

Mac版hadoop,Spark,Hbase的伪分布式的安装(全网最全)

• 注意:

- 如果你利用Mac从事开发工作，请务必去了解Homebrew，他会给你带来很大的便利，下文会细说
- 但是新版的Homebrew有个大问题，他在帮你装软件时会安装最新版，并且大部分软件不支持版本选择，所以要有取舍
- Hadoop3.X和Hadoop2.X有较多不同，所以选择目前的额主流Hadoop2.X，大数据分布式搭建尤其 注意版本兼容问题
- 在搭建前要先了解Hadoop和Spark的相关基础知识，* **Hbase如果你不需要的可以不用安装**

前言

- 如果你能具备上面的4条，接下来的搭建也相当的快速，大部分的时间是耗在下载和账号注册上面，我会在文章最后附上我的百度云安装包，相信会给大家节省60%以上的安装时间
- 安装过程中难免遇到文档中未提及的报错，需要对Linux系统终端操作（MacOS类似）有所了解
- 很多人更多的想学习Spark的pyspark，但是Spark是需要依赖Hadoop的HDFS以及YARN的框架
- **所有版本附Mac版本的安装包的百度云链接**

版本

系统：MacOS Catalina 10.15.4

JDK：jdk1.8.0_211

scala：scala 2.12

Hadoop：hadoop-2.7.7

spark：spark-2.3.0

Hbase：hase 1.3.5

命令终端：系统自带终端其环境参数在~/.base_profile下，网上的教程有的是iTerm，其参数是~/.zshrc中

1、SSH免密登录的配置

配置ssh是为了能实现免密登录，这样方便远程管理Hadoop并无需登录密码在Hadoop集群上共享文件资源。

- 生成SSH Keys:

```
ssh-keygen -t rsa -P ""
```

- 授权你的公钥到本地可以无需密码实现登录。上个命令会在当前用户目录中的.ssh文件夹中生成id_rsa文件，在执行如下命令：

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

- 用如下的命令来测试是否可以免密登录

```
ssh localhost
```

问题汇总:

ssh的配置其实对于新Mac还是有个坑的，你可能按照教程依旧会有如下的报错：

链接拒绝：

```
ssh: connect to host localhost port 22: Connection refused
```

```
~ % ssh localhost  
ssh: connect to host localhost port 22: Connection refused 链接拒绝
```

执行如下命令：

```
sudo systemsetup -f -setremotelogin on
```

又会出现如下错误：

```
setremotelogin: Turning Remote Login on or off requires Full Disk Access  
privileges.
```

这个问题的是Mac没有打开远程登录：

- a、先查看有没有打开“远程登录”**系统偏好设置 -> 共享 -> 远程登录**



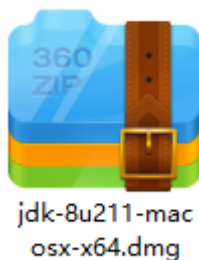
- b、打开sshd的权限，mac的隐私设置中终端sshd没有开启磁盘完全访问权限，勾选上



2、安装jdk1.8以及环境变量的配置

2.1、[官网](#)下载如下的JDK安装包，

我们需要1.8版本目前看最好：但是这个安装包需要十分繁琐的注册Oracle账号，嫌麻烦的百度云链接最后汇总奉上：



名称解释：8u211指的是java8版本号为211的JDK安装包

双击安装，一路Next

2.2、配置系统的环境变量：告诉系统环境你安装到哪了

如果是这种安装方式，那你的路径也应该是这个：

“/Library/Java/JavaVirtualMachines/jdk1.8.0_211.jdk/Contents/Home”

- 打开.bash_profile（为啥是.bash_profile在“版本”模块讲过）

```
open ~/.bash_profile
```

- 在文件末尾加入如下的一句话就是JAVA_HOME的路径(注意这里的jdk如果不是我这个版本要换成自己的),保存

```
/Library/Java/JavaVirtualMachines/jdk1.8.0_211.jdk/Contents/Home
```

- 使配置立即生效

```
source ~/.bash_profile
```

- 验证JDK1.8是否安装成功

```
java -version
```

- 得到如下图片表示成功

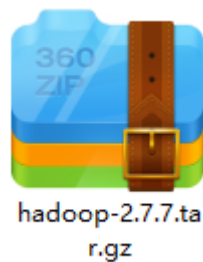
```
~ % java -version
java version "1.8.0_211" ← 显示版本号
Java(TM) SE Runtime Environment (build 1.8.0_211-b12)
Java HotSpot(TM) 64-Bit Server VM (build 25.211-b12, mixed mode)
```

3、安装Hadoop2.7.7

写这篇文档的区别就是从这开始的，就像我说的homebrew是安装最新版本的hadoop，也就是hadoop3.X但是这个并不兼容，所以我们只能官网下载而抛弃homebrew，话说homebrew下载是真的香，我们要多配置一点东西

网上的文档有很多参数是配置不全的，实际跑起来还是有不少问题，尽量以我综合的为主吧

- [官网下载](#) 然后解压到自己想放的目录，我这边是直接放到home目录，然后自己创建dev目录的，



- 同样安装包后面汇总到百度云

```
# 这个代码作参考，移动加解压
mv ~/Downloads/hadoop-2.7.7.tar.gz ~/dev/hadoop
tar -zxvf hadoop-2.7.2.tar.gz
```

- 在~/.bash_profile中添加hadoop的环境变量（比网上的有的要复杂，但都是有用的，建议添加）
- 再次提示配置完环境后都要执行 `source ~/.bash_profile`来使你的环境变量立即生效

```
HADOOP_HOME="/Users/xxx/dev/hadoop/hadoop-2.7.7"
export HADOOP_HOME
export PATH=$PATH:HADOOP_HOME/sbin:$HADOOP_HOME/bin
export LD_LIBRARY_PATH=$HADOOP_HOME/lib/native/
export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/lib/native
export HADOOP_OPTS="-Djava.library.path=$HADOOP_HOME/lib/native:$HADOOP_COMMON_LIB_NATIVE_DIR"
```

4、配置Hadoop

- 配置hadoop-env.sh

这个在安装的配置文件中：比如我的路径：“/Users/xxx/dev/hadoop/hadoop-2.7.7/etc/hadoop”
主要看JAVA_HOME的注释有没有打开，（一般都是打开的，不需要配置）

```
# The only required environment variable is JAVA_HOME. All others are
# optional. When running a distributed configuration it is best to
# set JAVA_HOME in this file, so that it is correctly defined on
# remote nodes.

# The java implementation to use.
export JAVA_HOME=${JAVA_HOME} ← 要打开注释

# The jsvc implementation to use. Jsvc is required to run secure datanodes
# that bind to privileged ports to provide authentication of data transfer
# protocol. Jsvc is not required if SASL is configured for authentication of
# data transfer protocol using non-privileged ports.
#export JSVC_HOME=${JSVC_HOME}
```

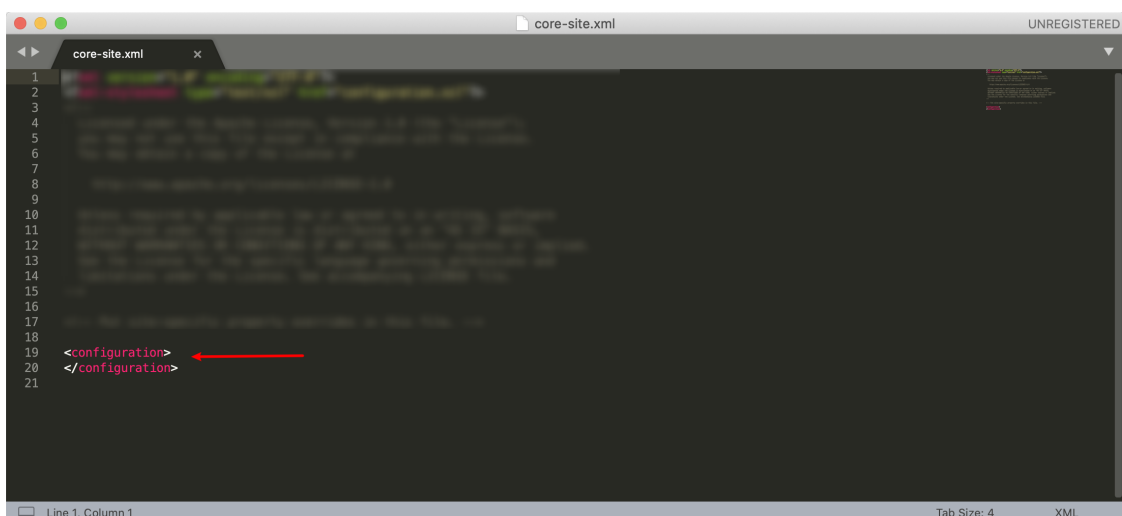
- 统一说明这些配置文件都在此目录，但是都是XXXX.site.xml.template,就是说这些都是模板，你只要执行命令copy一份就好

```
cp core-site.xml.template core-site.xml
```

- 配置 core-site.xml

该配置文件用于指定NameNode的主机名和端口（后续的Hbase也需要用），在之间添加如下内容

```
<configuration>
  <property>
    <name>fs.defaultFS</name>
    <value>hdfs://localhost:9000</value>
  </property>
  <property>
    <name>hadoop.tmp.dir</name>
    <value>file:/Users/xxx/dev/hadoop/hadoop-2.7.7/tmp</value>
  </property>
  <property>
    <name>io.file.buffer.size</name>
    <value>131702</value>
  </property>
</configuration>
```



- 配置hdfs-site.xml

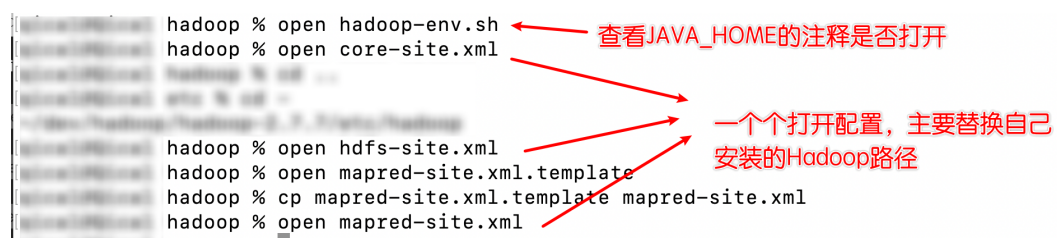
这个文件是制定了HDFS的默认参数以及副本数，因为仅运行一个节点，所以这里的副本用1

```
<configuration>
  <property>
    <name>dfs.namenode.name.dir</name>
    <value>file:/Users/xxx/dev/hadoop/hadoop-2.7.7/tmp/hdfs/name</value>
  </property>
  <property>
    <name>dfs.datanode.data.dir</name>
    <value>file:/Users/xxx/dev/hadoop/hadoop-2.7.7/tmp/hdfs/data</value>
  </property>
  <property>
    <name>dfs.replication</name>
    <value>1</value>
  </property>
  <property>
    <name>dfs.namenode.secondary.http-address</name>
    <value>localhost:9001</value>
  </property>
  <property>
    <name>dfs.webhdfs.enabled</name>
    <value>true</value>
  </property>
</configuration>
```

- 配置mapred-site.xml

这个文件指定了JobTracker的主机名与端口

```
<configuration>
  <property>
    <name>mapred.job.tracker</name>
    <value>localhost:9001</value>
  </property>
</configuration>
```



Terminal output showing the steps to configure Hadoop:

```
hadoop % open hadoop-env.sh
hadoop % open core-site.xml
hadoop % open hdfs-site.xml
hadoop % open mapred-site.xml.template
hadoop % cp mapred-site.xml.template mapred-site.xml
hadoop % open mapred-site.xml
```

Annotations (in red):

- 查看JAVA_HOME的注释是否打开 (Check if the JAVA_HOME comment is opened) - points to `hadoop % open hadoop-env.sh`
- 一个个打开配置，主要替换自己安装Hadoop路径 (Open configurations one by one, mainly replacing the Hadoop installation path) - points to `hadoop % open hdfs-site.xml` and `hadoop % open mapred-site.xml.template`

- 运行Hadoop

进行一些初始化操作，以及常用命令

```
// 初始化HDFS，进入Hadoop安装目录
bin/hdfs namenode -format
```

启动和关闭HDFS

// 启动和关闭HDFS服务

// 开启NameNode和DataNode守护进程(注意这一步要是报错就去hadoop的主目录sbin文件中执行)
sbin/start-dfs.sh

// 关闭的命令
sbin/stop-dfs.sh

在浏览器输入: <http://localhost:50070> 可以看到

The screenshot shows the Hadoop NameNode web interface in a browser window. The address bar shows 'localhost'. The page has a green header with 'Hadoop' and a navigation menu with 'Overview', 'Datanodes', 'Datanode Volume Failures', 'Snapshot', 'Startup Progress', and 'Utilities'. The main content area is titled 'Overview' and shows 'localhost:9000' (active). Below this is a table with the following information:

Started:	Fri Jun 12 19:31:03 CST 2015
Version:	2.7.7, rc1aad94bd27cd7
Compiled:	2018-07-18T22:47Z by
Cluster ID:	CID-ef4c2961-cdb7-473
Block Pool ID:	BP-662160353-192.168

Below the table is a 'Summary' section. It states: 'Security is off.', 'Safemode is off.', '1 files and directories, 0 blocks = 1 total filesystem object(s).', 'Heap Memory used 103.91 MB of 222 MB Heap Memory. Max Heap Memory is 889 MB.', and 'Non Heap Memory used 41.54 MB of 42.94 MB Committed Non Heap Memory. Max Non Heap Memory is -1 B.' Below this is a table with the following information:

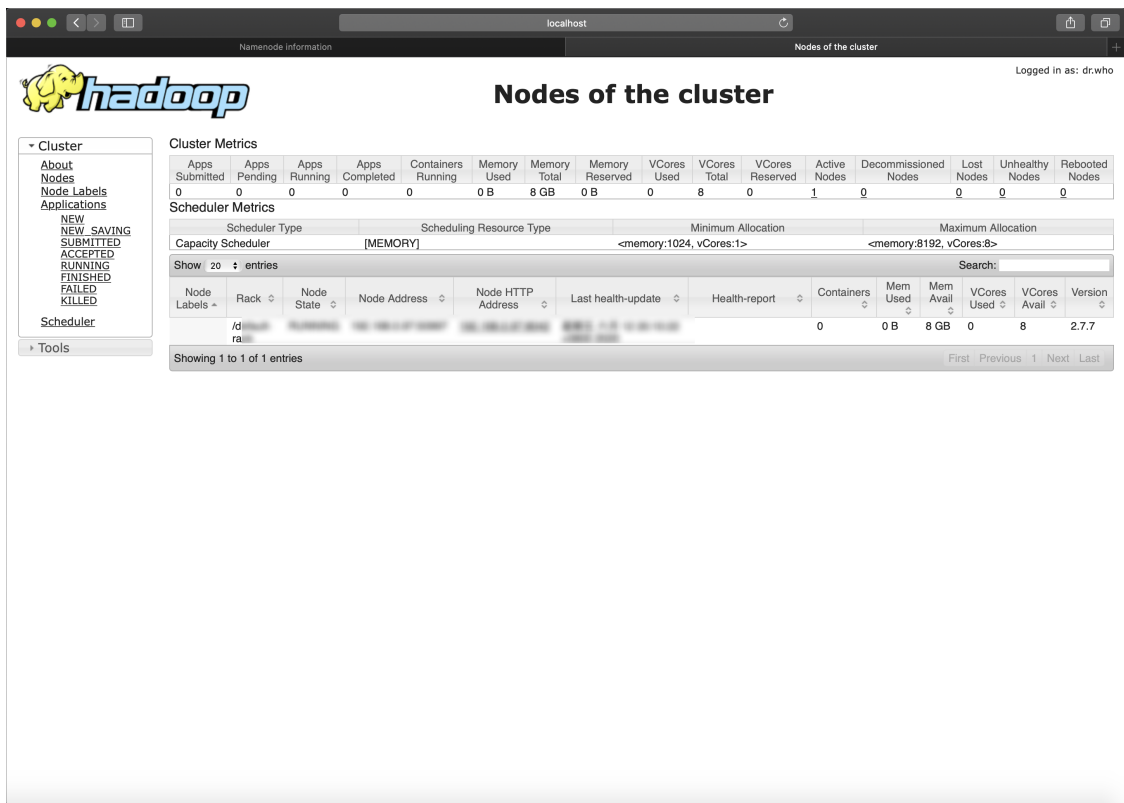
Configured Capacity:	889 MB
DFS Used:	1 MB (0%)
Non DFS Used:	10 MB (1%)
DFS Remaining:	889 MB (99%)
Block Pool Used:	1 MB (0%)
DataNodes usages% (Min/Median/Max/stdDev):	1.00% / 1.00% / 1.00% / 1.00%
Live Nodes	1 (Recommended: 1)
Dead Nodes	0 (Recommended: 0)
Decommissioning Nodes	0

启动/关闭YARN服务

// 开启NameNode和DataNode守护进程(注意这一步要是报错就去hadoop的主目录sbin文件中执行)
sbin/start-yarn.sh

// 关闭
sbin/stop-yarn.sh

在浏览器输入: <http://localhost:8088> 可以看到:



同时启动和关闭上面两个

```
// 开启NameNode和DataNode守护进程(注意这一步要是报错就去hadoop的主目录sbin文件中执行)
sbin/start-all.sh
// 关闭
sbin/stop-all.sh
```

查看进程的命令：

```
jps
```

5、安装配置Scala2.12

幸运的是homebrew提供了好几个版本的scala，我们可以使用homebrew来安装，至于为什么安装它，是由于Spark是用Scala设计的，所以必须要安装

我们执行 `brew search Scala`发现有多版本这里执行 `brew scala@2.12`

- 配置环境变量

```
export SCALA_HOME=/usr/local/opt/scala@2.12
PATH="$SCALA_HOME/bin:$PATH"
```

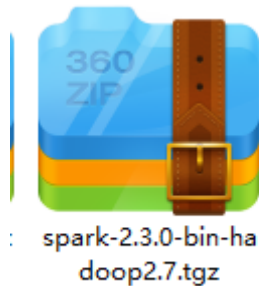
- 查看版本

```
scala -version
```

```
hadoop-2.7.7 % scala -version
Scala code runner version 2.12.11 -- Copyright 2002-2020, LAMP/EPFL and Lightbend, Inc.
```

6、安装Spark

- [官网下载](#) 我们选择适配的hadoop2.7版本的，这里注意Homebrew下载的有spark 和 apache-spark的选择区别，并且版本是最新的，为了不必要的版本冲突，这里还是选择自己下载，后期百度云汇总



- 创建自己的目录进行解压和改名，（我这里将其放在hadoop同级目录下 ~/dev/spark，这个按自己的目录要求）
- 配置spark的环境变量，在~/.bash_profile最后添加

```
export SPARK_HOME=/Users/xxx/dev/spark
export PATH=$PATH:$SPARK_HOME/bin
```

- 从Spark安装目录进入conf，从template复制一份sh文件

```
[xxx ~ % cd ~/dev/spark
spark % ls
LICENSE      R            RELEASE     conf         examples    kubernetes  python      yarn
NOTICE       README.md   bin         data         jars        licenses    sbin
[xxx ~ % cd conf
conf % ls
docker.properties.template  log4j.properties.template  slaves.template          spark-env.sh.template
fairscheduler.xml.template  metrics.properties.template spark-defaults.conf.template
[xxx ~ % cp spark-env.sh.template spark-env.sh
conf % ls
docker.properties.template  log4j.properties.template  slaves.template          spark-env.sh
fairscheduler.xml.template  metrics.properties.template spark-defaults.conf.template  spark-env.sh.template
[xxx ~ %]
```

进入spark的配置文件目录

copy 模板，进行配置

```
open spark-env.sh
```

- 修改配置文件spark-env.sh，在其中添加（很多的文档是没有这些文档的配置的，其实是不完善的）

```
export SCALA_HOME=/usr/local/opt/scala@2.12
export SPARK_HOME=/Users/xxx/dev/spark
export HADOOP_HOME=/Users/xxx/dev/hadoop/hadoop-2.7.7
export
JAVA_HOME=/Library/Java/JavaVirtualMachines/jdk1.8.0_211.jdk/Contents/Home
export SPARK_WORKER_MEMORY=1g
```

- 编辑文件slaves：添加所有主机的hostname（添加对方的hostname和ip的关系），一般里面有localhost，不需要作出修改

```
cp slaves.template slaves
vi slaves
```


- 进入目录，进行启动和关闭：

```
// 进入Spark安装目录
cd /usr/local/spark/bin

// 启动
sbin/start-all.sh

// 关闭
sbin/stop-all.sh
```

- 启动后在浏览器中进行检查：

 **Spark Master at spark://192.168.0.91:7077**

URL: spark://192.168.0.91:7077
REST URL: spark://192.168.0.91:6066 (cluster mode)
Alive Workers: 1
Cores in use: 12 Total, 0 Used
Memory in use: 1024.0 MB Total, 0.0 B Used
Applications: 0 Running, 0 Completed
Drivers: 0 Running, 0 Completed
Status: ALIVE

Workers (1)

Worker Id	Address	State	Cores	Memory
worker-20200614163354-192.168.0.91	192.168.0.91:295	ALIVE	12 (0 Used)	1024.0 MB (0.0 B Used)

Running Applications (0)

Application ID	Name	Cores	Memory per Executor	Submitted Time	User	State	Duration
----------------	------	-------	---------------------	----------------	------	-------	----------

Completed Applications (0)

Application ID	Name	Cores	Memory per Executor	Submitted Time	User	State	Duration
----------------	------	-------	---------------------	----------------	------	-------	----------

7、Hbase的安装

至此hadoop和spark的伪分布式安装和配置基本完成，这里继续安装相关组件
这里有个好消息，在测试之后我发现homebrew是可以直接安装兼容的Hbase的，

- 直接按照homebrew来直接安装

```
brew install hbase
```

- 进入HBase配置文件目录

```
cd /usr/local/Cellar/hbase/1.3.5/libexec/conf
```

- 配置hbase-env.sh，配置JAVA_HOME和hbase_classpath（指向hadoop的配置文件目录）环境变量

```
export HBASE_CLASSPATH=/Users/xxx/dev/hadoop/hadoop-2.7.7/etc/hadoop
export HBASE_MANAGES_ZK=true
export HBASE_HOME=/usr/local/Cellar/hbase/1.3.5/libexec
export HBASE_LOG_DIR=${HBASE_HOME}/logs
export HBASE_REGIONSERVERS=${HBASE_HOME}/conf/regionserver
```

- 编辑hbase-site.xml文件 Hbase.rootdir要配置hdfs上的路径，（对照<>,有就修改，没有就添加）

```
<property>
  <name>hbase.rootdir</name>
  <value>hdfs://localhost:9000/hbase</value>
</property>
<property>
  <name>hbase.cluster.distributed</name>
  <value>true</value>
</property>
<property>
  <name>dfs.replication</name>
  <value>1</value>
</property>
```

hbase.rootdir这个值需要设置成之前Hadoop的core-site.xml配置的fs.default.name值

- 运行Hbase(在运行Hbase之前, 我们需要先运行Hadoop伪分布式模式, 在运行Hbase)

```
// 进入Hadoop目录
sbin/start-dfs.sh # 运行hadoop

// 进入Hbase目录
bin/start-hbase.sh
```

至此, 基本的部署操作完成: haoddp/Spark/JDK 安装包汇总如下[链接](#)提取码: vk45

如果过期了可以添加公众号, 会在那边维护这个链接的更新, 也欢迎你和我讨论关于数据的东西:



8、参考链接:

[Hadoop \(O\) macOS上搭建伪分布式Hadoop环境](#)

[解决mac下 ssh: connect to host localhost port 22: Connection refused](#)

[Mac系统安装JDK1.8及环境变量配置](#)

[学习Spark——环境搭建 \(Mac版\)](#)

[Mac下Spark2.1.0的伪分布式安装配置](#)

[Hadoop+HBase+Spark伪分布式整合部署\(mac\)](#)