all.sh

检查是否运行成功

#执行jps命令可以查看到hadoop的几个主要进程:

$ jps

ResourceManager

NodeManager

Jps

NameNode

SecondaryNameNode

DataNode

#### 3.安装与配置 Spark

#### （1）解压并安装Spark

本次实验我们将spark安装在/hadoop/app下，因此我们建立spark的安装目录

$ mkdir /hadoop/app

下载安装包有如下两个方法：

$ wget http://archive.apache.org/dist/spark/spark-3.1.2/spark-3.1.2-bin-hadoop3.2.tgz

解压

$ cd /hadoop/app

$ tar -zxvf spark-3.0.1-bin-hadoop3.2.tgz

删除安装文件

$ rm -r spark-3.0.1-bin-hadoop3.2.tgz

修改文件名称

$ mv spark-3.0.1-bin-hadoop3.2 spark

#### （2）配置 Hadoop 环境变量

在 Yarn 上运行 Spark 需要配置 HADOOP\_CONF\_DIR、 YARN\_CONF\_DIR 和 HDFS\_CONF\_DIR 环境变量

命令：

$ vi /hadoop/.bash\_profile

在下面添加如下代码：

export SPARK\_HOME=/hadoop/app/spark

export HADOOP\_CONF\_DIR=$HADOOP\_HOME/etc/hadoop

export HDFS\_CONF\_DIR=$HADOOP\_HOME/etc/hadoop

export YARN\_CONF\_DIR=$HADOOP\_HOME/etc/hadoop

保存关闭后，执行以下命令使得环境变量生效：

$ source /hadoop/.bash\_profile

#### （2）修改配置文件

$ cd /hadoop/app/spark/conf/

$ cp spark-env.sh.template spark-env.sh

$ vi spark-env.sh

在第一行“#!/usr/bin/env bash”下，写入以下内容

export SPARK\_MASTER\_HOST=127.0.0.1

export SPARK\_MASTER\_PORT=7077

export SPARK\_WORKER\_CORES=1

export SPARK\_WORKER\_MEMORY=512M

#### 4. Spark的 启动

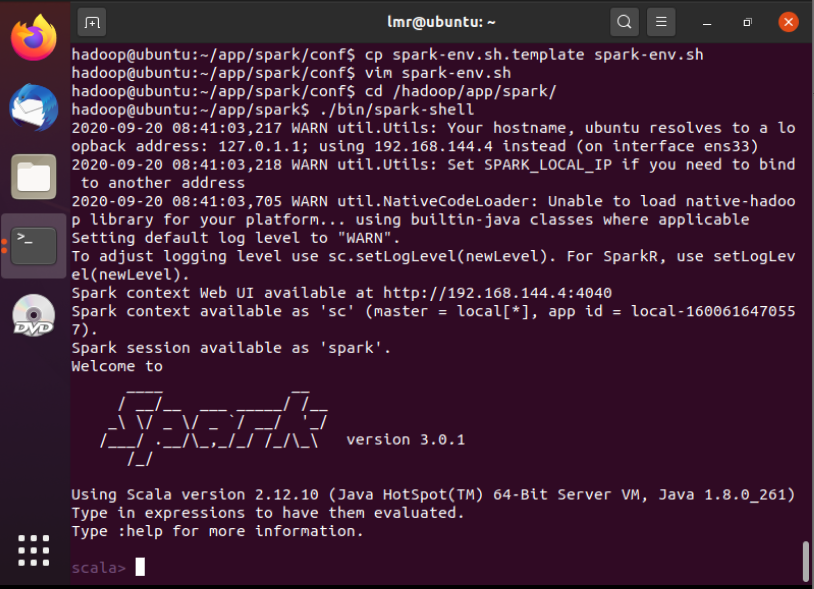
#### （1）进入spark-shell

进入 Spark 安装主目录,

$ cd /hadoop/app/spark

进入spark的shell界面

$ ./bin/spark-shell



* 使用“CTRL+C”键退出spark-shell界面。

# 启动shell或者sql报错：Error: A JNI error has occurred, please check your installation and try again Exception in thread “main” java.lang.NoClassDefFoundError: org/slf4j/Logger

.bash\_profile加入以下环境变量

#添加Hadoop的classpath到SPARK\_DIST\_CLASSPAHT中

export SPARK\_DIST\_CLASSPATH=$(hadoop classpath)

#### （2）启动spark

①首先启动master

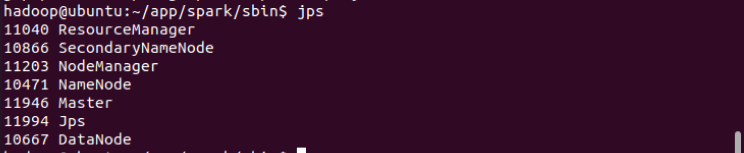
$ cd /hadoop/app/spark/sbin/

$ ./start-master.sh



查看Master进程是否启动

$ jps



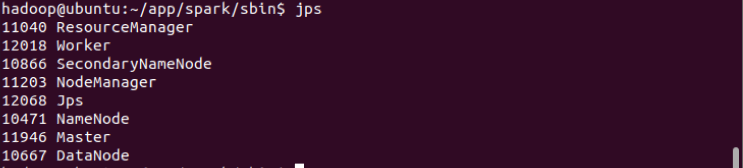
②启动slave

$ ./start-slave.sh spark://127.0.0.1:7077



查看Woker进程是否启动

$ jps



#### 5. 验证Spark

#### 运行pi（π）的实例

$ cd /hadoop/app/spark/

$ ./bin/spark-submit --class org.apache.spark.examples.SparkPi --master spark://127.0.0.1:7077 --driver-memory 512M --executor-memory 512M --executor-cores 1 ./examples/jars/spark-examples\*.jar

结果在执行过程中的其中一行，需要大家仔细查看，如下图：

Pi is roughly 3.14\*\*

说明：

spark-submit 可以提交任务到 spark 集群执行，也可以提交到 hadoop 的 yarn 集群执行。

参数的含义：

--class 应用程序的主类，仅针对 java 或 scala 应用。这里我们使用的是spark自带的计算pi的类。

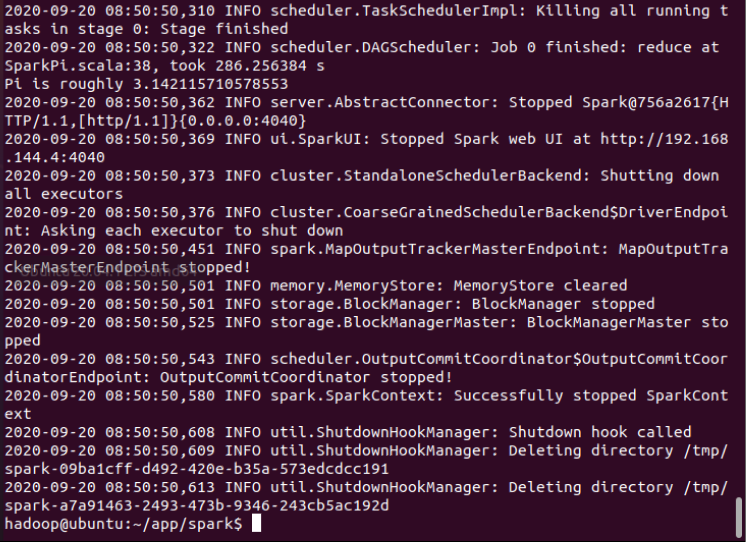
--master master 的地址，提交任务到哪里执行。

--driver-memory Driver内存，默认 1G。

--executor-memory 每个 executor 的内存，默认是1G。

--driver-cores Driver 的核数，默认是1。在 yarn 或者 standalone 下使用。

./examples/jars/spark-examples\*.jar 指的是/hadoop/app/spark/examples/jars下的spark-examples\*.jar包，运行pi的类就写在这些jar包里。



执行时会输出非常多的运行信息，输出结果不容易找到，可以通过 grep 命令进行过滤（命令中的 2>&1 可以将所有的信息都输出到 stdout 中，否则由于输出日志的性质，还是会输出到屏幕中）：

$ ./bin/spark-submit --class org.apache.spark.examples.SparkPi --master spark://127.0.0.1:7077 --driver-memory 512M --executor-memory 512M --executor-cores 1 ./examples/jars/spark-examples\*.jar 2>&1 | grep "Pi is roughly"

结果如下图（结果可能会有微小差别）：

