

# Hadoop和Spark安装配置

## 实验一

信息与软件工程学院  
大数据分析 & 智能计算





- hadoop单机模式安装
- hadoop伪分布式安装
- Spark安装
- 测试安装



- ubuntu-18.04及以上环境（实验指导书示例使用20.04环境）
- jdk-8u261-linux-x64.tar.gz
- Hadoop 3.1.4
- spark-3.0.1-bin-hadoop3.2.tgz
- 提示：虚拟机在安装的时候 请为虚拟机分配大点的磁盘空间和内存，避免后期资源，尤其磁盘空间不足（建议60G以上，有条件设置100G）

## • 一、Hadoop单机模式安装配置

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
```

## • 一、Hadoop安装配置

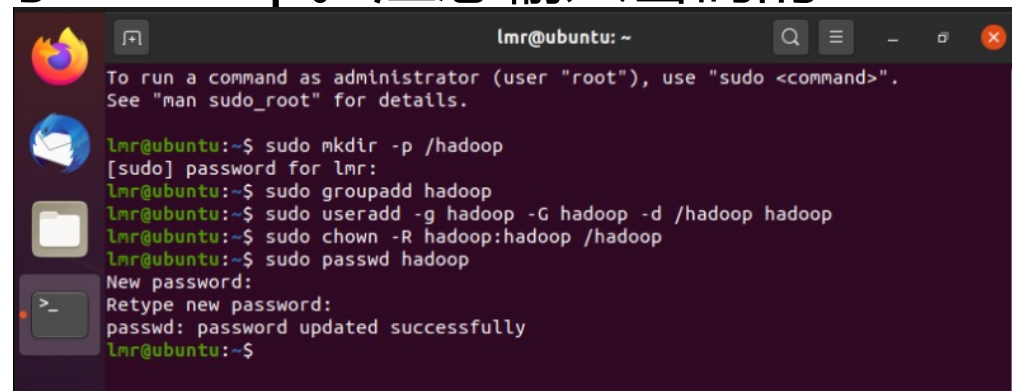
按照提示输入hadoop用户的密码，例如密码设定为 hadoop。注意输入密码的时候是不显示的。

```
$ sudo passwd hadoop
```

Enter new UNIX password:

hadoopRetype new UNIX password:

hadooppasswd: password updated successfully



```
lmr@ubuntu: ~  
To run a command as administrator (user "root"), use "sudo <command>".  
See "man sudo_root" for details.  
  
lmr@ubuntu:~$ sudo mkdir -p /hadoop  
[sudo] password for lmr:  
lmr@ubuntu:~$ sudo groupadd hadoop  
lmr@ubuntu:~$ sudo useradd -g hadoop -G hadoop -d /hadoop hadoop  
lmr@ubuntu:~$ sudo chown -R hadoop:hadoop /hadoop  
lmr@ubuntu:~$ sudo passwd hadoop  
New password:  
Retype new password:  
passwd: password updated successfully  
lmr@ubuntu:~$
```

## 2. 添加sudo权限

将hadoop用户添加进sudo用户组

```
$ sudo usermod -G sudo hadoop
```

## • 一、Hadoop安装配置

### 3. 安装及配置依赖的软件包

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。

#### 3.1 安装并启动`openssh-server`:

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

```
$ sudo service ssh start
```

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

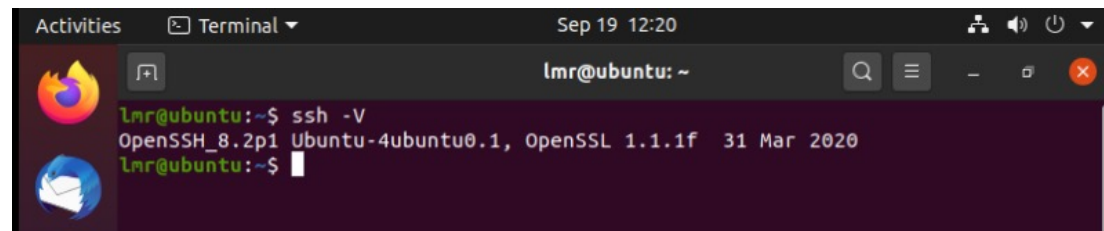
请执行以下命令

```
$ sudo apt-get update
```

```
$ sudo apt-get upgrade
```

验证环境执行下列指令:

```
$ ssh -V
```



```
Activities Terminal Sep 19 12:20  
lmr@ubuntu: ~  
lmr@ubuntu:~$ ssh -V  
OpenSSH_8.2p1 Ubuntu-4ubuntu0.1, OpenSSL 1.1.1f 31 Mar 2020  
lmr@ubuntu:~$
```



## • 一、Hadoop安装配置

### 3.2 配置ssh免密码登录

(1) 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
```

(2) 配置ssh环境免密码登录。

在/hadoop目录下执行

```
$ ssh-keygen -t rsa
```

一直按回车键

然后将生成的rsa密钥复制为ssh认证key

```
$ cat .ssh/id_rsa.pub >> .ssh/authorized_keys
```

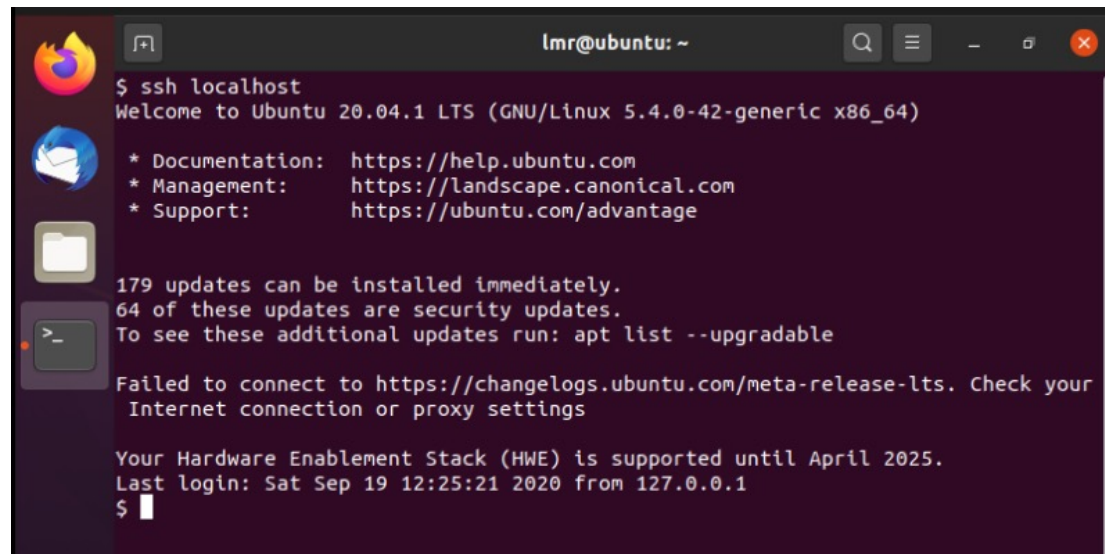
```
$ chmod 755 .ssh/authorized_keys
```

## • 一、Hadoop安装配置

### 3.3 本机验证ssh免密码登录

\$ 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.
```



```
lmr@ubuntu: ~
$ ssh localhost
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-42-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

179 updates can be installed immediately.
64 of these updates are security updates.
To see these additional updates run: apt list --upgradable

Failed to connect to https://changelogs.ubuntu.com/meta-release-lts. Check your
Internet connection or proxy settings

Your Hardware Enablement Stack (HWE) is supported until April 2025.
Last login: Sat Sep 19 12:25:21 2020 from 127.0.0.1
$
```





## • 一、Hadoop安装配置

### 4. 下载JAVA和Hadoop

JAVA 安装包下载地址（本地址会链接到jdk1.8.0\_xxx的最新版本）：

<https://www.oracle.com/java/technologies/javase/javase-jdk8-downloads.html#license-lightbox>

Hadoop安装包下载地址：

<https://archive.apache.org/dist/hadoop/common/hadoop-3.1.4/hadoop-3.1.4.tar.gz>

## • 一、Hadoop安装配置

**安装JAVA和Hadoop**：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
```

## • 一、Hadoop安装配置

### (3) 配置Java和Hadoop环境变量

\$ vi /hadoop/.bash\_profile增加以下内容：（或者使用gedit编辑器）

配置成功后，激活新加的环境变量  
\$ bash  
\$ source ~/.bash\_profile

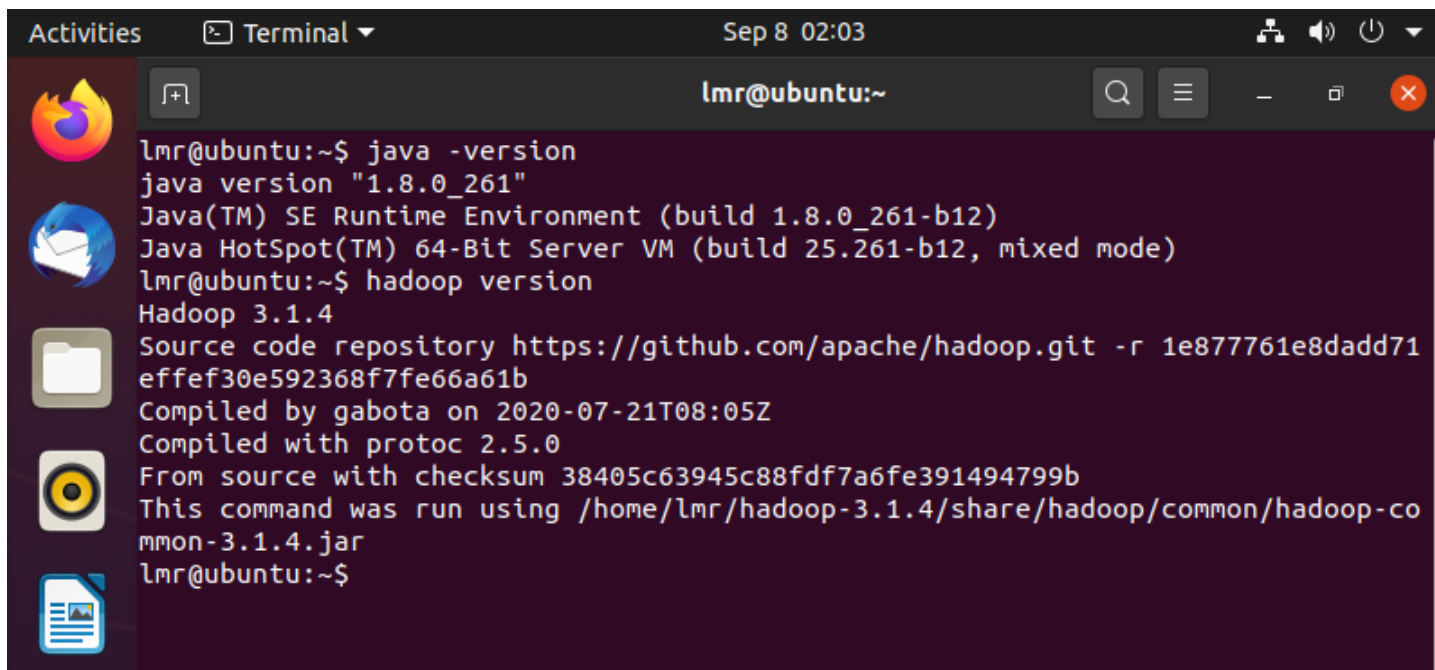
```
#HADOOP START
export JAVA_HOME=/hadoop/jdk1.8.0_261
export HADOOP_HOME=/hadoop/hadoop-3.1.4
export PATH=$PATH:$HADOOP_HOME/bin:$HADOOP_HOME/sbin:$JAVA_HOME/bin
export HADOOP_MAPRED_HOME=$HADOOP_HOME
export HADOOP_COMMON_HOME=$HADOOP_HOME
export HADOOP_HDFS_HOME=$HADOOP_HOME
export YARN_HOME=$HADOOP_HOME
export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/lib/native
export HADOOP_OPTS="-Djava.library.path=$HADOOP_HOME/lib"
#HADOOP END
```

## • 一、Hadoop安装配置

### (4)查看JAVA和Hadoop版本

\$ java -version

\$ hadoop version



```
Activities Terminal Sep 8 02:03
lmr@ubuntu:~$ java -version
java version "1.8.0_261"
Java(TM) SE Runtime Environment (build 1.8.0_261-b12)
Java HotSpot(TM) 64-Bit Server VM (build 25.261-b12, mixed mode)
lmr@ubuntu:~$ hadoop version
Hadoop 3.1.4
Source code repository https://github.com/apache/hadoop.git -r 1e877761e8dadd71
effef30e592368f7fe66a61b
Compiled by gabota on 2020-07-21T08:05Z
Compiled with protoc 2.5.0
From source with checksum 38405c63945c88fdf7a6fe391494799b
This command was run using /home/lmr/hadoop-3.1.4/share/hadoop/common/hadoop-co
mmon-3.1.4.jar
lmr@ubuntu:~$
```

## • 一、Hadoop安装配置

- 后面如果启动Hadoop的时候找不到JAVA\_HOME,

1.切换到: [hadoop路径]/etc/hadoop

2.执行: gedit hadoop-env.sh

3.修改java\_home路径和hadoop\_conf\_dir路径为具体的安装路径

```
export JAVA_HOME= /hadoop/jdk1.8.0_261  
export HADOOP_CONF_DIR=/usr/hadoop-3.1.4/etc/hadoop
```

4. 重新加载使修改生效: source hadoop-env.sh

## • 一、Hadoop安装配置

### 验证Hadoop环境

#### (1) 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
```

#### (2) 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
```

#### (3) 查看生成的单词统计数据

```
$ cat output/*
```

```
lmr@ubuntu: /mnt/hgfs/Share
hadoop@ubuntu:~/hadoop_project$ cat output/*
"reliable" 1
# 64
(Cisco) 2
(IP) 1
(officially 2
0 2
1 1
103 1
```



## • 一、Hadoop安装配置

### 练习题

请使用hadoop的wordcount对日志文件/var/log/dpkg.log进行词频统计。 最终需要将执行的命令和输出的结果粘贴到报告文件中。



## • 二、Hadoop伪分布式配置

1. 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>
```





## • 二、Hadoop伪分布式配置

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>
```

## • 二、Hadoop伪分布式配置

### 1. 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。

格式化失败注意权限问题

```
Sudo Unable to load native-hadoop library for your platform
```

```
$ Chmod -R a+w /hadoop
```

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

## • 二、Hadoop伪分布式配置

### 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的主要进程 **(这几个进程一定要都启动起来才成功)**

```
$ jps
```

ResourceManager

NodeManager

Jps

NameNode

SecondaryNameNode

DataNode



## • 二、Hadoop伪分布式配置

通过web 访问resourcemanager界面

<http://localhost:8088>

通过web访问namenode HDFS web 界面

<http://localhost:9870>

- ubuntu 关闭防火墙:ufw disable
- hadoop3.0以下版本web访问端口50070; 3.0及以上web访问端口9870



## • 三、Spark安装配置

### 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`



## • 二、Hadoop伪分布式配置

### 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**可以替换为**gedit**)

```
$ vi /hadoop/.bash_profile
```

在下面继续添加如下代码：

```
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
```



## • 三、Spark安装配置

### 3) 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
```

### • 三、Spark安装配置

## 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
```

```
hadoop@ubuntu:~/app/spark/conf$ cp spark-env.sh.template spark-env.sh
hadoop@ubuntu:~/app/spark/conf$ vim spark-env.sh
hadoop@ubuntu:~/app/spark/conf$ cd /hadoop/app/spark/
hadoop@ubuntu:~/app/spark$ ./bin/spark-shell
2020-09-20 08:41:03,217 WARN util.Utils: Your hostname, ubuntu resolves to a lo
opback address: 127.0.1.1; using 192.168.144.4 instead (on interface ens33)
2020-09-20 08:41:03,218 WARN util.Utils: Set SPARK_LOCAL_IP if you need to bind
to another address
2020-09-20 08:41:03,705 WARN util.NativeCodeLoader: Unable to load native-hadoo
p library for your platform... using builtin-java classes where applicable
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel
(newLevel).
Spark context Web UI available at http://192.168.144.4:4040
Spark context available as 'sc' (master = local[*], app id = local-160061647055
7).
Spark session available as 'spark'.
Welcome to

  ____      _
 / ___|  _ \| | | |
 \___ \ |_) | | | |
  ___) ||  __| | | |
 |____||___|_| |_| |
                        version 3.0.1

Using Scala version 2.12.10 (Java HotSpot(TM) 64-Bit Server VM, Java 1.8.0_261)
Type in expressions to have them evaluated.
Type :help for more information.

scala>
```



## • 三、Spark安装配置

### (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Worker进程是否启动

`$ jps`

```
hadoop@ubuntu:~/app/spark/sbin$ ./start-master.sh
starting org.apache.spark.deploy.master.Master, logging to /hadoop/app/spark/logs/spark-hadoop-org.apache.spark.deploy.master.Master-1-ubuntu.out
```

```
hadoop@ubuntu:~/app/spark/sbin$ jps
11040 ResourceManager
12018 Worker
10866 SecondaryNameNode
11203 NodeManager
12068 Jps
10471 NameNode
11946 Master
10667 DataNode
```

## • 三、Spark安装配置

### 验证Spark

#### 运行pi ( $\pi$ ) 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\*\*

## • 三、Spark安装配置

```
2020-09-20 08:50:50,310 INFO scheduler.TaskSchedulerImpl: Killing all running t
asks in stage 0: Stage finished
2020-09-20 08:50:50,322 INFO scheduler.DAGScheduler: Job 0 finished: reduce at
SparkPi.scala:38, took 280.250384 s
Pi is roughly 3.142115710578553
2020-09-20 08:50:50,362 INFO server.AbstractConnector: Stopped Spark@756a2617{H
TTP/1.1,[http/1.1]}{0.0.0.0:4040}
2020-09-20 08:50:50,369 INFO ui.SparkUI: Stopped Spark web UI at http://192.168
.144.4:4040
2020-09-20 08:50:50,373 INFO cluster.StandaloneSchedulerBackend: Shutting down
all executors
2020-09-20 08:50:50,376 INFO cluster.CoarseGrainedSchedulerBackend$DriverEndpoi
nt: Asking each executor to shut down
2020-09-20 08:50:50,451 INFO spark.MapOutputTrackerMasterEndpoint: MapOutputTra
ckerMasterEndpoint stopped!
2020-09-20 08:50:50,501 INFO memory.MemoryStore: MemoryStore cleared
2020-09-20 08:50:50,501 INFO storage.BlockManager: BlockManager stopped
2020-09-20 08:50:50,525 INFO storage.BlockManagerMaster: BlockManagerMaster sto
pped
2020-09-20 08:50:50,543 INFO scheduler.OutputCommitCoordinator$OutputCommitCoor
dinatorEndpoint: OutputCommitCoordinator stopped!
2020-09-20 08:50:50,580 INFO spark.SparkContext: Successfully stopped SparkCont
ext
2020-09-20 08:50:50,608 INFO util.ShutdownHookManager: Shutdown hook called
2020-09-20 08:50:50,609 INFO util.ShutdownHookManager: Deleting directory /tmp/
spark-09ba1cfff-d492-420e-b35a-573edcdcc191
2020-09-20 08:50:50,613 INFO util.ShutdownHookManager: Deleting directory /tmp/
spark-a7a91463-2493-473b-9346-243cb5ac192d
hadoop@ubuntu:~/app/spark$
```

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。



## • 三、Spark安装配置

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）：

```
hadoop@ubuntu:~/app/spark$ ./bin/spark-submit --class org.apache.spark.examples  
.SparkPi --master spark://127.0.0.1:7077 --driver-memory 512M --executor-memor  
y 512M --executor-cores 1 ./examples/jars/spark-examples*.jar 2>&1 | grep "Pi i  
s roughly"  
Pi is roughly 3.142355711778559
```



- 在Hadoop启动以后，namenode是通过SSH（Secure Shell）来启动和停止各个节点上的各种[守护进程](#)的，这就需要在节点之间执行指令的时候是不需要输入密码的方式
- 故我们需要配置SSH使用无密码公钥认证的方式。



- **core-site.xml (有多个配置项可选) :**

- ① **fs.defaultFS**指定namenode的位置
- ② **hadoop.tmp.dir**是hadoop文件系统依赖的基础配置，很多路径都依赖它。  
如果hdfs-site.xml中不配置namenode和datanode的存放位置，默认就放在这个路径中。

- **hdfs-site.xml:**

- ① **dfs.namenode.name.dir/dfs.datanode.data.dir**配置namenode和datanode存放文件的具体路径
- ② **dfs.replication**配置副本的数量，最小值为3，否则会影响到数据的可靠性



- **yarn-site.xml**

- ① **Yarn.resourcemanager.hostname**: 资源管理器所在节点的主机名
  - ② **Yarn.nodemanager.aux-services**: 一个逗号分隔的辅助服务列表，这些服务由节点管理器执行。该属性默认为空。
- **mapred-site.xml**: 此文件如果没有，需要将mapred-site.xml.template重命名  
**Mapreduce.framework.name**: 决定mapreduce作业是提交到 YARN 集群还是使用本地作业执行器本地执行。