


Hadoop2.8和Spark2.1完全分布式搭建详解

1. 一、前期准备工作：

1.1.1 1.安装包的准备：

- VMware(10.0 版本以上)：
 - 官方网站：<https://www.vmware.com/cn.html>
 - 官方下载地址：<http://www.vmware.com/products/player/playerpro-evaluation.html>
 - 10.0 版本注册码：
 - v1Z0G9-67285-FZG78-ZL3Q2-234JG
 - 4C4EK-89KDL-5ZFP9-1LA5P-2A0J0
 - HY086-4T01N-CZ3U0-CV0QM-13DNU
 - 11.0 版本注册码：
 - 1F04Z-6D111-7Z029-AV0Q4-3AEH8
 - 12.0 版本注册码：
 - 5A02H-AU243-TZJ49-GTC7K-3C61N
- ubuntu14.0 系统:(64 位)选择 ubuntu 纯属个人喜好, Linux 发行版有很多都支持 Hadoop, 而 14.0 版本是个比较稳定的版本, 不算太新所以很多东西支持的比较好, 重要的是支持 CDH 的 hadoop 生态系统构建。
 - 官方地址：<https://www.ubuntu.com/download/alternative-downloads> 选择 14.0 版本即可。
- jdk1.8 安装包：
 - 官方下载：<http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html> 选择 linux_x64。
- scala2.11 安装包：
 - 官方下载：<http://www.scala-lang.org/> 点击 Download 即可下载。
- spark2.1 安装包：
 - 官方下载：<http://spark.apache.org/downloads.html> 这里 spark 提供了和 hadoop 绑定的版本, 但是由于没有提供 2.8 的 hadoop 绑定版, 所以这里选择通用版 [spark-2.1.0-bin-without-hadoop](#) 来进行下载。如下图：第一个是安装包。

	spark-2.1.0-bin-without-hadoop.tgz	2016-12-22 21:58	117M
	spark-2.1.0-bin-without-hadoop.tgz.asc	2016-12-22 21:58	490
	spark-2.1.0-bin-without-hadoop.tgz.md5	2016-12-22 21:58	121
	spark-2.1.0-bin-without-hadoop.tgz.sha	2016-12-22 21:58	288

1.1.2 2.辅助工具安装包：

- Putty：一个十分简洁的连接服务器的工具。因为虚拟机太卡了长期在上面操作的话会卡到爆。用 Putty 可以在主机用一个终端来操作虚拟机。<http://www.linuxidc.com/Linux/2016-08/133991.htm>
- FlashFxp：用于在宿主机上传和下载虚拟机的文件, 当然 VMware 安装了 tools 之后可以随意拖拽很方便, 但是还是考虑到卡爆的问题, 虚拟机启动后我们完全不管他, 就当服务器来用。
 - 下载地址：<http://www.linuxidc.net/thread-1188-1-1.html>

1.1.3 3.系统基本配置：（未说明则均在主机 rzxmaster 上操作）

第一步：安装 VMware，创建虚拟机 Master，安装 Vim，Mysql（mysql 也可以暂且不装，但是考虑到后面组件的扩展还是先安上）。

第二步：克隆虚拟机(选择完全克隆)rxmater，分别命名为 rxslave1，rxslave2（这里的命名可以自行修改）。然后启动三个虚拟机。

- - 分别修改 hostname 主机名。(这里实例修改 rzxmaster 这台机器的主机名)。
 - 输入：sudo vim /etc/hostname 回车进入 hostname 文件的编辑。
 - 在 hostname 文件输入：rxmaster 另外两台机器同理修改保存退出即可。
 - 输入：source /etc/hostname 使配置立即生效。关闭终端重新打开即可看到主机名已经改变了。

第四步：静态 Ip 设置：

静态 IP 设置：这篇 <http://www.linuxidc.com/Linux/2017-04/143102.htm> 介绍的很详细，但是其中有部分问题，不知道是 16.0 和 14.0 版本差异的问题还是教程本身的问题，一个是网络重启之后 DNS 配置丢失的问题。每次重启之后会发现配置的 DNS 文件恢复成了 127.0.0.1

下面关于 **Hadoop** 的文章您也可能喜欢，不妨看看：

Ubuntu14.04 下 Hadoop2.4.1 单机/伪分布式安装配置教程 <http://www.linuxidc.com/Linux/2015-02/113487.htm>

CentOS 6.3 下 Hadoop 伪分布式平台搭建 <http://www.linuxidc.com/Linux/2016-11/136789.htm>

Ubuntu 14.04 LTS 下安装 Hadoop 1.2.1（伪分布模式） <http://www.linuxidc.com/Linux/2016-09/135406.htm>

Ubuntu 上搭建 Hadoop 环境（单机模式+伪分布模式） <http://www.linuxidc.com/Linux/2013-01/77681.htm>

实战 CentOS 系统部署 Hadoop 集群服务 <http://www.linuxidc.com/Linux/2016-11/137246.htm>

Hadoop 2.6.0 HA 高可用集群配置详解 <http://www.linuxidc.com/Linux/2016-08/134180.htm>

Spark 1.5、Hadoop 2.7 集群环境搭建 <http://www.linuxidc.com/Linux/2016-09/135067.htm>

在 Ubuntu X64 上编译安装 Hadoop <http://www.linuxidc.com/Linux/2016-12/138568.htm>

CentOS 6.7 安装 Hadoop 2.7.3 <http://www.linuxidc.com/Linux/2017-01/139089.htm>

CentOS7+Hadoop2.5.2+Spark1.5.2 环境搭建 <http://www.linuxidc.com/Linux/2017-01/139364.htm>

这个问题是由于 interface，networkManager 两种网络管理冲突造成的。解决方法就是在编辑链接的时候将 DNS 也一起编辑。这样就不用再编辑 DNS 的配置文件。如下图所示：



其他步骤按博文所说就可以完成静态 IP 的配置。

第五步：hosts 配置，**特别强调主机名称不要含有下划线"_"**，最好是纯英文。因为 hadoopXML 配置的时候部分 value 不能
有下划线，会报错。

- - 修改 hosts：添加主机名和 IP 的对应，目的是为了使用主机名的时候能够定位（通过 IP）到不同的机器。
 - 输入：`sudo vim /etc/hosts`
 - 把三台机器的 IP 和主机名对应填入 hosts 文件，实例如下所示。填写完之后保存退出。
 - 192.168.8.137 rzxmaster
 - 192.168.8.136 rzxslave1
 - 192.168.8.138 rzxslave2
 - 输入：`source /etc/hosts` 使配置立即生效。
 - 输入：`ping rzxslave1` 可以查看输出是否有对应的 IP 地址，这里先不考虑弄否 ping 通。实例如下：

```
64 bytes from rzxslave1 (192.168.8.136): icmp_seq=40 ttl=64 time=0.216 ms
```

第六步：SSH 免密码登录：

- 1 sudo apt-get update //更新源
2 sudo apt-get install openssh-server //安装 ssh 服务器
3 sudo ps -e |grep ssh //查看 ssh 服务是否启动
4 sudo service ssh start //开启 ssh 服务
5 ssh-keygen -t rsa //生成公钥密钥 一路 enter 就行了
6 cat /home/linuxidc/.ssh/id_rsa.pub >>/home/linuxidc/.ssh/authorized_keys //将公钥添加到用户公钥文件。
- ssh 的配置比较简单教程也很多。不外乎以上几条命令。在三台机子上都进行了如下操作之后，要是 rzxmaster 免密码登录到 rzxslave1，rxslave2。需要把 master 的公钥放到 slave1,2 的 authorized_keys 文件中，这里只需要拷贝然后打开 rzxslave1,2 的 authorized_keys 文件粘贴上即可。保存之后重启虚拟机。重启之后在 rzxmaster，输入 ssh rzxslave1 如果不需要密码就能登录到 rzxslave1 说明成功，同理实验 rxslave2。rxmaster 可以登录到所有 slave 节点则 SSH 设置完毕。

2. 二、集群搭建

通过前面的准备工作我们已经获取到了所有需要的安装包，设置好了静态 IP，配好了 ssh 免密码登录，接下就是集群的安装了。首先我所有的包都是安装在当前用户的根目录下,也就是终端打开的目录(一般是：/home/username username 是当前的用户名)，这个目录是当亲前用户的工作空间我把这个目录的位置记作 basePath=/home/username .这个 basePath 可以根据自己的喜好安装到别的目录下。（basePath=="~"==/home/linuxidc,我这里的 basePath=/home/linuxidc）

三台虚拟机分别如下：rxmaster 是主节点(datanode)，rxslave1,rxslave2 是分支节点(namenode)

192.168.8.137 rxmaster

192.168.8.136 rxslave1

192.168.8.138 rxslave2

为了方便管理这里在主目录建了三个文件夹：Java,spark,hadoop. mkdir Java spark hadoop

现在将jdk,hadoop,scala,spark 的安装包分别传到路径 basePath/Java,basePath/hadoop,basePah/spark 下,(scala 和 spark 的压缩包都放在 spark 文件夹下)。

```
hadoop-2.8.0.tar.gz      scala-2.11.8.tgz
jdk-8u11-linux-x64.tar.gz    spark-2.1.0-bin-without-hadoop.tgz
```

2.1.1 1.Jdk 配置:

- - linuxidc@rxmaster:~\$ cd ~
 - linuxidc@rxmaster:~\$ pwd
 - /home/linuxidc
 - cxin@rxmaster:~\$ cd Java/
 - linuxidc@rxmaster:~/Java\$ tar -zxvf jdk-8u11-linux-x64.tar.gz
 - linuxidc@rxmaster:~/Java\$ mv jdk-8u11-linux-x64 jdk1.8

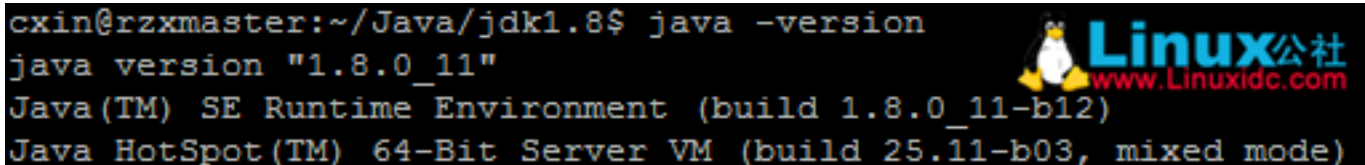
linuxidc@rxmaster:~/Java\$ ls

jdk1.8

```
linuxidc@rzxmaster:~/Java$ cd jdk1.8/
linuxidc@rzxmaster:~/Java/jdk1.8$ ls
bin      include  lib      README.html  THIRDPARTYLICENSEREADME-JAVAFX.txt
COPYRIGHT  javafx-src.zip  LICENSE  release      THIRDPARTYLICENSEREADME.txt
db       jre      man      src.zip
linuxidc@rzxmaster:~/Java/jdk1.8$ pwd
/home/linuxidc/Java/jdk1.8
```

解压并修改了名称，pwd 命令获取到了 JAVA_HOME 路径： /home/linuxidc/Java/jdk1.8 下面配置环境变量：

- `sudo vim /etc/profile` 在文件末尾添加如下代码：
 - `#java`
 - `export JAVA_HOME=/home/linuxidc/Java/jdk1.8`
 - `export JRE_HOME=$JAVA_HOME/jre`
 - `export CLASSPATH=.:$JAVA_HOME/lib:$JRE_HOME/lib:$CLASSPATH`
 - `export PATH=$JAVA_HOME/bin:$JRE_HOME/bin:$PATH`
- 保存退出之后：输入 `source /etc/profile` 使配置立即生效。
- `java -version` 查看是否配置成功，结果如下图，则成功，否则检查配置是否存在问题。



```
cxin@rzxmaster:~/Java/jdk1.8$ java -version
java version "1.8.0_11"
Java(TM) SE Runtime Environment (build 1.8.0_11-b12)
Java HotSpot(TM) 64-Bit Server VM (build 25.11-b03, mixed mode)
```

- 至此 jdk 已经配置完成了。

2.1.2 2.Hadoop 配置：

- 解压并修改名称。(过程同 jdk 一样)
- 配置环境变量： `sudo vim /etc/profile` ,添加如下代码：
 - `#Hadoop`
 - `export HADOOP_HOME=/home/linuxidc/hadoop`
 - `export CLASSPATH=.:$HADOOP_HOME/lib:$CLASSPATH`
 - `export PATH=$PATH:$HADOOP_HOME/bin`
 - `export PATH=$PATH:$HADOOP_HOME/sbin`
 - `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_ROOT_LOGGER=INFO,console`
 - `export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/lib/native`
 - `export HADOOP_OPTS="-Djava.library.path=$HADOOP_HOME/lib"`
- 保存退出，输入 `source /etc/profile`
- 修改 **hadoop** 的四个配置文件： `cd hadoop/etc/hadoop` 新版本的配置文件在 `etc/hadoop` 目录下，配置分两 2 个部分：Hadoop 守护进程(Hadoop Daemons)的环境配置和守护进程的详细配置。
 - `hadoop-env.sh`:hadoop 守护进程[Hadoop 守护进程指 NameNode/DataNode 和 JobTracker/TaskTracker。JobTracker/TaskTracker 是较低版本的资源管理调度模式，已经被 yarn 所取代。]的运行环境配置，这里只设置 JAVA_HOME。编辑 `hadoop-env.sh` 并添加如下代码： `export JAVA_HOME=/home/linuxidc/Java/jdk1.8`
 - `core-site.xml`:
 - `<configuration>`

- <!-- 指定 hdfs 的 namenode 为 rzxmaster -->
- <property>
- <name>fs.defaultFS</name>
- <value>hdfs://**rxmaster**:9000</value>
- </property>
- <!-- Size of read/write buffer used in SequenceFiles. -->
- <property>
- <name>io.**file**.buffer.size</name>
- <value>**131072**</value>
- </property>
- <!-- 指定 hadoop 临时目录,自行创建 -->
- <property>
- <name>hadoop.tmp.**dir**</name>
- <value>/home/linuxidc/hadoop/tmp</value>
- </property>
- </configuration>
- **hdfs-site.xml**:配置 namenode 和 datanode 存储命名空间和 log 的路径
- <configuration>
- <!-- 备份数: 默认为 3-->
- <property>
- <name>dfs.replication</name>
- <value>**2**</value>
- </property>
- <!-- namenode-->
- <property>
- <name>dfs.namenode.name.**dir**</name>
- <value>**file**:/home/linuxidc/hadoop/dfs/name</value>
- </property>
- <!-- datanode-->
- <property>
- <name>dfs.datanode.data.**dir**</name>
- <value>**file**:/home/linuxidc/hadoop/dfs/data</value>
- </property>
- <!--权限控制: **false**: 不做控制即开放给他用户访问 -->
- <property>
- <name>dfs.permissions</name>
- <value>**false**</value>
- </property>
- </configuration>
- **mapred-site.xml**: 配置 MapReduce。
- <configuration>
- <!-- mapreduce 任务执行框架为 yarn-->
- <property>
- <name>mapreduce.framework.name</name>
- <value>yarn</value>
- </property>

- <!-- mapreduce 任务记录访问地址-->
- <property>
- <name>mapreduce.jobhistory.address</name>
- <value>rxmaster:10020</value>
- </property>
- <property>
- <name>mapreduce.jobhistory.webapp.address</name>
- <value>rxmaster:19888</value>
- </property>
- </configuration>

其中还有很多具体应用需要的配置暂且不做配置。参数如下：

mapreduce.map.memory.mb	1536	Larger resource limit for maps.
mapreduce.map.java.opts	-Xmx1024M	Larger heap-size for child jvms of maps.
mapreduce.reduce.memory.mb	3072	Larger resource limit for reduces.
mapreduce.reduce.java.opts	-Xmx2560M	Larger heap-size for child jvms of reduces.
mapreduce.task.io.sort.mb	512	Higher memory-limit while sorting data for efficiency.
mapreduce.task.io.sort.factor	100	More streams merged at once while sorting files.
mapreduce.reduce.shuffle.parallelcopies	50	Higher number of parallel copies run by reduces to fetch outputs from very large number of maps.

-
- **yarn-site.xml**: 配置 resourcesmanager 和 nodemanager
-
- <configuration>
- <property>
- <description>The [hostname](#) of the RM.</description>
- <name>yarn.resourcemanager.address</name>
- <value>rxmaster:8032</value>
- </property>
- <property>
- <name>yarn.resourcemanager.scheduler.address</name>
- <value>rxmaster:8030</value>
- </property>
- <property>

```

○          <name>yarn.resourcemanager.resource-tracker.address</name>
○          <value>rxmaster:8031</value>
○      </property>
○      <property>
○
○          <name>yarn.resourcemanager.admin.address</name>
○          <value>rxmaster:8033</value>
○      </property>
○      <property>
○
○          <name>yarn.resourcemanager.webapp.address</name>
○          <value>rxmaster:8088</value>
○      </property>
○      <property>
○
○          <name>yarn.nodemanager.aux-services</name>
○          <value>mapreduce_shuffle</value>
○      </property>
○      <property>
○
○          <name>yarn.nodemanager.aux-services.mapreduce_shuffle.class</name>
○
○          <value>org.apache.hadoop.mapred.ShuffleHandler</value>
○      </property>
○  </configuration>

```

- **slaves**:配置集群的 DataNode 节点,这些节点是 slaves,NameNode 是 Master。在 conf/slaves 文件中列出所有 slave 的主机名或者 IP 地址,一行一个。配置如下:

```

○ rxslave1
○ rxslave2

```

- 以上的操作把基本的配置工作都完成了。至此已经完成了 hadoop 和 jdk 的配置,将 Java 和 hadoop 文件夹发送到其他节点的主机上。(使用 scp 命令)

- linuxidc@rxmaster:~/hadoop/etc/hadoop\$ cd ~
- linuxidc@rxmaster:~\$ ls
- Desktop Downloads hadoop Music Public Templates
- Documents examples.desktop Java Pictures spark Videos
- //将 java 目录发送到 rxslave1 主机的主目录下
- linuxidc@rxmaster:~\$ scp -r Java linuxidc@rxslave1:/home/linuxidc/
- //将 hadoop 目录发送到 rxslave1 主机的主目录下
- linuxidc@rxmaster:~\$ scp -r hadoop linuxidc@rxslave1:/home/linuxidc/

- 同理将 java, hadoop 发送到 rxslave2 目录下。
- 最后一步在 rxslave1,rxslave2 环境变量中加上 rxmaster 中配置的 java,hadoop 的环境变量(可直接复制粘贴,因为三台机器的配置路径是相同的)
- 至此, hadoop 集群就搭建完成了。

2.1.3 3.hadoop 集群启动:(pwd=/home/linuxidc/hadoop)

- 格式化文件系统 : /bin/hdfs namenode -format
- 启动服务 : sbin/satrt-all.sh

- 查看服务：jps 结果如下：则说明服务启动成功。（具体是否启动还要按启动日志是否报错，有时候未启动成功守护进程也会存在）
 - master : rzxmaster
 - 16002 Jps
 - 7764 NameNode
 - 7992 SecondaryNameNode
 - 8152 ResourceManager
 - slave: rzxslave1, rzxslave2
 - 4529 NodeManager
 - 6968 Jps
 - 4383 DataNode
- 页面访问：192.168.8.137:8088, 如果启动成功可以看到存活的节点如下：



Cluster

- About
- Nodes
- Node Labels
- Applications
- NEW
- NEW SAVING
- SUBMITTED
- ACCEPTED
- RUNNING
- FINISHED
- FAILED
- KILLED
- Scheduler

Tools

Nodes of the cluster

Cluster Metrics					
Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used
0	0	0	0	0	0 B
Cluster Nodes Metrics					
Active Nodes	Decommissioning Nodes	Decommissioned Nodes	Lost Nodes		
2	0	0	0		
Scheduler Metrics					
Scheduler Type	Scheduling Resource Type	Minimum Allocation			
Capacity Scheduler	[MEMORY]	<memory:1024, vCores:1>			
Show 20 entries					
Node Labels	Rack	Node State	Node Address	Node HTTP Address	Last health update
/default-rack		RUNNING	rxslave1:38701	rxslave1:8042	星期一 四月 17 18:27:36 +0800 2017
/default-rack		RUNNING	rxslave2:35931	rxslave2:8042	星期一 四月 17 18:27:36 +0800 2017
Showing 1 to 2 of 2 entries					

- 访问 50070 端口：192.168.8.137:50070

Hadoop Overview Datanodes Datanode Volume Failures Snapshot Startup Progress Utilities

Datanode Information

✓ In service
⚠ Down
🔧 Decommissioned
🔌 Decommissioned & dead

In operation

Show 25 entries Search:

Node	Http Address	Last contact	Capacity	Blocks	Block pool used	Version
✓ rxslave1:50010 (192.168.8.136:50010)	rxslave1:50075	2s	18.58 GB	10	364 KB (0%)	2.8.0
✓ rxslave2:50010 (192.168.8.138:50010)	rxslave2:50075	2s	18.58 GB	10	364 KB (0%)	2.8.0

Showing 1 to 2 of 2 entries

Previous 1 Next

2.1.4 4.Spark 配置：

spark 是依赖与 scala 和 java,hadoop 的，前面配置好了 java,hadoop，这里需要配置 scala 和 spark 的环境以及 spark 的详细信息。

- scala 和 spark 的环境变量配置：(前面已经将 scala 和 spark 的安装包放到了 spark 文件夹下)

- o 解压缩并重命名（参考 jdk 的配置），结果如下：

```
cxin@rxxmaster:~/spark$ ls
scala2.11.8  spark2.1
```

- o 配置环境变量 vim /etc/profile，添加如下代码

- o #scala
- o export SCALA_HOME=/home/linuxidc/spark/scala2.11.8
- o export PATH=\$SCALA_HOME/bin:\$PATH
- o #spark
- o export SPARK_HOME=/home/linuxidc/spark/spark2.1

- o 保存退出，输入 scala 如下图则证明 scala 配置成功

```
cxin@rxxslave1:~$ scala
Welcome to Scala 2.11.8 (Java HotSpot(TM) 64-Bit Server VM, Java 1.8.0_11).
Type in expressions for evaluation. Or try :help.

scala>
```

- o spark 的详细配置：修改 spark 的配置文件(spark 根目录的 conf 目录

下：pwd=/home/linuxidc/spark/spark2.1/conf)

- o **spark-env.sh**:spark 执行任务的环境配置，需要根据自己的机器配置来设置，内存和核心数配置的时候主要不要超出虚拟机的配置，尤其是存在默认值的配置需要仔细查看，修改。

- o
- o export SPARK_DIST_CLASSPATH=\$(/home/linuxidc/hadoop/bin/hadoop classpath)
- o #rxx---config
- o SPARK_LOCAL_DIRS=/home/linuxidc/spark/spark2.1/local #配置 spark 的 local 目录
- o SPARK_MASTER_IP=rxxmaster #master 节点 ip 或 hostname
- o SPARK_MASTER_WEBUI_PORT=8085 #web 页面端口
- o
- o #export SPARK_MASTER_OPTS="-Dspark.deploy.defaultCores=4" #spark-shell 启动使用核数
- o SPARK_WORKER_CORES=1 #Worker 的 cpu 核数
- o SPARK_WORKER_MEMORY=512m #worker 内存大小
- o SPARK_WORKER_DIR=/home/linuxidc/spark/spark2.1/worker #worker 目录
- o SPARK_WORKER_OPTS="-Dspark.worker.cleanup.enabled=true -Dspark.worker.cleanup.appDataTtl=604800" #worker 自动清理及清理时间间隔
- o SPARK_HISTORY_OPTS="-Dspark.history.ui.port=18080 -Dspark.history.retainedApplications=3 -Dspark.history.fs.logDirectory=hdfs://rxxmaster:9000/spark/history" #history server 页面端口>、备份数、log 日志在 HDFS 的位置
- o SPARK_LOG_DIR=/home/linuxidc/spark/spark2.1/logs #配置 Spark 的 log 日志
- o JAVA_HOME=/home/linuxidc/Java/jdk1.8 #配置 java 路径
- o SCALA_HOME=/home/linuxidc/spark/scala2.11.8 #配置 scala 路径
- o HADOOP_HOME=/home/linuxidc/hadoop/lib/native #配置 hadoop 的 lib 路径
- o HADOOP_CONF_DIR=/home/linuxidc/hadoop/etc/hadoop/ #配置 hadoop 的配置路径

- o **spark-default.conf**:

- o spark.master spark://rxxmaster:7077

- o spark.eventLog.enabled true
 - o spark.eventLog.dir hdfs://rxmaster:9000/spark/history
 - o spark.serializer org.apache.spark.serializer.KryoSerializer
 - o spark.driver.memory 1g
 - o spark.executor.extraJavaOptions -XX:+PrintGCDetails -Dkey=value -Dnumbers="one two three"
- o slaves:配置 worker 节点
 - o rxmaster
 - o rxslave1
 - o rxslave2
- 至此 spark 已经配置完成了,将 spark 文件夹(包含 scala 和 spark)发送到其他节点:
 - o scp -r spark linuxidc@rxslave1:/home/linuxidc
 - o scp -r spark linuxidc@rxslave2:/home/linuxidc
- 在 slave 节点配置 scala,spark 的环境变量, 最终三台主机的环境变量配置如下:
 - #java
 - export JAVA_HOME=/home/linuxidc/Java/jdk1.8
 - export JRE_HOME=\$JAVA_HOME/jre
 - export CLASSPATH=.:\$JAVA_HOME/lib:\$JRE_HOME/lib:\$CLASSPATH
 - export PATH=\$JAVA_HOME/bin:\$JRE_HOME/bin:\$PATH
 - #Hadoop
 - export HADOOP_HOME=/home/linuxidc/hadoop
 - export CLASSPATH=.:\$HADOOP_HOME/lib:\$CLASSPATH
 - export PATH=\$PATH:\$HADOOP_HOME/bin
 - export PATH=\$PATH:\$HADOOP_HOME/sbin
 - 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_ROOT_LOGGER=INFO,console
 - export HADOOP_COMMON_LIB_NATIVE_DIR=\$HADOOP_HOME/lib/native
 - export HADOOP_OPTS="-Djava.library.path=\$HADOOP_HOME/lib"
 - #scala
 - export SCALA_HOME=/home/linuxidc/spark/scala2.11.8
 - export PATH=\$SCALA_HOME/bin:\$PATH
 - #spark
 - export SPARK_HOME=/home/linuxidc/spark/spark2.1
 - export PATH=\$SPARK_HOME/bin:\$PATH

- 启动 spark 服务: sbin/start-all.sh

```

cxin@rxmaster:~$ ls
Desktop    Downloads      hadoop  Music      Public  Templates  x1  x3
Documents  examples.desktop  Java   Pictures   spark   Videos    x2
cxin@rxmaster:~$ cd spark/spark2.1/
cxin@rxmaster:~/spark/spark2.1$ ls
bin    examples      LICENSE  logs    R        sbin        yarn
conf   hs_err_pid5599.log  licenses NOTICE  README.md spark-warehouse
data   jars          local    python  RELEASE  worker

```

```

cxin@rxzmaster:~/spark/spark2.1$ sbin/start-all.sh
starting org.apache.spark.deploy.master.Master, logging to /home/cxin/spark/spark2.1/logs/spark-cxin-org.apache.spark.deploy.master.Master-1-rxzmaster.out
rxzslave1: starting org.apache.spark.deploy.worker.Worker, logging to /home/cxin/spark/spark2.1/logs/spark-cxin-org.apache.spark.deploy.worker.Worker-1-rxzslave1.out
rxzslave2: starting org.apache.spark.deploy.worker.Worker, logging to /home/cxin/spark/spark2.1/logs/spark-cxin-org.apache.spark.deploy.worker.Worker-1-rxzslave2.out
rxzmaster: starting org.apache.spark.deploy.worker.Worker, logging to /home/cxin/spark/spark2.1/logs/spark-cxin-org.apache.spark.deploy.worker.Worker-1-rxzmaster.out

```

- 查看服务(rxzmaster): jps 由于在配置文件 slaves 中添加了 rxzmaster,所以在此处有一个 Worker 进程。

```

16656 Master
16784 Worker

```


- 查看服务(rxzslave1,rxzslave2): jps

```

7073 Worker

```

- 页面查看: 访问 192.168.8.137:8085(8085 端口是设置在 spark-env.sh 中的 SPARK_MASTER_WEBUI_PORT,可自行设置), 结果如下: 则说明成功了。

 **Spark Master at spark://rxzmaster:7077**

URL: spark://rxzmaster:7077
 REST URL: spark://rxzmaster:6066 (cluster mode)
 Alive Workers: 3
 Cores in use: 3 Total, 0 Used
 Memory in use: 1536.0 MB Total, 0.0 B Used
 Applications: 0 Running, 0 Completed
 Drivers: 0 Running, 0 Completed
 Status: ALIVE

Workers

Worker Id	Address	State	Cores	Memory
worker-20170417173137-192.168.8.136-38442	192.168.8.136:38442	ALIVE	1 (0 Used)	512.0 MB (0.0 B Used)
worker-20170417173138-192.168.8.138-43290	192.168.8.138:43290	ALIVE	1 (0 Used)	512.0 MB (0.0 B Used)
worker-20170417173142-192.168.8.137-44540	192.168.8.137:44540	ALIVE	1 (0 Used)	512.0 MB (0.0 B Used)

- spark 示例运行: ./bin/run-example SparkPi 2>&1 | grep "Pi is roughly" 计算圆周率。如下图所示

```

cxin@rxzslave2:~$ cd spark/spark2.1/
cxin@rxzslave2:~/spark/spark2.1$ ./bin/run-example SparkPi 2>&1 | grep "Pi is roughly"
Pi is roughly 3.14479572397862

```

- 在 8085 端口可以看到运行的任务:

Completed Applications

Application ID	Name	Cores	Memory per Node	Submitted Time	User	State	Duration
app-20170417174803-0000	Spark Pi	3	512.0 MB	2017/04/17 17:48:03	cxin	FINISHED	30 s

- 至此说明 spark 配置成功, 当然其他的内容需要在使用的時候去排除 bug, 如果出现问题需要查看日志信息找到问题出现的原因, 然后修改配置。

2.1.5 5.配置总结:

在配置的过程中出现了很多的问题, 包括: namenode, datanode 启动失败, 伪成功(存在守护线程但是实际服务未启动), spark-shell 启动失败爆出资源无法分配。有租在配置的时候没有做好问题的记录, 所以这里列不出具体的异常信息。这里一部分异常是由于 IP 的设置和 hosts 中的配置不对应, 一部分是 ssh 连接失败, 资源配置没有根据虚拟机的配置做匹配。建议不要多次格式化 namenode, 每次格式化都会生成一个 clusterID, 多个 clusterID 导致启动报错, 如需格式化就必须清除 dfs/name, dfs/data 下的文件。

hadoop, spark 只是基础组建, 提供文件系统和数据运算, hadoop 生态还包括 hive, habse, kafka... 等数据存储和分析的组件, 后面可以在此基础上一步步安装。

更多 Hadoop 相关信息见 [Hadoop 专题页面](http://www.linuxidc.com/topicnews.aspx?tid=13) <http://www.linuxidc.com/topicnews.aspx?tid=13>

本文永久更新链接地址: <http://www.linuxidc.com/Linux/2017-04/143103.htm>

欢迎点击这里的链接进入精彩的[Linux公社](http://www.Linuxidc.com)网站

Linux公社（www.Linuxidc.com）于2006年9月25日注册并开通网站，Linux现在已经成为一种广受关注和支持的一种操作系统，IDC是互联网数据中心，LinuxIDC就是关于Linux的数据中心。

[Linux公社](http://www.Linuxidc.com)是专业的Linux系统门户网站，实时发布最新Linux资讯，包括Linux、Ubuntu、Fedora、RedHat、红旗Linux、Linux教程、Linux认证、SUSE Linux、Android、Oracle、Hadoop、CentOS、MySQL、Apache、Nginx、Tomcat、Python、Java、C语言、OpenStack、集群等技术。

Linux公社（LinuxIDC.com）设置了有一定影响力的Linux专题栏目。

Linux公社 主站网址：www.linuxidc.com 旗下网站：www.linuxidc.net

包括：[Ubuntu 专题](#) [Fedora 专题](#) [Android 专题](#) [Oracle 专题](#) [Hadoop 专题](#)
[RedHat 专题](#) [SUSE 专题](#) [红旗 Linux 专题](#) [CentOS 专题](#)



Linux 公社微信公众号：[linuxidc_com](#)



微信扫一扫

Linuxidc.com

订阅专业的最新Linux资讯及开源技术教程。

搜索微信公众号：[linuxidc_com](#)