目录

1. 实验一：Scala 的安装

1.1. 实验目的

完成本实验，您应该能够：

- 掌握 Scala 的安装部署
- 掌握启动与关闭 Scala Shell 的方法

1.2. 实验要求

- 熟悉常用 Linux 操作系统命令
- 熟练 vi 编辑器的基本操作命令
- 熟悉 Linux 环境变量的配置

1.3. 实验环境

本实验所需之主要资源环境如表 1-1 所示。

表 1-1 资源环境

服务器集群	3 个以上节点，节点间网络互通，各节点最低配置：双核 CPU、8GB 内存、100G 硬盘
运行环境	CentOS 7.4，Java JDK 1.8
用户名/密码	root/password hadoop/password

服务和组件	Scala 2.11.8

1.4. 实验视图

1.5. 实验过程

1.5.1. 步骤一：解压 Scala 压缩文件并重命名

本实验所使用的版本是Scala 2.11.8，可以官网下载：https://downloads.lightbend.com/scala/2.11.8/scala-2.11.8.tgz。所有实验下载好的安装包都需放到/opt/software目录下，所以，Scala-2.11.8.tgz也放到/opt/software目录下，解压scala到/usr/local/src文件夹，然后将解压的scala-2.11.8目录重命名为scala，如下所示：

[root@master ~]# tar -zxvf /opt/software/scala-2.11.8.tgz -C /usr/local/src/

其中，tar –zxvf是解压命令；

[root@master ~]# mv /usr/local/src/scala-2.11.8/ /usr/local/src/scala

其中，mv是移动文件或者目录的命令。

分发scala到子节点

[root@master ~]# scp -r /usr/local/src/scala/ root@slave1:/usr/local/src/

[root@master ~]# scp -r /usr/local/src/scala/ root@slave2:/usr/local/src/

1.5.2. 步骤二： 修改 scala 目录的用户权限

如果实验的一开始就已经对目录“/urs/local/src”赋予了 hadoop 用户权限，该步骤可略过，否则，就需要对重命名过的 scala 目录进行用户权限修改，便于后续 hadoop 用户对该目录进行相关的操作，其命令为：

[root@master ~]# chown -R hadoop:hadoop /usr/local/src/scala

[root@master ~]# chown -R hadoop:hadoop /usr/local/src/scala

[root@master ~]# chown -R hadoop:hadoop /usr/local/src/scala

即可完成用户权限的修改，其中，chown 是修改用户权限的命令。

1.5.3. 步骤三： 配置环境变量

由于每次启动 scala shell 都需要进入到“/usr/local/src/scala/bin”目录下，否则，会提示无法识别 scala 命令。因此在“/etc/profile”文件中配置 scala 的环境变量，就可以在任意位置启动 scala shell 进入交互式编程。先打开“/etc/profile”如下所示：

[root@master ~]# vi /etc/profile

[root@slave1 ~]# vi /etc/profile

[root@slave2 ~]# vi /etc/profile

然后在键盘输入字母“i”或者“o”进入编辑模式，在文件中加入图 1-4 所示内容：

export SCALA_HOME=/usr/local/src/scala

export PATH=$PATH:$SCALA_HOME/bin

然后在键盘按“Esc”键退出编辑模式，并键盘输入“:wq”进行内容保存并退出，最后将该“.bashrc”文件生效即可完成 scala 的环境配置。

[root@master ~]# su hadoop

[hadoop@master root]$ source /etc/profile

[root@master ~]# su hadoop

[hadoop@master root]$ source /etc/profile

[root@master ~]# su hadoop

[hadoop@master root]$ source /etc/profile

1.5.4. 步骤四： 验证 Scala 安装是否成功

如果配置了步骤三中的环境变量，可以直接在任意路径输入命令“scala -version”即可，但是，整个实验中我们都是使用的绝对路径，所以，完成 scala 验证的命令为：

[hadoop@master ~]$ cd /usr/local/scr/scala/bin

即可进入 bin 目录

输入 scala 进入 scala shell 交互编程界面：

[hadoop@master bin]$ scala

scala>

然后退出 scala shell 的命令为：

scala>:quit

就可以退出 scala shell 交互编程界面。

2. 实验二 安装 Spark

2.1. 实验目的

完成本实验，您应该能够：

- 独立安装 Spark 集群
- 独立修改 Spark 参数
- 独立启动 Spark 集群
- 独立启动和关闭 Spark-shell

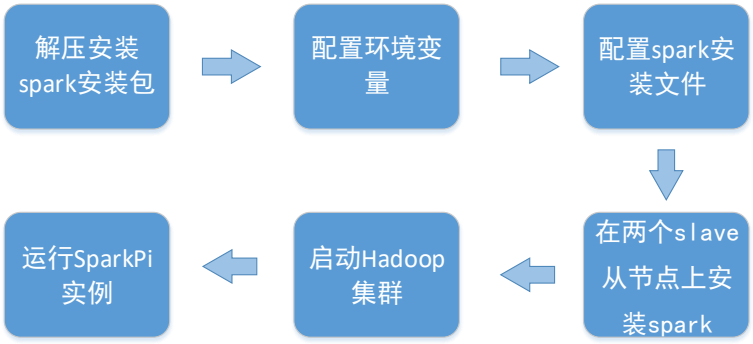
2.2. 实验要求

- 熟悉 zookeeper 的启动与关闭
- 熟悉 Hadoop 分布式集群的启动与关闭

2.3. 实验环境

服务器集群	3 个以上节点，节点间网络互通，各节点最低配置：双核 CPU、8GB 内存、100G 硬盘
运行环境	CentOS 7.4、Java JDK 1.8
用户名/密码	root/password hadoop/password
服务和组件	Hadoop 2.7.1、Zookeeper 3.4.8、Scala 2.11.8

2.4. 实验视图



2.5. 实验过程

本实验是在已有的 Hadoop 分布式集群上搭建 Spark 集群，实验中有 3 个节点，其中，1 个是 master 主节点，另外 2 个是 slave 从节点。

2.5.1. 任务一：在 master 节点上安装 spark

2.5.1.1 步骤一：在 master 主节点上解压 Spark 安装包

本实验所使用的版本是 Spark2.0.0，官网下载地址：http://spark.apache.org/downloads.html。该实验中的 Spark 包下载到了放到/opt/software 目录下，将 Spark 包解压到/usr/local/src 下的命令（该命令可以在任意路径执行）为：

```
[hadoop@master bin]$ su root
[root@master bin]# cd
[root@master ~]# tar -zxvf /opt/software/spark-2.0.0-bin-hadoop2.6.tgz -C /usr/local/src/
```

2.5.1.2 步骤二：Spark 解压后的重命名操作

解压到“/usr/local/src”下的 Spark 目录名为“spark-2.0.0-bin-hadoop2.7”，重命名为 spark 的命令为：

```
[root@master ~]# mv    /usr/local/src/spark-2.0.0-bin-hadoop2.6/    /usr/local/src/spark
```

2.5.1.3 步骤三：修改 spark 目录的用户权限

同样，如果实验的一开始就已经对目录“/urs/local/src”赋予了 hadoop 用户权限，该步骤可略过，否则，需对重命名过的 spark 目录进行用户权限修改，便于后续 hadoop 用户对该目录进行相关的操作，在终端执行命令：

```
[root@master ~]# chown    -R    hadoop:hadoop    /usr/local/src/spark
```

2.5.1.4 步骤四：配置环境变量

由于每次启动 spark shell 都需要进入到“/usr/local/src/spark/bin”目录下，否则，会提示无法识别 spark-shell 命令。因此在“~/.bashrc”文件中配置 Spark 的环境变量，这样就可以在任意位置启动 spark shell 进行交互式编程。先打开“~/.bashrc”文件：

```
[root@master ~]# vi /etc/profile
```

然后在键盘输入字母“i”或者“o”进入编辑模式，在文件中加入如下内容：

```
export SPARK_HOME=/usr/local/src/spark
export PATH=$PATH:$SPARK_HOME/bin:$PATH
```

如果已经存在“export PATH”这样的字眼在该文件中，需要在其末尾加一个冒号作为与之前存在内容间的分隔符，然后再加入上面指定的内容，再在键盘按“Esc”键退出编辑模式，并在键盘输入“**:wq**”进行内容保存并退出，最后将修改过的“.bashrc”文件生效，命令为：

```
[root@master ~]# su hadoop
[hadoop@master root]$ source /etc/profile
```

2.5.2 任务二：修改 Spark 参数

2.5.2.1 步骤一：修改 spark-env.sh

由于要建立 Spark 与 Hadoop 之间的连接，需修改 Spark 参数，先进入 Spark 的 conf 目录把“spark-env.sh.template”拷贝为“spark-env.sh”文件并修改配置，

进入 Spark 的配置文件目录“conf”：

[hadoop@master root]$ cd /usr/local/src/spark/conf/

将已有的文件“spark-env.sh.template”复制出来并命名为 spark-evn.sh：

[hadoop@master conf]$ cp /usr/local/src/spark/conf/spark-env.sh.template /usr/local/src/spark/conf/spark-env.sh

进入 spark 配置文件“spark-env.sh”，命令为：

[hadoop@master conf]$ vi /usr/local/src/spark/conf/spark-env.sh

然后将下面所示内容加入到文件“spark-env.sh”中：

```
export JAVA_HOME=/usr/local/src/java

export HADOOP_HOME=/usr/local/src/hadoop

export SCALA_HOME=/usr/local/src/scala

export SPARK_MASTER_IP=master

export SPARK_MASTER_PORT=7077

export SPARK_DIST_CLASSPATH=$(/usr/local/src/hadoop/bin/hadoop classpath)

export HADOOP_CONF_DIR=/usr/local/src/hadoop/etc/hadoop

export SPARK_YARN_USER_ENV="CLASSPATH=/usr/local/src/hadoop/etc/hadoop"

export YARN_CONF_DIR=/usr/local/src/hadoop/etc/hadoop
```

其中，三个参数的意义分别为： SPARK_DIST_CLASSPATH 是完成 spark 和 hadoop 的挂接，HADOOP_CONF_DIR 是说明了 hadoop 相关配置信息的目录，SPARK_MASTER_IP 是指明该集群中主节点的 IP 地址或者名称。

2.5.2.2 步骤二：配置 slaves 文件

在 master 节点上安装好后，需建立 master 节点与 slave1 和 slave2 节点的链接关系，所以需将 spark 中的 conf 目录下的 slaves.template 文件重命名为 slaves，执行命令为：

[hadoop@master conf]$ cp /usr/local/src/spark/conf/slaves.template /usr/local/src/spark/conf/slaves

然后通过 vi 编辑器进入 slaves 文件，并将文件中的内容修改为以下内容：

[hadoop@master conf]$ vi slaves

```
master

slave1

slave2
```

其中，master、slave1、slave2 分别为主节点名和两从节点名

接着在键盘按“Esc”键退出编辑模式，并在键盘输入“:wq”进行保存并退出。

2.5.3 任务三：在两个 slaves 从节点上安装 Spark

2.5.3.1 步骤一：将 master 主节点上的 Spark 安装目录和.bashrc 环境变量复制到两个 slaves 从节点上

由于在各个节点上安装 Spark 的过程都一样，所以，不用重复安装，只需在 master 主节点上将已经安装好的 Spark 目录和.bashrc 文件复制到两个从节点上。如下所示：

```
[hadoop@master spark]$ su root
[root@master spark]# scp -r /usr/local/src/spark/ root@slave1:/usr/local/src/
[root@master spark]# scp -r /usr/local/src/spark/ root@slave2:/usr/local/src/
[root@master spark]# scp /etc/profile root@slave1:/etc/
[root@master spark]# scp /etc/profile root@slave2:/etc/
```

其中，scp 是节点之间复制文件的命令，hadoop@slave1 是指在 slave1 节点上的 hadoop 用户。

2.5.3.2 步骤二：在 slave1、slave2 节点上分别安装 Spark

由于 Spark 的安装目录已经复制到 slave1、slave2 节点上了。将两个 slaves 从节点上目录“/usr/local/spark”的用户权限修改为 hadoop，分别在 slave1 和 slave2 节点上执行命令为：

```
[root@slave1 spark]# chown -R hadoop:hadoop /usr/local/src/spark/
[root@slave1 spark]# su hadoop
[hadoop@slave1 spark]$ source /etc/profile
[root@slave2 spark]# chown -R hadoop:hadoop /usr/local/src/spark/
[root@slave2 spark]# su hadoop
[hadoop@slave2 spark]$ source /etc/profile
```

2.5.4 任务四：运行示例

2.5.4.1 步骤一：启动 Hadoop 集群

在三个节点启动 zookeeper，命令如下所示：

```
[root@master spark]$ su hadoop
[hadoop@master spark]$ cd /usr/local/src/zookeeper/bin/
[hadoop@master bin]$ ./zkServer.sh start
[hadoop@slave1 spark]$ cd /usr/local/src/zookeeper/bin/
[hadoop@slave1 bin]$ ./zkServer.sh start
[hadoop@slave2 spark]$ cd /usr/local/src/zookeeper/bin/
[hadoop@slave2 bin]$ ./zkServer.sh start
```

在 master 节点启动 hadoop 集群，命令如下所示：

```
[hadoop@master spark]$ cd /usr/local/src/hadoop/sbin/
[hadoop@master sbin]$ ./start-all.sh
```

2.5.4.2 步骤二：以集群模式运行 SparkPi 实例程序

在 master 节点上启动 SparkPi 实例程序，如下所示：

```
[hadoop@master spark]$ cd /usr/local/src/spark/
```

[hadoop@master spark]$ ./bin/spark-submit --class org.apache.spark.examples.SparkPi --master yarn --deploy-mode client --driver-memory 512M --executor-memory 512M --executor-cores 1 examples/jars/spark-examples_2.11-2.0.0.jar 40

在运行结果中间可以找到我们需要的 pi 值，如下所示：

20/07/04 05:48:48 INFO scheduler.DAGScheduler: Job 0 finished: reduce at SparkPi.scala:38, took 3.866892 s

Pi is roughly 3.141053785263446

因为我们使用 yarn 集群来管理资源，所以在 master 节点上打开浏览器，访问 http://master:8088 显示 yarn 的信息，就可以看到我们运行的 SparkPi 实例程序，如图 2-1 所示：

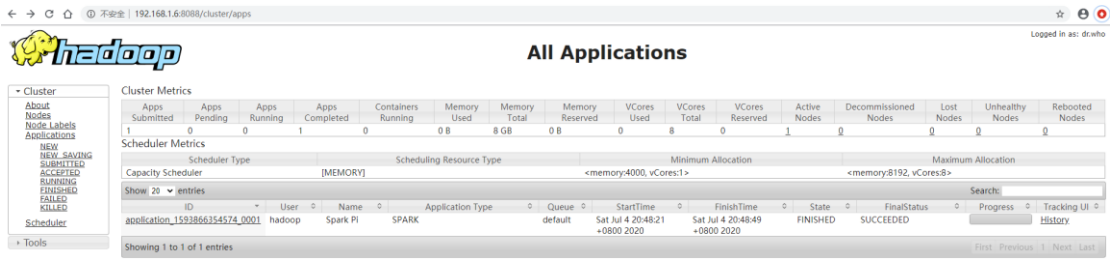


图 2-1 master 节点上浏览器查看 yarn

3. 实验三 Spark shell 编程

3.1. 实验目的

完成本实验，您应该能够：

- 独立启动与关闭 spark-shell
- 熟练调用 spark 的 actions 与 transformations 中常用方法

3.2. 实验要求

- 熟悉 Hadoop 集群和 spark 集群的启动和关闭
- 具有 scala 语言基础

3.3. 实验环境

服务器集群	3 个以上节点，节点间网络互通，各节点最低配置：双核 CPU、8GB 内存、100G 硬盘
运行环境	CentOS 7.4、Java 1.8
用户名/密码	root/password hadoop/password
服务和组件	Hadoop 2.7.1 、Scala 2.11.8、Spark 2.0.0

3.4. 实验视图

3.5. 实验过程

3.5.1. 任务一：在 Yarn 集群管理器上运行 spark-shell

首先启动集群：

三个节点启动 zookeeper，命令如下所示：

[hadoop@master spark]$ cd /usr/local/src/zookeeper/bin/

[hadoop@master bin]$ ./zkServer.sh start

[hadoop@slave1 spark]$ cd /usr/local/src/zookeeper/bin/

[hadoop@slave1 bin]$ ./zkServer.sh start

[hadoop@slave2 spark]$ cd /usr/local/src/zookeeper/bin/

[hadoop@slave2 bin]$ ./zkServer.sh start

在 master 节点启动 hadoop 集群，命令如下所示：

[hadoop@master spark]$ cd /usr/local/src/hadoop/sbin/

[hadoop@master sbin]$ ./start-all.sh

再在 Yarn 集群管理器上启动 spark-shell，命令为：

[hadoop@master spark]$ cd /usr/local/src/spark/bin/

[hadoop@master bin]$ ./spark-shell --master yarn --deploy-mode client

3.5.2. 任务二：在 spark-shell 上运行一个 WordCount 案例

3.5.2.1 步骤一：通过加载文件新建一个 RDD

该 WordCount 案例我们统计的文件就是本地磁盘上 file:///usr/local/src/spark/README.md 文件，其中 file://前缀指定本地文件，spark shell 默认是读取 HDFS 中的文件，需要先上传该文件到 HDFS 中，否则会有报错：

[hadoop@master ~]$ cd /usr/local/src/spark

[hadoop@master spark]$ hadoop fs -put README.md /

通过加载 README.md 文件新建一个 RDD：

scala> val textFile=sc.textFile("/README.md")

3.5.2.2 步骤二：对 RDD 进行 actions 和 transformations 操作

下面我们就来演示 actions 动作操作中的 first()和 count()两个操作，如下所示

scala> textFile.first()　　#查看 textFile 中的第一条数据

scala> textFile.count()　#统计 textFile 中的单词总数

接着演示 transformations 转换操作，运行代码如下所示：

scala> val

wordcount=textFile.flatMap(line=>line.split(",")).map(word=>(word,1)).reduceByKey(_+_)

其中，reduceByKey(_+_)是 reduceByKey((x,y)=>x+y)的简化写法，同样是寻找相同 key 的数据，当找到这样的两条记录时会对其 value 求和，只是不指定将两个 value 存入 x 和 y 中，同样只保留求和之后的数据作为 value。反复执行这个操作直至每个 key 只留下一条记录。以上四种方式等价。然后通过 collect 操作将远程数据通过网络传输到本地进行词频统计：

```
scala> wordcount.collect()
```

collect()方法得到的结果是一个 list，然后可以通过 foreach()方法遍历 list 中的每一个元组数据并返回其结果，如下所示：

```
scala> wordcount.collect().foreach(println)
```

注意：在 spark shell 交互式编程环境下，如果代码一行放不下，可以在圆点后回车，在下一行继续输入

结束之后退出 spark-shell

```
scala>:q
```