Spark 编译与部署（下）
--Spark 编译安装

目 录

Spark 编译与部署（下）

1 编译 Spark

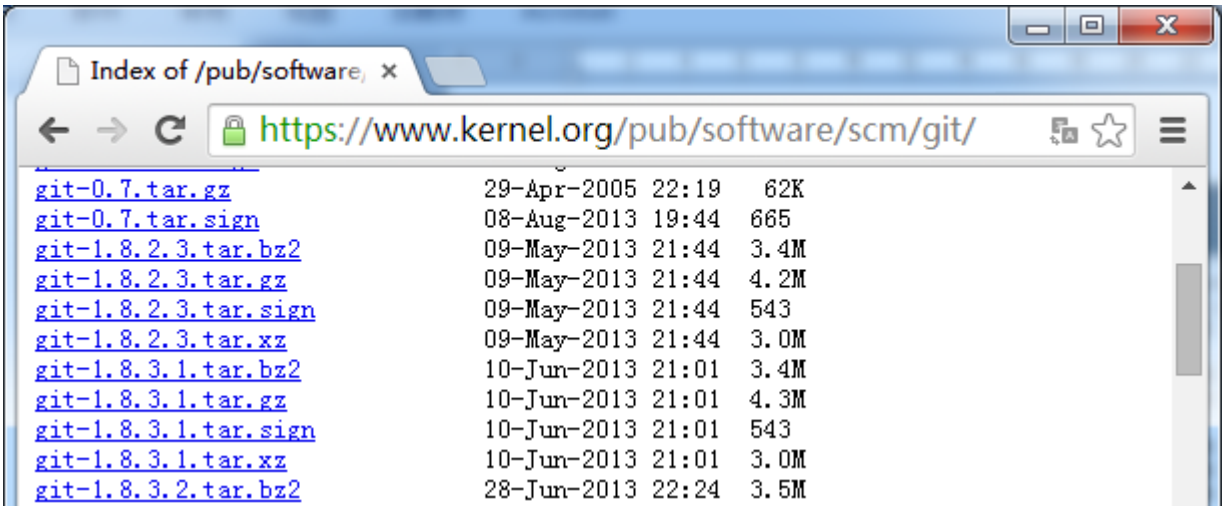
Spark 可以通过 SBT 和 Maven 两种方式进行编译，再通过 make-distribution.sh 脚本生成部署包。SBT 编译需要安装 git 工具，而 Maven 安装则需要 maven 工具，两种方式均需要在联网下进行，通过比较发现 SBT 编译速度较慢（原因有可能是 1、时间不一样，SBT 是白天编译，Maven 是深夜进行的，获取依赖包速度不同 2、maven 下载大文件是多线程进行，而 SBT 是单进程），Maven 编译成功前后花了 3、4 个小时。

1.1 编译 Spark（SBT）

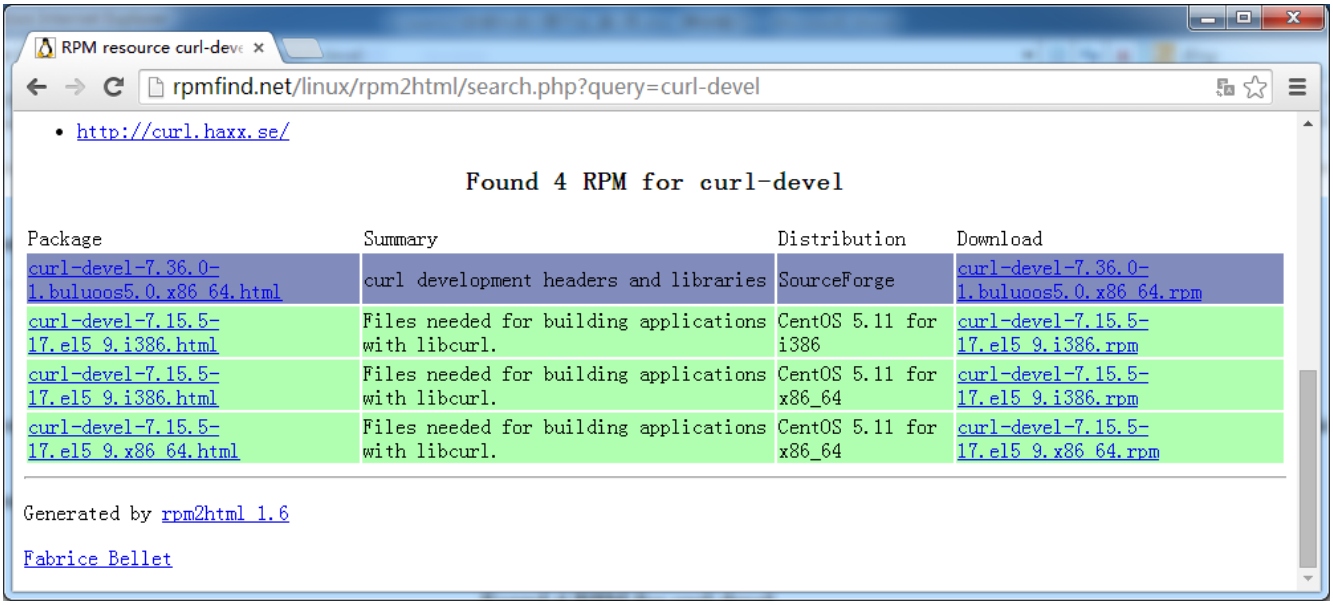
1.1.1 安装 git 并编译安装

1. 从如下地址下载 git 安装包

http://www.onlinedown.net/softdown/169333_2.htm
https://www.kernel.org/pub/software/scm/git/
如果 linux 是 CentOS 操作系统可以通过：yum install git 直接进行安装



由于从 https 获取内容，需要安装 curl-devel，可以从如下地址获取
http://rpmfind.net/linux/rpm2html/search.php?query=curl-devel
如果 linux 是 CentOS 操作系统可以通过：yum install curl-devel 直接进行安装



2. 上传 git 并解压缩

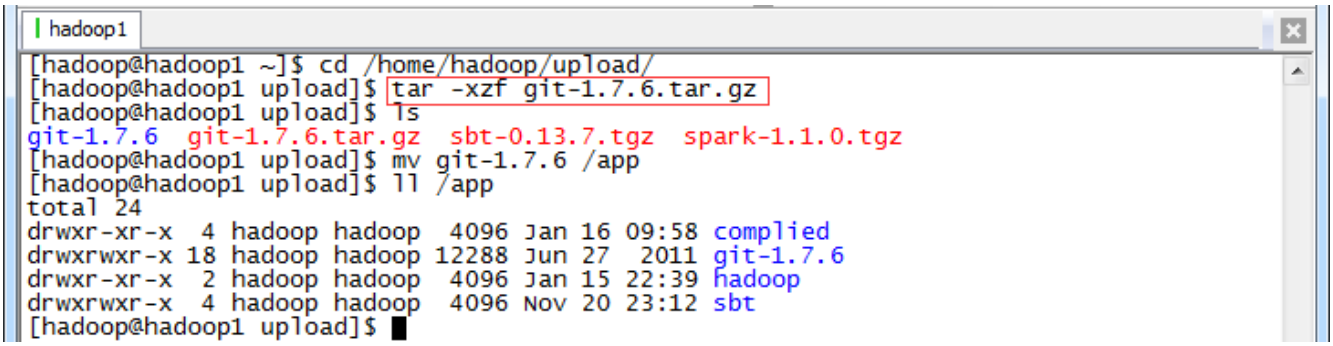
把 git-1.7.6.tar.gz 安装包上传到/home/hadoop/upload 目录中，解压缩然后放到/app 目录下

$cd /home/hadoop/upload/

$tar -xzf git-1.7.6.tar.gz

$mv git-1.7.6 /app

$ll /app



3. 编译安装 git

以 root 用户进行在 git 所在路径编译安装 git

#yum install curl-devel

#cd /app/git-1.7.6

#./configure

#make

#make install

```
hadoop1
[hadoop@hadoop1 ~]$ su
Password:
[root@hadoop1 hadoop]# yum install curl-devel
Loaded plugins: fastestmirror, refresh-packagekit, security
Loading mirror speeds from cached hostfile
 * base: mirrors.btte.net
 * extras: mirrors.aliyun.com
 * updates: mirror.neu.edu.cn
base                                                     | 3.7 kB     00:00
extras                                                   | 3.4 kB     00:00
updates                                                  | 3.4 kB     00:00
Setting up Install Process
```

```
hadoop1
[root@hadoop1 ~]# cd /app/git-1.7.6
[root@hadoop1 git-1.7.6]# ll configure
-rwxrwxr-x 1 hadoop hadoop 212688 Jun 27  2011 configure
[root@hadoop1 git-1.7.6]# ./configure
configure: Setting lib to 'lib' (the default)
configure: Will try -pthread then -lpthread to enable POSIX Threads.
configure: CHECKS for site configuration
configure: CHECKS for programs
checking for cc... cc
checking whether the C compiler works... yes
checking for C compiler default output file name... a.out
[root@hadoop1 git-1.7.6]# make
GIT_VERSION = 1.7.6
    * new build flags or prefix
    CC daemon.o
    CC abspath.o
    CC advice.o
    CC alias.o
    CC alloc.o
    CC archive.o
[root@hadoop1 git-1.7.6]# make install
    SUBDIR gitweb
    SUBDIR ../
make[2]: `GIT-VERSION-FILE' is up to date.
    GEN git-instaweb
    SUBDIR git-gui
    SUBDIR gitk-git
```

4. 把 git 加入到 PATH 路径中

打开/etc/profile 把 git 所在路径加入到 PATH 参数中

export GIT_HOME=/app/git-1.7.6

export PATH=$PATH:$JAVA_HOME/bin:$MAVEN_HOME/bin:$GIT_HOME/bin

```
hadoop1
export JAVA_HOME=/usr/lib/java/jdk1.7.0_55
export MAVEN_HOME=/app/apache-maven-3.0.5
export GIT_HOME=/app/git-1.7.6
export PATH=$PATH:$JAVA_HOME/bin:$MAVEN_HOME/bin:$GIT_HOME/bin
export CLASSPATH=.:$JAVA_HOME/lib/dt.jar:$JAVA_HOME/lib/tools.jar
```

重新登录或者使用 source /etc/profile 使参数生效，然后使用 git 命令查看配置是否正确

```
hadoop1
[root@hadoop1 ~]# git
usage: git [--version] [--exec-path[=<path>]] [--html-path] [--man-path] [--info-path]
           [-p|--paginate|--no-pager] [--no-replace-objects]
           [--bare] [--git-dir=<path>] [--work-tree=<path>]
           [-c name=value] [--help]
           <command> [<args>]

The most commonly used git commands are:
   add        Add file contents to the index
   bisect     Find by binary search the change that introduced a bug
   branch     List, create, or delete branches
```

1.1.2 下载 Spark 源代码并上传

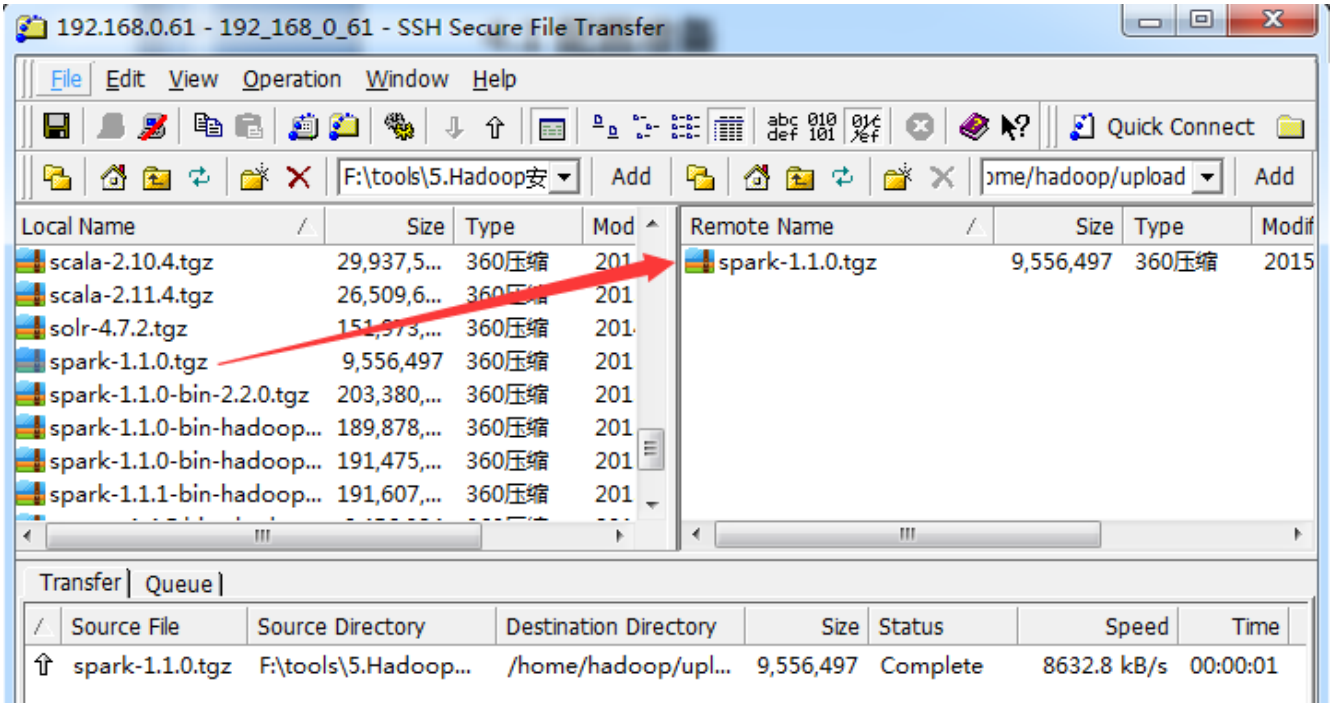
1. 可以从如下地址下载到 spark 源代码：

   http://spark.apache.org/downloads.html

   http://d3kbcqa49mib13.cloudfront.net/spark-1.1.0.tgz

   git clone https://github.com/apache/spark.git

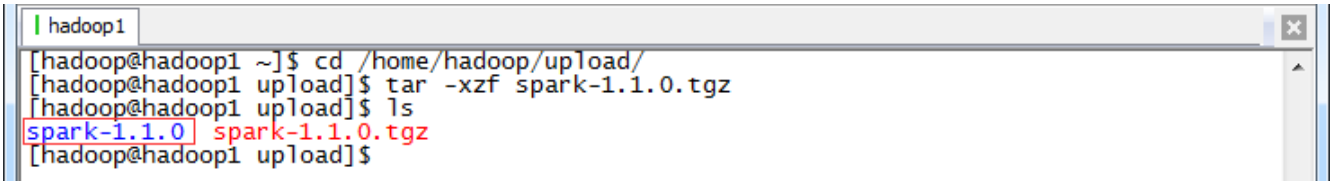
   把下载好的 spark-1.1.0.tgz 源代码包使用 1.1.3.1 介绍的工具上传到 /home/hadoop/upload 目录下



2. 在主节点上解压缩

$cd /home/hadoop/upload/

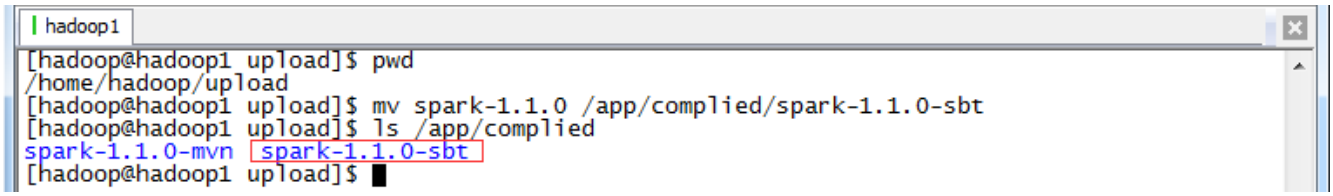
$tar -xzf spark-1.1.0.tgz



3. 把 spark-1.1.0 改名并移动到/app/complied 目录下

$mv spark-1.1.0 /app/complied/spark-1.1.0-sbt

$ls /app/complied



1.1.3 编译代码

编译 spark 源代码的时候，需要从网上下载依赖包，所以整个编译过程机器必须保证在联网状态。编译执行如下脚本：

$cd /app/complied/spark-1.1.0-sbt

$sbt/sbt assembly -Pyarn -Phadoop-2.2 -Pspark-ganglia-lgpl -Pkinesis-asl -Phive

```
hadoop1
[hadoop@hadoop1 sbt]$
[hadoop@hadoop1 sbt]$ cd /app/complied/spark-1.1.0-sbt
[hadoop@hadoop1 spark-1.1.0-sbt]$ sbt/sbt assembly -Pyarn -Phadoop-2.2 -Pspark-ganglia-lgpl -Pkinesis-asl -Phive
Using /usr/lib/java/jdk1.7.0_55 as default JAVA_HOME.
Note, this will be overridden by -java-home if it is set.
Getting org.scala-sbt sbt 0.13.5 ...
downloading https://repo.typesafe.com/typesafe/ivy-releases/org.scala-sbt/sbt/0.13.5/jars/sbt.jar ...
        [SUCCESSFUL ] org.scala-sbt#sbt;0.13.5!sbt.jar (4992ms)
downloading https://repo1.maven.org/maven2/org/scala-lang/scala-library/2.10.4/scala-library-2.10.4.jar ...
downloading https://repo.typesafe.com/typesafe/ivy-releases/org.scala-sbt/main/0.13.5/jars/main.jar ...
        [SUCCESSFUL ] org.scala-sbt#main;0.13.5!main.jar (205168ms)
downloading https://repo.typesafe.com/typesafe/ivy-releases/org.scala-sbt/compiler-interface/0.13.5/jars/compiler-interface-src.jar ...
        [SUCCESSFUL ] org.scala-sbt#compiler-interface;0.13.5!compiler-interface-src.jar (7062ms)
downloading https://repo.typesafe.com/typesafe/ivy-releases/org.scala-sbt/compiler-interface/0.13.5/jars/compiler-interface-bin.jar ...
        [SUCCESSFUL ] org.scala-sbt#compiler-interface;0.13.5!compiler-interface-bin.jar (24150ms)
downloading https://repo.typesafe.com/typesafe/ivy-releases/org.scala-sbt/precompiled-2_8_2/0.13.5/jars/compiler-interface-bin.jar ...
        [SUCCESSFUL ] org.scala-sbt#precompiled-2_8_2;0.13.5!compiler-interface-bin.jar (35281ms)
downloading https://repo.typesafe.com/typesafe/ivy-releases/org.scala-sbt/precompiled-2_9_2/0.13.5/jars/compiler-interface-bin.jar ...
```

```
[info] Done updating.
[info] Compiling 9 Scala sources to /home/hadoop/.sbt/0.13/staging/ec3aa8f39111944cc5f2/sbt-pom-reader/target/scala-2.10/sbt-0.13/classes...
[warn] there were 1 deprecation warning(s); re-run with -deprecation for details
[warn] one warning found
[info] Compiling 1 Scala source to /app/complied/spark-1.1.0-sbt/project/spark-style/target/scala-2.10/classes...
[info] Compiling 3 Scala sources to /app/complied/spark-1.1.0-sbt/project/target/scala-2.10/sbt-0.13/classes...
[warn] there were 1 deprecation warning(s); re-run with -deprecation for details
[warn] one warning found
[info] Set current project to spark-parent (in build file:/app/complied/spark-1.1.0-sbt/)
[info] Updating {file:/app/complied/spark-1.1.0-sbt/}core...
[info] Resolving org.apache.hadoop#hadoop-main;2.2.0 ...
```

```
[warn] Merging 'parquet/schema/Type.class' with strategy 'first'
[warn] Merging 'parquet/schema/TypeConverter.class' with strategy 'first'
[warn] Merging 'parquet/schema/TypeVisitor.class' with strategy 'first'
[warn] Merging 'plugin.properties' with strategy 'first'
[warn] Merging 'plugin.xml' with strategy 'first'
[warn] Merging 'reference.conf' with strategy 'concat'
[warn] Merging 'rootdoc.txt' with strategy 'first'
[warn] Strategy 'concat' was applied to a file
[warn] Strategy 'discard' was applied to 1722 files
[warn] Strategy 'filterDistinctLines' was applied to 6 files
[warn] Strategy 'first' was applied to 2504 files
[info] SHA-1: 1163013000630f1eb073fd5fdf422f90fc4627b4
[info] Packaging /app/complied/spark-1.1.0-sbt/assembly/target/scala-2.10/spark-assembly-1.1.0-hadoop1.0.4.jar ...
[info] Done packaging.
[success] Total time: 3082 s, completed Jan 17, 2015 1:35:16 AM
```

整个编译过程编译了约十几个任务，重新编译 N 次，需要几个甚至十几个小时才能编译完成（主要看下载依赖包的速度）。

1.2 编译 Spark（Maven）

1.2.1 安装 Maven 并配置参数

在编译前最好安装 3.0 以上版本的 Maven，在/etc/profile 配置文件中加入如下设置：

export MAVEN_HOME=/app/apache-maven-3.0.5

export PATH=$PATH:$JAVA_HOME/bin:$MAVEN_HOME/bin:$GIT_HOME/bin

```
hadoop1

export JAVA_HOME=/usr/lib/java/jdk1.7.0_55
export MAVEN_HOME=/app/apache-maven-3.0.5
export GIT_HOME=/app/git-1.7.6
export PATH=$PATH:$JAVA_HOME/bin:$MAVEN_HOME/bin:$GIT_HOME/bin
export CLASSPATH=.:$JAVA_HOME/lib/dt.jar:$JAVA_HOME/lib/tools.jar
```

1.2.2 下载 Spark 源代码并上传

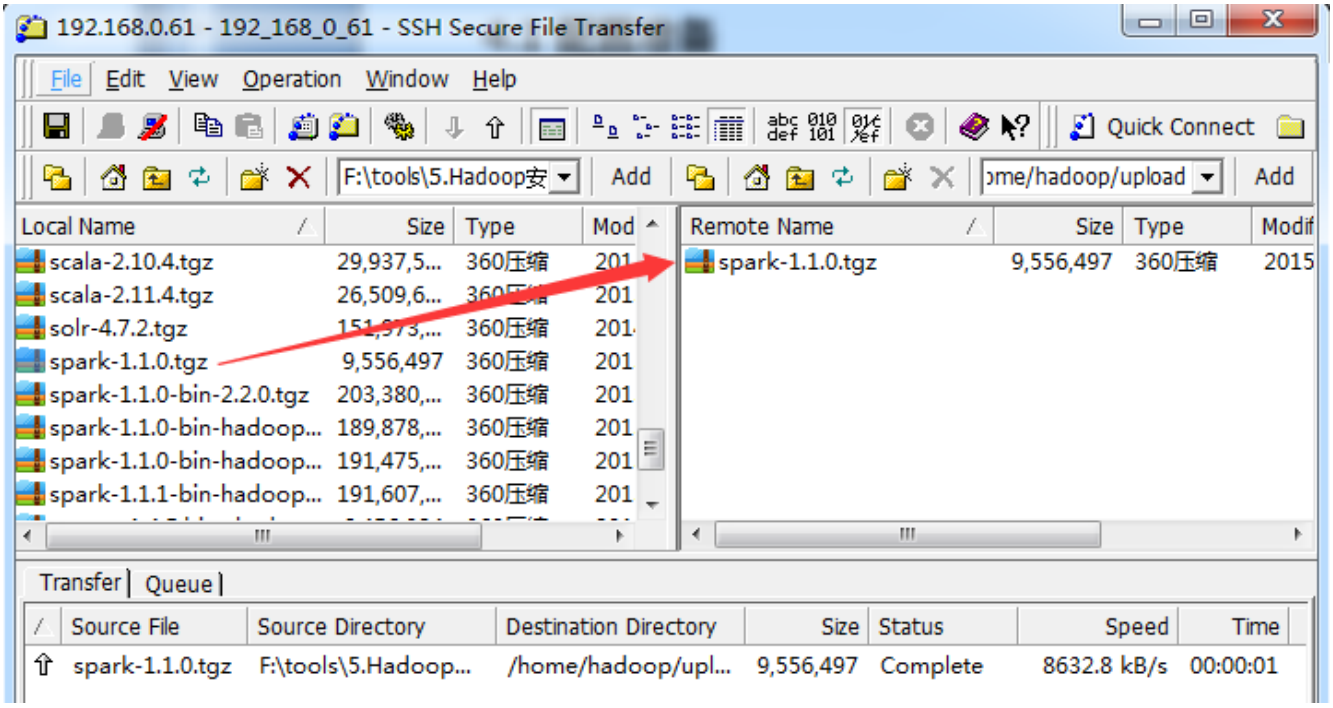
1. 可以从如下地址下载到 spark 源代码：

   http://spark.apache.org/downloads.html

   http://d3kbcqa49mib13.cloudfront.net/spark-1.1.0.tgz

   git clone https://github.com/apache/spark.git

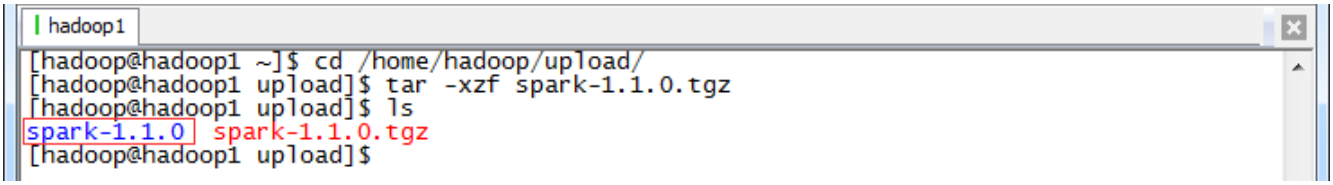
   把下载好的 spark-1.1.0.tgz 源代码包使用 1.1.3.1 介绍的工具上传到 /home/hadoop/upload 目录下



2. 在主节点上解压缩

$cd /home/hadoop/upload/

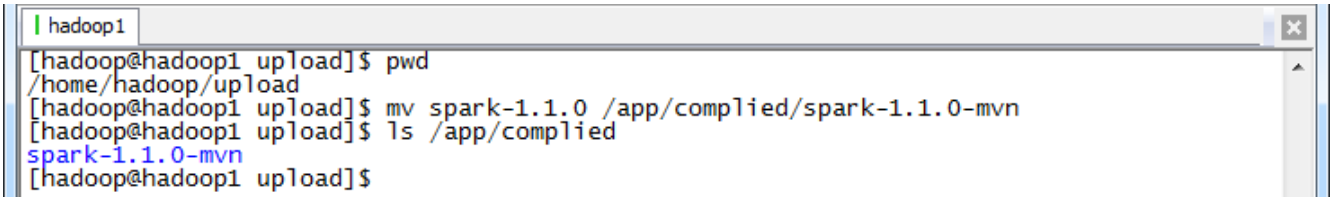
$tar -xzf spark-1.1.0.tgz



3. 把 spark-1.1.0 改名并移动到/app/complied 目录下

$mv spark-1.1.0 /app/complied/spark-1.1.0-mvn

$ls /app/complied



1.2.3 编译代码

编译 spark 源代码的时候，需要从网上下载依赖包，所以整个编译过程机器必须保证在联网状态。编译执行如下脚本：

$cd /app/complied/spark-1.1.0-mvn

$export MAVEN_OPTS="-Xmx2g -XX:MaxPermSize=512M -XX:ReservedCodeCacheSize=512m"

$mvn -Pyarn -Phadoop-2.2 -Pspark-ganglia-lgpl -Pkinesis-asl -Phive -DskipTests clean package

```
hadoop1
[hadoop@hadoop1 ~]$ cd /app/complied/spark-1.1.0-mvn
[hadoop@hadoop1 spark-1.1.0-mvn]$ export MAVEN_OPTS="-Xmx2g -XX:MaxPermSize=512M -XX:ReservedCodeCacheSize=512m"
[hadoop@hadoop1 spark-1.1.0-mvn]$ mvn -Pyarn -Phadoop-2.2 -Pspark-ganglia-lgpl -Pkinesis-asl -Phive -DskipTests clean package
[INFO] Scanning for projects...
[INFO] ------------------------------------------------------------------------
[INFO] Reactor Build Order:
[INFO]
[INFO] Spark Project Parent POM
[INFO] Spark Project Core
[INFO] Spark Project Bagel
[INFO] Spark Project GraphX
[INFO] Spark Project Streaming
[INFO] Spark Project ML Library
[INFO] Spark Project Tools
[INFO] Spark Project Catalyst
[INFO] Spark Project SQL
[INFO] Spark Project Hive
[INFO] Spark Project REPL
[INFO] Spark Project YARN Parent POM
[INFO] Spark Project YARN Stable API
[INFO] Spark Project Hive Thrift Server
[INFO] Spark Ganglia Integration
[INFO] Spark Project Assembly
[INFO] Spark Project External Twitter
[INFO] Spark Project External Kafka
```

整个编译过程编译了约 24 个任务，整个过程耗时 1 小时 45 分钟。

```
hadoop1
[INFO] ------------------------------------------------------------------------
[INFO] Reactor Summary:
[INFO]
[INFO] Spark Project Parent POM ........................... SUCCESS [17:04.571s]
[INFO] Spark Project Core ................................. SUCCESS [28:13.189s]
[INFO] Spark Project Bagel ................................ SUCCESS [1:22.231s]
[INFO] Spark Project GraphX ............................... SUCCESS [5:55.681s]
[INFO] Spark Project Streaming ............................ SUCCESS [5:04.976s]
[INFO] Spark Project ML Library ........................... SUCCESS [7:44.069s]
[INFO] Spark Project Tools ................................ SUCCESS [45.759s]
[INFO] Spark Project Catalyst ............................. SUCCESS [4:41.588s]
[INFO] Spark Project SQL .................................. SUCCESS [5:08.278s]
[INFO] Spark Project Hive ................................. SUCCESS [6:27.472s]
[INFO] Spark Project REPL ................................. SUCCESS [2:39.759s]
[INFO] Spark Project YARN Parent POM ...................... SUCCESS [18.324s]
[INFO] Spark Project YARN Stable API ...................... SUCCESS [2:33.087s]
[INFO] Spark Project Hive Thrift Server ................... SUCCESS [1:34.832s]
[INFO] Spark Ganglia Integration .......................... SUCCESS [17.066s]
[INFO] Spark Project Assembly ............................. SUCCESS [52.689s]
[INFO] Spark Project External Twitter ..................... SUCCESS [50.959s]
[INFO] Spark Project External Kafka ....................... SUCCESS [1:22.130s]
[INFO] Spark Project External Flume Sink .................. SUCCESS [1:20.157s]
[INFO] Spark Project External Flume ....................... SUCCESS [1:32.762s]
[INFO] Spark Project External ZeroMQ ...................... SUCCESS [1:08.724s]
[INFO] Spark Project External MQTT ........................ SUCCESS [1:20.290s]
[INFO] Spark Kinesis Integration .......................... SUCCESS [3:10.133s]
[INFO] Spark Project Examples ............................. SUCCESS [4:08.258s]
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 1:45:39.285s
[INFO] Finished at: Fri Jan 16 01:11:02 CST 2015
[INFO] Final Memory: 53M/497M
[INFO] ------------------------------------------------------------------------
```

1.3 生成 Spark 部署包

在 Spark 源码根目录下有一个生成部署包的脚本 make-distribution.sh，可以通过执行如下命令进行打包 *./make-distribution.sh [--name] [--tgz] [--with-tachyon] <maven build options>*

- **--name NAME** 和--tgz 结合可以生成 spark-$VERSION-bin-$NAME.tgz 的部署包，不加此参数时 NAME 为 hadoop 的版本号
- **--tgz** 在根目录下生成 spark-$VERSION-bin.tgz ，不加此参数时不生成 tgz 文件，只生成/dist 目录
- **--with-tachyon** 是否支持内存文件系统 Tachyon ，不加此参数时不支持 tachyon

```
hadoop1
[hadoop@hadoop1 ~]$ cd /app/complied/spark-1.1.0-mvn/
[hadoop@hadoop1 spark-1.1.0-mvn]$ ls
assembly     docker       LICENSE                          README.md              target
bagel        docs         make-distribution.sh             repl                   tools
bin          ec2          maven-remote-resources-plugin-1.5.pom  sbin             tox.ini
CHANGES.txt  examples     mllib                            sbt                    yarn
conf         external     NOTICE                           scalastyle-config.xml
core         extras       pom.xml                          scalastyle-output.xml
data         graphx       project                          sql
dev          lib_managed  python                           streaming
[hadoop@hadoop1 spark-1.1.0-mvn]$
```

例子：

1. 生成支持 yarn 、hadoop2.2.0 、hive 的部署包：

./make-distribution.sh --tgz --name 2.2.0 -Pyarn -Phadoop-2.2 -Phive

2. 生成支持 yarn 、hadoop2.2.0 、hive 、ganglia 的部署包：

./make-distribution.sh --tgz --name 2.2.0 -Pyarn -Phadoop-2.2 -Pspark-ganglia-lgpl -P hive

1.3.1 生成部署包

使用如下命令生成 Spark 部署包，由于该脚本默认在 JDK1.6 进行，在开始时会进行询问是否继续，只要选择 Y 即可

$cd /app/complied/spark-1.1.0-mvn/

$./make-distribution.sh --tgz --name 2.2.0 -Pyarn -Phadoop-2.2 -Pspark-ganglia-lgpl -P hive

```
hadoop1
[hadoop@hadoop1 ~]$ cd /app/complied/spark-1.1.0-mvn/
[hadoop@hadoop1 spark-1.1.0-mvn]$ ./make-distribution.sh --tgz --name 2.2.0 -Pyarn -Phadoop-2.2 -Pspark-ganglia-lgpl -P hive
***NOTE***: JAVA_HOME is not set to a JDK 6 installation. The resulting
            distribution may not work well with PySpark and will not run
            with Java 6 (See SPARK-1703 and SPARK-1911).
            This test can be disabled by adding --skip-java-test.
Output from 'java -version' was:
java version "1.7.0_55"
Java(TM) SE Runtime Environment (build 1.7.0_55-b13)
Java HotSpot(TM) 64-Bit Server VM (build 24.55-b03, mixed mode)
Would you like to continue anyways? [y,n]: y
Spark version is 1.1.0
Making spark-1.1.0-bin-2.2.0.tgz
Tachyon Disabled
[INFO] ------------------------------------------------------------------------
[INFO] Building Spark Project Bagel 1.1.0
[INFO] ------------------------------------------------------------------------
[INFO]
[INFO] --- maven-clean-plugin:2.5:clean (default-clean) @ spark-bagel_2.10 ---
[INFO] Deleting /app/complied/spark-1.1.0-mvn/bagel/target
[INFO]
[INFO] --- maven-enforcer-plugin:1.3.1:enforce (enforce-versions) @ spark-bagel_2.10 ---
[INFO]
[INFO] --- build-helper-maven-plugin:1.8:add-source (add-scala-sources) @ spark-bagel_2.10 ---
[INFO] Source directory: /app/complied/spark-1.1.0-mvn/bagel/src/main/scala added.
[INFO]
[INFO] --- maven-remote-resources-plugin:1.5:process (default) @ spark-bagel_2.10 ---
[INFO]
[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ spark-bagel_2.10 ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] skip non existing resourceDirectory /app/complied/spark-1.1.0-mvn/bagel/src/main/resources
[INFO] Copying 3 resources
[INFO]
```

```
hadoop1 | hadoop1 (1)
[INFO] ------------------------------------------------------------------------
[INFO] Reactor Summary:
[INFO]
[INFO] Spark Project Parent POM ........................... SUCCESS [11:32.891s]
[INFO] Spark Project Core ................................. SUCCESS [25:52.315s]
[INFO] Spark Project Bagel ................................ SUCCESS [1:24.230s]
[INFO] Spark Project GraphX ............................... SUCCESS [7:12.380s]
[INFO] Spark Project Streaming ............................ SUCCESS [5:07.506s]
[INFO] Spark Project ML Library ........................... SUCCESS [9:31.010s]
[INFO] Spark Project Tools ................................ SUCCESS [44.548s]
[INFO] Spark Project Catalyst ............................. SUCCESS [4:21.802s]
[INFO] Spark Project SQL .................................. SUCCESS [6:15.114s]
[INFO] Spark Project Hive ................................. SUCCESS [6:31.824s]
[INFO] Spark Project REPL ................................. SUCCESS [2:29.144s]
[INFO] Spark Project YARN Parent POM ...................... SUCCESS [9.654s]
[INFO] Spark Project YARN Stable API ...................... SUCCESS [2:07.880s]
[INFO] Spark Project Hive Thrift Server ................... SUCCESS [1:30.351s]
[INFO] Spark Ganglia Integration .......................... SUCCESS [18.345s]
[INFO] Spark Project Assembly ............................. SUCCESS [44.294s]
[INFO] Spark Project External Twitter ..................... SUCCESS [59.895s]
[INFO] Spark Project External Kafka ....................... SUCCESS [1:28.462s]
[INFO] Spark Project External Flume Sink .................. SUCCESS [1:37.191s]
[INFO] Spark Project External Flume ....................... SUCCESS [1:28.526s]
[INFO] Spark Project External ZeroMQ ...................... SUCCESS [1:12.855s]
[INFO] Spark Project External MQTT ........................ SUCCESS [1:22.810s]
[INFO] Spark Project Examples ............................. SUCCESS [4:12.026s]
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 1:38:17.675s
[INFO] Finished at: Sun Jan 18 00:56:46 CST 2015
[INFO] Final Memory: 52M/455M
[INFO] ------------------------------------------------------------------------
```

生成 Spark 部署包编译了约 24 个任务，用时大概 1 小时 38 分钟。

1.3.2 查看生成结果

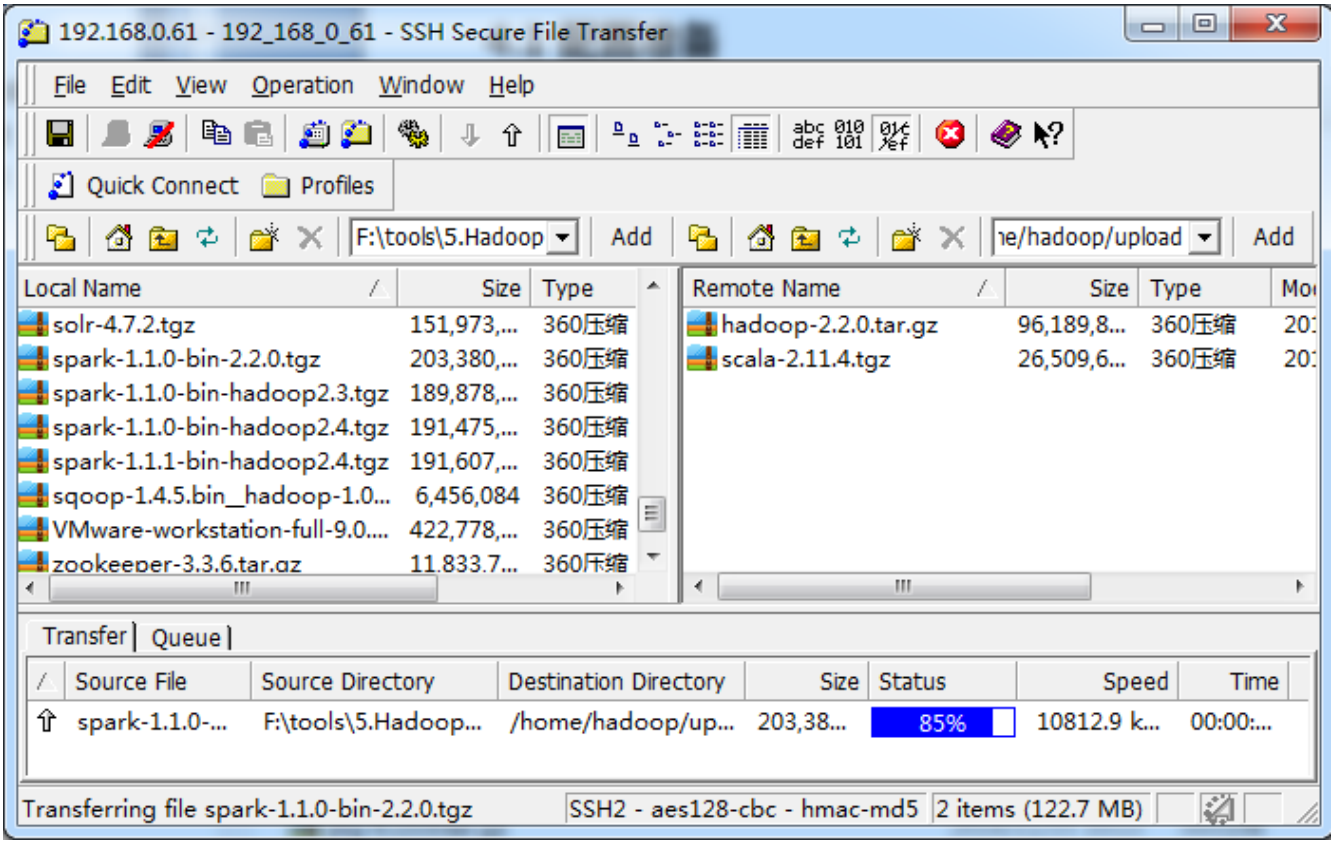
生成在部署包位于根目录下，文件名类似于 spark-1.1.0-bin-2.2.0.tgz。

```
hadoop1 | hadoop1 (1)
[hadoop@hadoop1 spark-1.1.0-mvn]$ ls
assembly     external                                repl
bagel        extras                                  sbin
bin          graphx                                  sbt
CHANGES.txt  lib_managed                             scalastyle-config.xml
conf         LICENSE                                 scalastyle-output.xml
core         make-distribution.sh                    spark-1.1.0-bin-2.2.0.tgz
data         maven-remote-resources-plugin-1.5.pom   sql
dev          mllib                                   streaming
dist         NOTICE                                  target
docker       pom.xml                                 tools
docs         project                                 tox.ini
ec2          python                                  yarn
examples     README.md
[hadoop@hadoop1 spark-1.1.0-mvn]$
```

2 安装 Spark

2.1 上传并解压 Spark 安装包

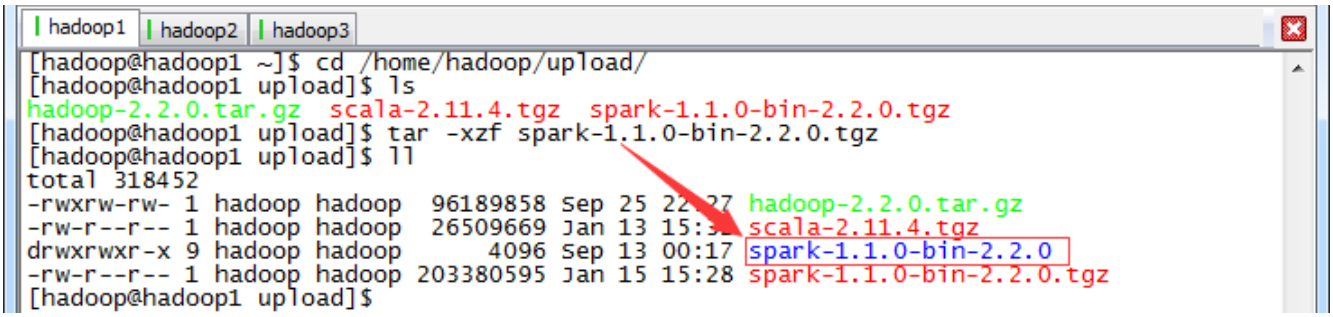
1. 我们使用上一步骤编译好的 spark-1.1.0-bin-2.2.0.tgz 文件作为安装包(也可以从网上下载 native 文件夹或者打包好的 64 位 hadoop 安装包) ,使用" Spark 编译与部署(上)"中 1. 3.1 介绍的工具上传到/home/hadoop/upload 目录下



2. 在主节点上解压缩

$cd /home/hadoop/upload/

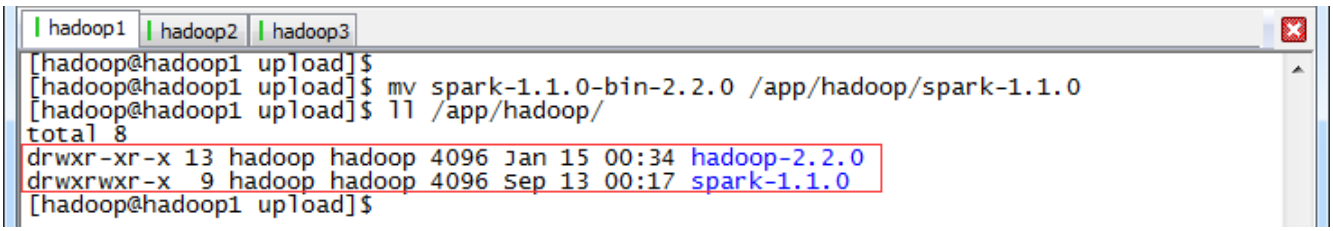
$tar -xzf spark-1.1.0-bin-2.2.0.tgz



3. 把 spark 改名并移动到/app/hadoop 目录下

$mv spark-1.1.0-bin-2.2.0 /app/hadoop/spark-1.1.0

$ll /app/hadoop



2.2 配置/etc/profile

1. 打开配置文件/etc/profile

$sudo vi /etc/profile

2. 定义 SPARK_HOME 并把 spark 路径加入到 PATH 参数中

SPARK_HOME=/app/hadoop/spark-1.1.0
PATH=$PATH:$SPARK_HOME/bin:$SPARK_HOME/sbin

2.3 配置 conf/slaves

1. 打开配置文件 conf/slaves

$cd /app/hadoop/spark-1.1.0/conf
$sudo vi slaves

```
hadoop1  hadoop2  hadoop3
[hadoop@hadoop1 ~]$ cd /app/hadoop/spark-1.1.0/conf
[hadoop@hadoop1 conf]$ ls
fairscheduler.xml.template  metrics.properties.template  spark-defaults.conf.template
log4j.properties.template   slaves                        spark-env.sh.template
[hadoop@hadoop1 conf]$ sudo vi slaves
```

2. 加入 slave 配置节点

hadoop1
hadoop2
hadoop3

```
hadoop1  hadoop2  hadoop3
# A Spark Worker will be started on each of the machines listed below.
hadoop1
hadoop2
hadoop3
~
```

2.4 配置 conf/spark-env.sh

1. 打开配置文件 conf/spark-env.sh

$cd /app/hadoop/spark-1.1.0/conf
$cp spark-env.sh.template spark-env.sh
$sudo vi spark-env.sh

```
hadoop1 | hadoop2 | hadoop3
[hadoop@hadoop1 ~]$ cd /app/hadoop/spark-1.1.0/conf
[hadoop@hadoop1 conf]$ ls
fairscheduler.xml.template  metrics.properties.template  spark-defaults.conf.template
log4j.properties.template   slaves                        spark-env.sh.template
[hadoop@hadoop1 conf]$ cp spark-env.sh.template spark-env.sh
[hadoop@hadoop1 conf]$ ls
fairscheduler.xml.template   slaves                        spark-env.sh.template
log4j.properties.template    spark-defaults.conf.template
metrics.properties.template  spark-env.sh
[hadoop@hadoop1 conf]$
```

2. 加入 Spark 环境配置内容，设置 hadoop1 为 Master 节点

export SPARK_MASTER_IP=hadoop1

export SPARK_MASTER_PORT=7077

export SPARK_WORKER_CORES=1

export SPARK_WORKER_INSTANCES=1

export SPARK_WORKER_MEMORY=512M

```
hadoop1 | hadoop2 | hadoop3
# - SPARK_WORKER_DIR, to set the working directory of worker processes
# - SPARK_WORKER_OPTS, to set config properties only for the worker (e.g. "-Dx=y")
# - SPARK_HISTORY_OPTS, to set config properties only for the history server (e.g. "-Dx=y")
# - SPARK_DAEMON_JAVA_OPTS, to set config properties for all daemons (e.g. "-Dx=y")
# - SPARK_PUBLIC_DNS, to set the public dns name of the master or workers

export SPARK_MASTER_IP=hadoop1
export SPARK_MASTER_PORT=7077
export SPARK_WORKER_CORES=1
export SPARK_WORKER_INSTANCES=1
export SPARK_WORKER_MEMORY=512M
```

2.5 向各节点分发 Spark 程序

1. 进入 hadoop1 机器/app/hadoop 目录，使用如下命令把 spark 文件夹复制到 hadoop2 和 hadoop3 机器

$cd /app/hadoop

$scp -r spark-1.1.0 hadoop@hadoop2:/app/hadoop/

$scp -r spark-1.1.0 hadoop@hadoop3:/app/hadoop/

```
hadoop1 | hadoop2 | hadoop3
[hadoop@hadoop1 ~]$ cd /app/hadoop
[hadoop@hadoop1 hadoop]$ ll
total 8
drwxr-xr-x 13 hadoop hadoop 4096 Jan 15 00:34 hadoop-2.2.0
drwxrwxr-x  9 hadoop hadoop 4096 Sep 13 00:17 spark-1.1.0
[hadoop@hadoop1 hadoop]$ scp -r spark-1.1.0 hadoop@hadoop2:/app/hadoop/
```

```
py4j_callback_example.py                          100%  648     0.6KB/s   00:00
__init__.py                                       100%    0     0.0KB/s   00:00
java_set_test.py                                  100% 3730     3.6KB/s   00:00
java_gateway_test.py                              100%   24KB  24.4KB/s   00:00
py4j_callback_example2.py                         100%  398     0.4KB/s   00:00
java_array_test.py                                100% 2036     2.0KB/s   00:00
java_list_test.py                                 100%   10KB  10.5KB/s   00:00
byte_string_test.py                               100% 1234     1.2KB/s   00:00
java_callback_test.py                             100% 7634     7.5KB/s   00:00
multithreadtest.py                                100% 3261     3.2KB/s   00:00
java_map_test.py                                  100% 2941     2.9KB/s   00:00
py4j_example.py                                   100%  325     0.3KB/s   00:00
finalizer_test.py                                 100% 4356     4.3KB/s   00:00
java_collections.py                               100%   17KB  16.8KB/s   00:00
run-tests                                         100% 3018     3.0KB/s   00:00
py4j-0.8.2.1-src.zip                              100%   37KB  36.7KB/s   00:00
PY4J_LICENSE.txt                                  100% 1445     1.4KB/s   00:00
spark-assembly-1.1.0-hadoop2.2.0.jar               32%   43MB   8.4MB/s   00:10 ETA
```

2. 在从节点查看是否复制成功

```
hadoop1  hadoop2  hadoop3
[hadoop@hadoop2 ~]$ cd /app/hadoop/
[hadoop@hadoop2 hadoop]$ ll
total 8
drwxr-xr-x 13 hadoop hadoop 4096 Jan 15 00:34 hadoop-2.2.0
drwxrwxr-x  9 hadoop hadoop 4096 Jan 15 15:56 spark-1.1.0
[hadoop@hadoop2 hadoop]$ cd spark-1.1.0/
[hadoop@hadoop2 spark-1.1.0]$ ls
bin          conf   examples  LICENSE   python     RELEASE
CHANGES.txt  ec2    lib       NOTICE    README.md  sbin
[hadoop@hadoop2 spark-1.1.0]$
```

2.6 启动 Spark

$cd /app/hadoop/spark-1.1.0/sbin

$./start-all.sh

```
hadoop1  hadoop2  hadoop3
[hadoop@hadoop1 ~]$ cd /app/hadoop/spark-1.1.0/sbin
[hadoop@hadoop1 sbin]$ ls
slaves.sh        spark-daemons.sh  start-history-server.sh  start-slaves.sh        stop-history-server.sh
spark-config.sh  spark-executor    start-master.sh          start-thriftserver.sh  stop-master.sh
spark-daemon.sh  start-all.sh      start-slave.sh           stop-all.sh            stop-slaves.sh
[hadoop@hadoop1 sbin]$ ./start-all.sh
starting org.apache.spark.deploy.master.Master, logging to /app/hadoop/spark-1.1.0/sbin/../logs/spark-hado
op-org.apache.spark.deploy.master.Master-1-hadoop1.out
hadoop1: starting org.apache.spark.deploy.worker.Worker, logging to /app/hadoop/spark-1.1.0/sbin/../logs/s
park-hadoop-org.apache.spark.deploy.worker.Worker-1-hadoop1.out
hadoop3: starting org.apache.spark.deploy.worker.Worker, logging to /app/hadoop/spark-1.1.0/sbin/../logs/s
park-hadoop-org.apache.spark.deploy.worker.Worker-1-hadoop3.out
hadoop2: starting org.apache.spark.deploy.worker.Worker, logging to /app/hadoop/spark-1.1.0/sbin/../logs/s
park-hadoop-org.apache.spark.deploy.worker.Worker-1-hadoop2.out
[hadoop@hadoop1 sbin]$
```

2.7 验证启动

此时在 hadoop1 上面运行的进程有：Worker 和 Master

```
hadoop1  hadoop2  hadoop3
[hadoop@hadoop1 sbin]$ jps
6568 Master
6709 Worker
6801 Jps
[hadoop@hadoop1 sbin]$
```

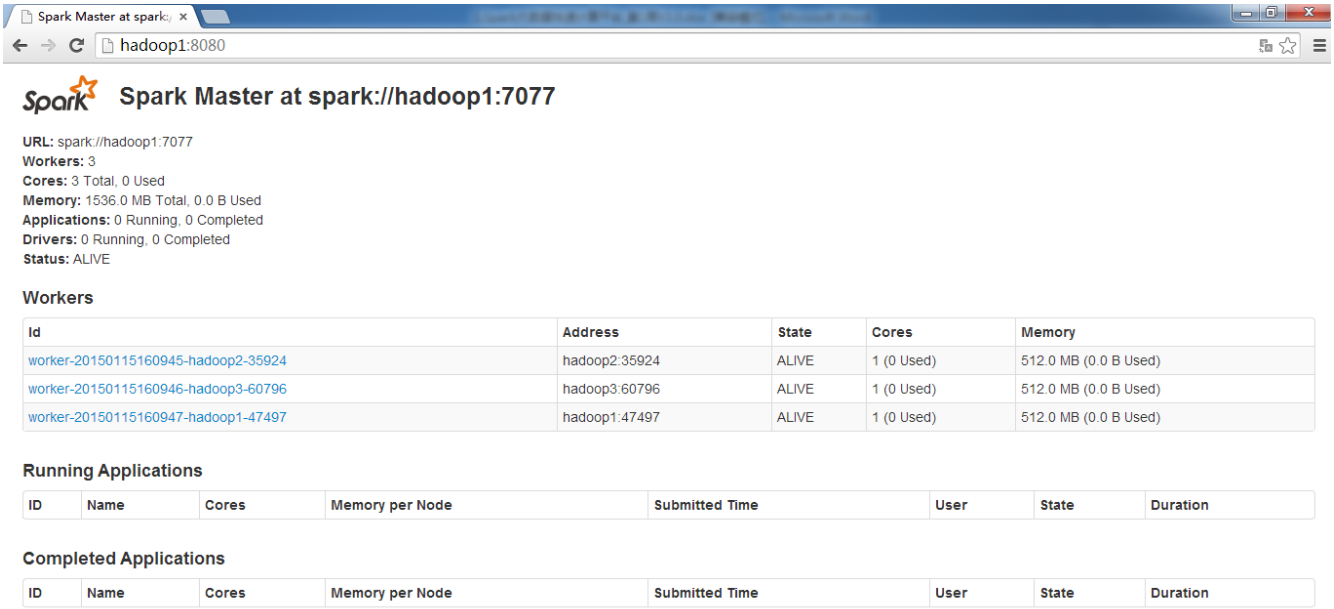
此时在 hadoop2 和 hadoop3 上面运行的进程有只有 Worker

```
hadoop1 | hadoop2 | hadoop3
[hadoop@hadoop2 ~]$ jps
4161 Worker
4202 Jps
[hadoop@hadoop2 ~]$
```

通过 netstat -nlt 命令查看 hadoop1 节点网络情况

```
hadoop1 | hadoop2 | hadoop3
[hadoop@hadoop1 ~]$ netstat -nlt
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address               Foreign Address             State
tcp        0      0 0.0.0.0:32972               0.0.0.0:*                   LISTEN
tcp        0      0 0.0.0.0:111                 0.0.0.0:*                   LISTEN
tcp        0      0 0.0.0.0:22                  0.0.0.0:*                   LISTEN
tcp        0      0 127.0.0.1:631               0.0.0.0:*                   LISTEN
tcp        0      0 127.0.0.1:25                0.0.0.0:*                   LISTEN
tcp        0      0 :::111                      :::*                        LISTEN
tcp        0      0 :::8080                     :::*                        LISTEN
tcp        0      0 :::8081                     :::*                        LISTEN
tcp        0      0 :::22                       :::*                        LISTEN
tcp        0      0 ::1:631                     :::*                        LISTEN
tcp        0      0 ::1:25                      :::*                        LISTEN
tcp        0      0 :::49915                    :::*                        LISTEN
tcp        0      0 ::ffff:10.88.147.221:7077   :::*                        LISTEN
tcp        0      0 ::ffff:10.88.147.221:47497  :::*                        LISTEN
```

在浏览器中输入 http://hadoop1:8080（需要注意的是要在网络设置中把 hadoop*除外，否则会到外网 DNS 解析，出现无法访问的情况） 既可以进入 Spark 集群状态页面



2.8 验证客户端连接

进入 hadoop1 节点，进入 spark 的 bin 目录，使用 spark-shell 连接集群

$cd /app/hadoop/spark-1.1.0/bin

$spark-shell --master spark://hadoop1:7077 --executor-memory 500m

```
hadoop1 | hadoop2 | hadoop3
[hadoop@hadoop1 bin]$ spark-shell --master spark://hadoop1:7077 --executor-memory 500m
Spark assembly has been built with Hive, including Datanucleus jars on classpath
Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties
15/01/15 16:32:37 INFO SecurityManager: Changing view acls to: hadoop,
15/01/15 16:32:37 INFO SecurityManager: Changing modify acls to: hadoop,
15/01/15 16:32:37 INFO SecurityManager: SecurityManager: authentication disabled; ui acls disabled; user
s with view permissions: Set(hadoop, ); users with modify permissions: Set(hadoop, )
15/01/15 16:32:37 INFO HttpServer: Starting HTTP Server
15/01/15 16:32:38 INFO Utils: Successfully started service 'HTTP class server' on port 32770.
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 1.1.0
      /_/

Using Scala version 2.10.4 (Java HotSpot(TM) 64-Bit Server VM, Java 1.7.0_55)
Type in expressions to have them evaluated.
Type :help for more information.
```

在命令中只指定了内存大小并没有指定核数，所以该客户端将占用该集群所有核并在每个节点分配 500M 内存

Spark Master at spark://hadoop1:7077

URL: spark://hadoop1:7077
Workers: 3
Cores: 3 Total, 3 Used
Memory: 1536.0 MB Total, 1500.0 MB Used
Applications: 1 Running, 0 Completed
Drivers: 0 Running, 0 Completed
Status: ALIVE

Workers

Id	Address	State	Cores	Memory
worker-20150115160945-hadoop2-35924	hadoop2:35924	ALIVE	1 (1 Used)	512.0 MB (500.0 MB Used)
worker-20150115160946-hadoop3-60796	hadoop3:60796	ALIVE	1 (1 Used)	512.0 MB (500.0 MB Used)
worker-20150115160947-hadoop1-47497	hadoop1:47497	ALIVE	1 (1 Used)	512.0 MB (500.0 MB Used)

Running Applications

ID	Name	Cores	Memory per Node	Submitted Time	User	State	Duration
app-20150115163304-0000	Spark shell	3	500.0 MB	2015/01/15 16:33:04	hadoop	RUNNING	14 s

Executor Summary

ExecutorID	Worker	Cores	Memory	State	Logs
1	worker-20150115160946-hadoop3-60796	1	500	EXITED	stdout stderr
5	worker-20150115160945-hadoop2-35924	1	500	RUNNING	stdout stderr
2	worker-20150115160945-hadoop2-35924	1	500	EXITED	stdout stderr
0	worker-20150115160947-hadoop1-47497	1	500	RUNNING	stdout stderr
4	worker-20150115160946-hadoop3-60796	1	500	RUNNING	stdout stderr
3	worker-20150115160946-hadoop3-60796	1	500	EXITED	stdout stderr

3 Spark 测试

3.1 使用 Spark-shell 测试

这里我们测试一下在 Hadoop 中大家都知道的 WordCout 程序，在 MapReduce 实现 WordCout 需要 Map、Reduce 和 Job 三个部分，而在 Spark 中甚至一行就能够搞定。下面就看一下是如何实现的：

3.1.1 启动 HDFS

$cd /app/hadoop/hadoop-2.2.0/sbin

$./start-dfs.sh

```
hadoop1 | hadoop2 | hadoop3
[hadoop@hadoop1 ~]$
[hadoop@hadoop1 ~]$ cd /app/hadoop/hadoop-2.2.0/sbin
[hadoop@hadoop1 sbin]$ ./start-dfs.sh
Starting namenodes on [hadoop1]
hadoop1: starting namenode, logging to /app/hadoop/hadoop-2.2.0/logs/hadoop-hadoop-namenode-hadoop1.out
hadoop1: starting datanode, logging to /app/hadoop/hadoop-2.2.0/logs/hadoop-hadoop-datanode-hadoop1.out
hadoop2: starting datanode, logging to /app/hadoop/hadoop-2.2.0/logs/hadoop-hadoop-datanode-hadoop2.out
hadoop3: starting datanode, logging to /app/hadoop/hadoop-2.2.0/logs/hadoop-hadoop-datanode-hadoop3.out
Starting secondary namenodes [hadoop1]
hadoop1: starting secondarynamenode, logging to /app/hadoop/hadoop-2.2.0/logs/hadoop-hadoop-secondaryname
node-hadoop1.out
[hadoop@hadoop1 sbin]$
```

通过 jps 观察启动情况 ,在 hadoop1 上面运行的进程有 :NameNode、SecondaryNameNode 和 DataNode

```
hadoop1 | hadoop2 | hadoop3
[hadoop@hadoop1 sbin]$
[hadoop@hadoop1 sbin]$ jps
3280 DataNode
3189 NameNode
3426 SecondaryNameNode
3562 Jps
[hadoop@hadoop1 sbin]$
```

hadoop2 和 hadoop3 上面运行的进程有：NameNode 和 DataNode

```
hadoop1 | hadoop2 | hadoop3
[hadoop@hadoop2 ~]$ jps
3120 Jps
3044 DataNode
[hadoop@hadoop2 ~]$
```

3.1.2 上传数据到 HDFS 中

把 hadoop 配置文件 core-site.xml 文件作为测试文件上传到 HDFS 中

$hadoop fs -mkdir -p /user/hadoop/testdata

$hadoop fs -put /app/hadoop/hadoop-2.2.0/etc/hadoop/core-site.xml /user/hadoop/testdata

```
hadoop1 | hadoop2 | hadoop3
[hadoop@hadoop1 ~]$ hadoop fs -mkdir -p /user/hadoop/testdata
[hadoop@hadoop1 ~]$ hadoop fs -ls /user/hadoop
Found 1 items
drwxr-xr-x   - hadoop supergroup          0 2015-01-16 16:38 /user/hadoop/testdata
[hadoop@hadoop1 ~]$
[hadoop@hadoop1 ~]$ hadoop fs -put /app/hadoop/hadoop-2.2.0/etc/hadoop/core-site.xml /
user/hadoop/testdata
[hadoop@hadoop1 ~]$ hadoop fs -ls /user/hadoop/testdata
Found 1 items
-rw-r--r--   2 hadoop supergroup       1432 2015-01-16 16:40 /user/hadoop/testdata/core-site.xml
```

3.1.3 启动 Spark

$cd /app/hadoop/spark-1.1.0/sbin

$./start-all.sh

```
hadoop1 | hadoop2 | hadoop3
[hadoop@hadoop1 ~]$ cd /app/hadoop/spark-1.1.0/sbin
[hadoop@hadoop1 sbin]$ ls
slaves.sh          spark-executor           start-slave.sh         stop-history-server.sh
spark-config.sh    start-all.sh             start-slaves.sh        stop-master.sh
spark-daemon.sh    start-history-server.sh  start-thriftserver.sh  stop-slaves.sh
spark-daemons.sh   start-master.sh          stop-all.sh
[hadoop@hadoop1 sbin]$ ./start-all.sh
starting org.apache.spark.deploy.master.Master, logging to /app/hadoop/spark-1.1.0/sbin/../logs/spark-hadoop-or
g.apache.spark.deploy.master.Master-1-hadoop1.out
hadoop1: starting org.apache.spark.deploy.worker.Worker, logging to /app/hadoop/spark-1.1.0/sbin/../logs/spark-
hadoop-org.apache.spark.deploy.worker.Worker-1-hadoop1.out
hadoop3: starting org.apache.spark.deploy.worker.Worker, logging to /app/hadoop/spark-1.1.0/sbin/../logs/spark-
hadoop-org.apache.spark.deploy.worker.Worker-1-hadoop3.out
hadoop2: starting org.apache.spark.deploy.worker.Worker, logging to /app/hadoop/spark-1.1.0/sbin/../logs/spark-
hadoop-org.apache.spark.deploy.worker.Worker-1-hadoop2.out
[hadoop@hadoop1 sbin]$ jps
2612 SecondaryNameNode
3221 Worker
3263 Jps
2486 NameNode
2547 DataNode
3079 Master
[hadoop@hadoop1 sbin]$
```

3.1.4 启动 Spark-shell

在 spark 客户端（这里在 hadoop1 节点)，使用 spark-shell 连接集群

$cd /app/hadoop/spark-1.1.0/bin

$./spark-shell --master spark://hadoop1:7077 --executor-memory 512m --driver-memory 500m

```
hadoop1 | hadoop2 | hadoop3
[hadoop@hadoop1 ~]$ cd /app/hadoop/spark-1.1.0/bin
[hadoop@hadoop1 bin]$ ./spark-shell --master spark://hadoop1:7077 --executor-memory 512m --driver-memory 500m
Spark assembly has been built with Hive, including Datanucleus jars on classpath
Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties
15/01/16 16:47:39 INFO SecurityManager: Changing view acls to: hadoop,
15/01/16 16:47:39 INFO SecurityManager: Changing modify acls to: hadoop,
15/01/16 16:47:39 INFO SecurityManager: SecurityManager: authentication disabled; ui acls disabled; users with v
iew permissions: Set(hadoop, ); users with modify permissions: Set(hadoop, )
15/01/16 16:47:39 INFO HttpServer: Starting HTTP Server
15/01/16 16:47:39 INFO Utils: Successfully started service 'HTTP class server' on port 34213.
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 1.1.0
      /_/

Using Scala version 2.10.4 (Java HotSpot(TM) 64-Bit Server VM, Java 1.7.0_55)
Type in expressions to have them evaluated.
```

3.1.5 运行 WordCount 脚本

下面就是 WordCount 的执行脚本，该脚本是 scala 编写，以下为一行实现：

scala>sc.textFile("hdfs://hadoop1:9000/user/hadoop/testdata/core-site.xml").flatMap(_.split(" ")).map(x=>(x,1)).reduceByKey(_+_).map(x=>(x._2,x._1)).sortByKey(false).map(x=>(x._2,x._1)).take(10)

为了更好看到实现过程，下面将逐行进行实现：

scala>val rdd=sc.textFile("hdfs://hadoop1:9000/user/hadoop/testdata/core-site.xml")

scala>rdd.cache()

scala>val wordcount=rdd.flatMap(_.split(" ")).map(x=>(x,1)).reduceByKey(_+_)

scala>wordcount.take(10)

scala>val wordsort=wordcount.map(x=>(x._2,x._1)).sortByKey(false).map(x=>(x._2,x._1))

scala>wordsort.take(10)

```
hadoop1 | hadoop2 | hadoop3
scala> val rdd=sc.textFile("hdfs://hadoop1:9000/user/hadoop/testdata/core-site.xml")
15/01/16 16:53:04 WARN BlockManagerMasterActor: Removing BlockManager BlockManagerId(2, hadoop3, 48061, 0) with
no recent heart beats: 95220ms exceeds 45000ms
15/01/16 16:53:04 WARN BlockManagerMasterActor: Removing BlockManager BlockManagerId(1, hadoop2, 56096, 0) with
no recent heart beats: 86928ms exceeds 45000ms
15/01/16 16:53:04 INFO BlockManagerMasterActor: Registering block manager hadoop3:48061 with 267.3 MB RAM
15/01/16 16:53:04 INFO BlockManagerMasterActor: Registering block manager hadoop2:56096 with 267.3 MB RAM
15/01/16 16:53:08 INFO MemoryStore: ensureFreeSpace(138675) called with curMem=0, maxMem=273701928
15/01/16 16:53:08 INFO MemoryStore: Block broadcast_0 stored as values in memory (estimated size 135.4 KB, free
260.9 MB)
15/01/16 16:53:09 INFO MemoryStore: ensureFreeSpace(10090) called with curMem=138675, maxMem=273701928
15/01/16 16:53:09 INFO MemoryStore: Block broadcast_0_piece0 stored as bytes in memory (estimated size 9.9 KB, f
ree 260.9 MB)
15/01/16 16:53:09 INFO BlockManagerInfo: Added broadcast_0_piece0 in memory on hadoop1:39690 (size: 9.9 KB, free
: 261.0 MB)
15/01/16 16:53:09 INFO BlockManagerMaster: Updated info of block broadcast_0_piece0
rdd: org.apache.spark.rdd.RDD[String] = hdfs://hadoop1:9000/user/hadoop/testdata/core-site.xml MappedRDD[1] at t
extFile at <console>:12

scala>

scala> rdd.cache()
res0: rdd.type = hdfs://hadoop1:9000/user/hadoop/testdata/core-site.xml MappedRDD[1] at textFile at <console>:12
```

```
scala> val wordsort=wordcount.map(x=>(x._2,x._1)).sortByKey(false).map(x=>(x._2,x._1))
15/01/16 16:56:01 INFO SparkContext: Starting job: sortByKey at <console>:16
15/01/16 16:56:01 INFO DAGScheduler: Got job 1 (sortByKey at <console>:16) with 2 output partitions (allowLocal=false)
15/01/16 16:56:01 INFO DAGScheduler: Final stage: Stage 2(sortByKey at <console>:16)
15/01/16 16:56:01 INFO DAGScheduler: Parents of final stage: List(Stage 3)
15/01/16 16:56:01 INFO DAGScheduler: Missing parents: List()
15/01/16 16:56:01 INFO DAGScheduler: Submitting Stage 2 (MapPartitionsRDD[7] at sortByKey at <console>:16), which has no missing
 parents
15/01/16 16:56:01 INFO MemoryStore: ensureFreeSpace(3080) called with curMem=157781, maxMem=273701928
15/01/16 16:56:01 INFO MemoryStore: Block broadcast_3 stored as values in memory (estimated size 3.0 KB, free 260.9 MB)
15/01/16 16:56:02 INFO MemoryStore: ensureFreeSpace(1821) called with curMem=160861, maxMem=273701928
15/01/16 16:56:02 INFO MemoryStore: Block broadcast_3_piece0 stored as bytes in memory (estimated size 1821.0 B, free 260.9 MB)
15/01/16 16:56:02 INFO BlockManagerInfo: Added broadcast_3_piece0 in memory on hadoop1:39690 (size: 1821.0 B, free: 261.0 MB)
15/01/16 16:56:02 INFO BlockManagerMaster: Updated info of block broadcast_3_piece0
15/01/16 16:56:02 INFO DAGScheduler: Submitting 2 missing tasks from Stage 2 (MapPartitionsRDD[7] at sortByKey at <console>:16)
15/01/16 16:56:02 INFO TaskSchedulerImpl: Adding task set 2.0 with 2 tasks
15/01/16 16:56:02 INFO TaskSetManager: Starting task 0.0 in stage 2.0 (TID 3, hadoop1, PROCESS_LOCAL, 948 bytes)
15/01/16 16:56:02 INFO TaskSetManager: Starting task 1.0 in stage 2.0 (TID 4, hadoop2, PROCESS_LOCAL, 948 bytes)
15/01/16 16:56:02 INFO BlockManagerInfo: Added broadcast_3_piece0 in memory on hadoop1:60549 (size: 1821.0 B, free: 267.2 MB)
15/01/16 16:56:03 INFO TaskSetManager: Finished task 0.0 in stage 2.0 (TID 3) in 991 ms on hadoop1 (1/2)
15/01/16 16:56:04 INFO BlockManagerInfo: Added broadcast_3_piece0 in memory on hadoop2:56096 (size: 1821.0 B, free: 267.3 MB)
15/01/16 16:56:09 INFO MapOutputTrackerMasterActor: Asked to send map output locations for shuffle 0 to sparkExecutor@hadoop2:54
644
15/01/16 16:56:10 INFO DAGScheduler: Stage 2 (sortByKey at <console>:16) finished in 8.383 s
15/01/16 16:56:10 INFO SparkContext: Job finished: sortByKey at <console>:16, took 9.61845561 s
15/01/16 16:56:10 INFO TaskSetManager: Finished task 1.0 in stage 2.0 (TID 4) in 8383 ms on hadoop2 (2/2)
15/01/16 16:56:10 INFO TaskSchedulerImpl: Removed TaskSet 2.0, whose tasks have all completed, from pool
wordsort: org.apache.spark.rdd.RDD[(String, Int)] = MappedRDD[9] at map at <console>:16
```

```
scala> wordsort.take(10)
15/01/16 16:56:24 INFO SparkContext: Starting job: take at <console>:19
15/01/16 16:56:24 INFO DAGScheduler: Registering RDD 5 (map at <console>:16)
15/01/16 16:56:24 INFO DAGScheduler: Got job 2 (take at <console>:19) with 1 output partitions (allowLocal=true)
15/01/16 16:56:24 INFO DAGScheduler: Final stage: Stage 4(take at <console>:19)
15/01/16 16:56:24 INFO DAGScheduler: Parents of final stage: List(Stage 6)
15/01/16 16:56:24 INFO DAGScheduler: Missing parents: List(Stage 6)
15/01/16 16:56:24 INFO DAGScheduler: Submitting Stage 6 (MappedRDD[5] at map at <console>:16), which has no missing parents
15/01/16 16:56:24 INFO MemoryStore: ensureFreeSpace(2896) called with curMem=162682, maxMem=273701928
15/01/16 16:56:24 INFO MemoryStore: Block broadcast_4 stored as values in memory (estimated size 2.8 KB, free 260.9 MB)
15/01/16 16:56:24 INFO MemoryStore: ensureFreeSpace(1753) called with curMem=165578, maxMem=273701928
15/01/16 16:56:24 INFO MemoryStore: Block broadcast_4_piece0 stored as bytes in memory (estimated size 1753.0 B, free 260.9 MB)
15/01/16 16:56:24 INFO BlockManagerInfo: Added broadcast_4_piece0 in memory on hadoop1:39690 (size: 1753.0 B, free: 261.0 MB)
15/01/16 16:56:24 INFO BlockManagerMaster: Updated info of block broadcast_4_piece0
15/01/16 16:56:24 INFO DAGScheduler: Submitting 2 missing tasks from Stage 6 (MappedRDD[5] at map at <console>:16)
```

```
15/01/16 16:56:25 INFO MapOutputTrackerMaster: Size of output statuses for shuffle 1 is 147 bytes
15/01/16 16:56:25 INFO DAGScheduler: Stage 4 (take at <console>:19) finished in 0.212 s
15/01/16 16:56:25 INFO SparkContext: Job finished: take at <console>:19, took 0.787049106 s
res2: Array[(String, Int)] = Array(("",100), (the,7), (</property>,6), (<property>,6), (under,3), (in,3), (License,3), (this,2),
 (-->,2), (file.,2))

scala> 15/01/16 16:56:25 INFO TaskSetManager: Finished task 0.0 in stage 4.0 (TID 7) in 233 ms on hadoop1 (1/1)
15/01/16 16:56:25 INFO TaskSchedulerImpl: Removed TaskSet 4.0, whose tasks have all completed, from pool
```

词频统计结果如下：

Array[(String, Int)] = Array(("",100), (the,7), (</property>,6), (<property>,6), (under,3), (in,3), (License,3), (this,2), (-->,2), (file.,2))

3.1.6 观察运行情况

通过 http://hadoop1:8080 查看 Spark 运行情况，可以看到 Spark 为 3 个节点，每个节点各为 1 个内核/512M 内存，客户端分配 3 个核，每个核有 512M 内存。

Workers

Id	Address	State	Cores	Memory
worker-20150116164509-hadoop1-36242	hadoop1:36242	ALIVE	1 (1 Used)	512.0 MB (512.0 MB Used)
worker-20150116164524-hadoop2-57106	hadoop2:57106	ALIVE	1 (1 Used)	512.0 MB (512.0 MB Used)
worker-20150116164524-hadoop3-59500	hadoop3:59500	ALIVE	1 (1 Used)	512.0 MB (512.0 MB Used)

Running Applications

ID	Name	Cores	Memory per Node	Submitted Time	User	State	Duration
app-20150116164822-0000	Spark shell	3	512.0 MB	2015/01/16 16:48:22	hadoop	RUNNING	16 min

通过点击客户端运行任务 ID，可以看到该任务在 hadoop2 和 hadoop3 节点上运行，在 hadoop1 上并没有运行，主要是由于 hadoop1 为 NameNode 和 Spark 客户端造成内存占用过大造成

Executor Summary

ExecutorID	Worker	Cores	Memory	State	Logs
2	worker-20150116164524-hadoop3-59500	1	512	RUNNING	stdout stderr
1	worker-20150116164524-hadoop2-57106	1	512	RUNNING	stdout stderr
0	worker-20150116164509-hadoop1-36242	1	512	RUNNING	stdout stderr

3.2 使用 Spark-submit 测试

从 Spark1.0.0 开始，Spark 提供了一个易用的应用程序部署工具 bin/spark-submit，可以完成 Spark 应用程序在 local、Standalone、YARN、Mesos 上的快捷部署。该工具语法及参数说明如下：

```
Usage: spark-submit [options] <app jar | python file> [app options]
Options:
  --master MASTER_URL          spark://host:port, mesos://host:port, yarn, or local.
  --deploy-mode DEPLOY_MODE    driver 运行之处，client 运行在本机，cluster 运行在集群
  --class CLASS_NAME           应用程序包的要运行的 class
  --name NAME                  应用程序名称
  --jars JARS                  用逗号隔开的 driver 本地 jar 包列表以及 executor 类路径
  --py-files PY_FILES          用逗号隔开的放置在 Python 应用程序
                               PYTHONPATH 上的.zip, .egg, .py 文件列表
  --files FILES                用逗号隔开的要放置在每个 executor 工作目录的文件列表
  --properties-file FILE       设置应用程序属性的文件放置位置，默认是 conf/spark-defaults.conf
  --driver-memory MEM          driver 内存大小，默认 512M
  --driver-java-options        driver 的 java 选项
  --driver-library-path        driver 的库路径 Extra library path entries to pass to the driver
  --driver-class-path          driver 的类路径，用--jars 添加的 jar 包会自动包含在类路径里
  --executor-memory MEM        executor 内存大小，默认 1G
```

Spark standalone with cluster deploy mode only:

--driver-cores NUM　　driver 使用内核数，默认为 1

--supervise　　如果设置了该参数，driver 失败是会重启

Spark standalone and Mesos only:

--total-executor-cores NUM　　executor 使用的总核数

YARN-only:

--executor-cores NUM　　每个 executor 使用的内核数，默认为 1

--queue QUEUE_NAME　　提交应用程序给哪个 YARN 的队列，默认是 default 队列

--num-executors NUM　　启动的 executor 数量，默认是 2 个

--archives ARCHIVES　　被每个 executor 提取到工作目录的档案列表，用逗号隔开

3.2.1 运行脚本 1

该脚本为 Spark 自带例子，在该例子中个计算了圆周率π的值，以下为执行脚本：

$cd /app/hadoop/spark-1.1.0/bin
$./spark-submit --master spark://hadoop1:7077 --class org.apache.spark.examples.SparkPi --executor-memory 512m ../lib/spark-examples-1.1.0-hadoop2.2.0.jar 200

参数说明（详细可以参考上面的参数说明）：

- **--master** Master 所在地址，可以有 Mesos、Spark、YARN 和 Local 四种，在这里为 Spark Standalone 集群，地址为 spark://hadoop1:7077
- **--class** 应用程序调用的类名，这里为 org.apache.spark.examples.SparkPi
- **--executor-memory** 每个 executor 所分配的内存大小，这里为 512M
- **执行 jar 包** 这里是../lib/spark-examples-1.1.0-hadoop2.2.0.jar
- **分片数目** 这里数目为 200

```
hadoop1  hadoop2  hadoop3
[hadoop@hadoop1 ~]$ cd /app/hadoop/spark-1.1.0/bin
[hadoop@hadoop1 bin]$ ./spark-submit --master spark://hadoop1:7077 --class org.apache.spark.examples.SparkPi --executor-memory 5
12m ../lib/spark-examples-1.1.0-hadoop2.2.0.jar 200
Spark assembly has been built with Hive, including Datanucleus jars on classpath
Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties
15/01/16 17:18:36 INFO SecurityManager: Changing view acls to: hadoop,
15/01/16 17:18:36 INFO SecurityManager: Changing modify acls to: hadoop,
15/01/16 17:18:36 INFO SecurityManager: SecurityManager: authentication disabled; ui acls disabled; users with view permissions:
 Set(hadoop, ); users with modify permissions: Set(hadoop, )
15/01/16 17:18:38 INFO Slf4jLogger: Slf4jLogger started
15/01/16 17:18:38 INFO Remoting: Starting remoting
15/01/16 17:18:39 INFO Remoting: Remoting started; listening on addresses :[akka.tcp://sparkDriver@hadoop1:48507]
15/01/16 17:18:39 INFO Remoting: Remoting now listens on addresses: [akka.tcp://sparkDriver@hadoop1:48507]
15/01/16 17:18:39 INFO Utils: Successfully started service 'sparkDriver' on port 48507.
15/01/16 17:18:39 INFO SparkEnv: Registering MapOutputTracker
```

```
15/01/16 17:19:41 INFO TaskSetManager: Starting task 192.0 in stage 0.0 (TID 192, hadoop2, PROCESS_LOCAL, 1230 bytes)
15/01/16 17:19:41 INFO TaskSetManager: Finished task 191.0 in stage 0.0 (TID 191) in 64 ms on hadoop2 (192/200)
15/01/16 17:19:41 INFO TaskSetManager: Starting task 193.0 in stage 0.0 (TID 193, hadoop2, PROCESS_LOCAL, 1230 bytes)
15/01/16 17:19:41 INFO TaskSetManager: Finished task 192.0 in stage 0.0 (TID 192) in 73 ms on hadoop2 (193/200)
15/01/16 17:19:41 INFO TaskSetManager: Starting task 194.0 in stage 0.0 (TID 194, hadoop2, PROCESS_LOCAL, 1230 bytes)
15/01/16 17:19:41 INFO TaskSetManager: Finished task 193.0 in stage 0.0 (TID 193) in 128 ms on hadoop2 (194/200)
15/01/16 17:19:41 INFO TaskSetManager: Starting task 195.0 in stage 0.0 (TID 195, hadoop2, PROCESS_LOCAL, 1230 bytes)
15/01/16 17:19:41 INFO TaskSetManager: Finished task 194.0 in stage 0.0 (TID 194) in 38 ms on hadoop2 (195/200)
15/01/16 17:19:41 INFO TaskSetManager: Starting task 196.0 in stage 0.0 (TID 196, hadoop2, PROCESS_LOCAL, 1230 bytes)
15/01/16 17:19:41 INFO TaskSetManager: Finished task 195.0 in stage 0.0 (TID 195) in 34 ms on hadoop2 (196/200)
15/01/16 17:19:41 INFO TaskSetManager: Starting task 197.0 in stage 0.0 (TID 197, hadoop2, PROCESS_LOCAL, 1230 bytes)
15/01/16 17:19:41 INFO TaskSetManager: Finished task 196.0 in stage 0.0 (TID 196) in 63 ms on hadoop2 (197/200)
15/01/16 17:19:42 INFO TaskSetManager: Starting task 198.0 in stage 0.0 (TID 198, hadoop2, PROCESS_LOCAL, 1230 bytes)
15/01/16 17:19:42 INFO TaskSetManager: Finished task 197.0 in stage 0.0 (TID 197) in 78 ms on hadoop2 (198/200)
15/01/16 17:19:42 INFO TaskSetManager: Starting task 199.0 in stage 0.0 (TID 199, hadoop2, PROCESS_LOCAL, 1230 bytes)
15/01/16 17:19:42 INFO TaskSetManager: Finished task 198.0 in stage 0.0 (TID 198) in 138 ms on hadoop2 (199/200)
15/01/16 17:19:42 INFO TaskSetManager: Finished task 199.0 in stage 0.0 (TID 199) in 184 ms on hadoop2 (200/200)
15/01/16 17:19:42 INFO TaskSchedulerImpl: Removed TaskSet 0.0, whose tasks have all completed, from pool
15/01/16 17:19:42 INFO DAGScheduler: Stage 0 (reduce at SparkPi.scala:35) finished in 41.104 s
15/01/16 17:19:42 INFO SparkContext: Job finished: reduce at SparkPi.scala:35, took 48.425582842 s
```

3.2.2 观察运行情况

通过观察 Spark 集群有 3 个 Worker 节点和正在运行的 1 个应用程序，每个 Worker 节点为 1 内核/512M 内存。由于没有指定应用程序所占内核数目，则该应用程序占用该集群所有 3 个内核，并且每个节点分配 512M 内存。

Workers

Id	Address	State	Cores	Memory
worker-20150116164509-hadoop1-36242	hadoop1:36242	ALIVE	1 (1 Used)	512.0 MB (512.0 MB Used)
worker-20150116164524-hadoop2-57106	hadoop2:57106	ALIVE	1 (1 Used)	512.0 MB (512.0 MB Used)
worker-20150116164524-hadoop3-59500	hadoop3:59500	ALIVE	1 (1 Used)	512.0 MB (512.0 MB Used)

Running Applications

ID	Name	Cores	Memory per Node	Submitted Time	User	State	Duration
app-20150116171855-0001	Spark Pi	3	512.0 MB	2015/01/16 17:18:55	hadoop	RUNNING	8 s

根据每个节点负载情况，每个节点运行 executor 并不相同，其中 hadoop1 的 executor 数目为 0。而 hadoop3 执行 executor 数为 10 个，其中 5 个 EXITED 状态，5 个 KILLED 状态。

Executor Summary

ExecutorID	Worker	Cores	Memory	State	Logs
2	worker-20150116164524-hadoop3-59500	1	512	EXITED	stdout stderr
1	worker-20150116164524-hadoop2-57106	1	512	EXITED	stdout stderr
3	worker-20150116164524-hadoop3-59500	1	512	EXITED	stdout stderr

Removed Executors

ExecutorID	Worker	Cores	Memory	State	Logs
4	worker-20150116164524-hadoop2-57106	1	512	KILLED	stdout stderr
5	worker-20150116164524-hadoop3-59500	1	512	KILLED	stdout stderr
0	worker-20150116164509-hadoop1-36242	1	512	KILLED	stdout stderr

3.2.3 运行脚本 2

该脚本为 Spark 自带例子，在该例子中个计算了圆周率π的值，区别脚本 1 这里指定了每个 executor 内核数据，以下为执行脚本：

$cd /app/hadoop/spark-1.1.0/bin

$./spark-submit --master spark://hadoop1:7077 --class org.apache.spark.examples.SparkPi --executor-memory 512m --total-executor-cores 2 ../lib/spark-examples-1.1.0-hadoop2.2.0.jar 200

参数说明（详细可以参考上面的参数说明）：

- **--master** Master 所在地址，可以有 Mesos、Spark、YARN 和 Local 四种，在这里为 Spark Standalone 集群，地址为 spark://hadoop1:7077
- **--class** 应用程序调用的类名，这里为 org.apache.spark.examples.SparkPi
- **--executor-memory** 每个 executor 所分配的内存大小，这里为 512M
- **--total-executor-cores 2** 每个 executor 分配的内核数
- **执行 jar 包** 这里是../lib/spark-examples-1.1.0-hadoop2.2.0.jar
- **分片数目** 这里数目为 200

3.2.4 观察运行情况

通过观察 Spark 集群有 3 个 Worker 节点和正在运行的 1 个应用程序，每个 Worker 节点为 1 内核/512M 内存。由于指定应用程序所占内核数目为 2，则该应用程序使用该集群所有 2 个内核。

Workers

Id	Address	State	Cores	Memory
worker-20150116164509-hadoop1-36242	hadoop1:36242	ALIVE	1 (1 Used)	512.0 MB (512.0 MB Used)
worker-20150116164524-hadoop2-57106	hadoop2:57106	ALIVE	1 (1 Used)	512.0 MB (512.0 MB Used)
worker-20150116164524-hadoop3-59500	hadoop3:59500	ALIVE	1 (0 Used)	512.0 MB (0.0 B Used)

Running Applications

ID	Name	Cores	Memory per Node	Submitted Time	User	State	Duration
app-20150116172746-0003	Spark Pi	2	512.0 MB	2015/01/16 17:27:46	hadoop	RUNNING	9 s

Executor Summary

ExecutorID	Worker	Cores	Memory	State	Logs
1	worker-20150116164524-hadoop2-57106	1	512	RUNNING	stdout stderr
0	worker-20150116164509-hadoop1-36242	1	512	RUNNING	stdout stderr