

Hadoop

Reference: <http://codingxiaoxw.cn/2016/12/06/59-mac-hadoop/>

1. 安裝Homebrew与Cask (Cask有點問題, 要用 **\$ brew install brew-cask-completion**)

<https://www.jianshu.com/p/7d055bebab46>

2. 安裝JAVA (/usr/libexec/java_home -V 或是 java -version)

https://blog.csdn.net/vvv_110/article/details/72897142

記得設置環境變數

echo \$JAVA_HOME 去檢查

/Library/Java/JavaVirtualMachines/jdk-10.0.1.jdk/Contents/Home

用 **\$vim ~/.bash_profile** 去設置環境變數, 然後用 **\$source .bash_profile** 去生效

3. 配置ssh

\$ ssh-keygen -t rsa

Generating public/private rsa key pair.

Enter file in which to save the key (/Users/alumi5566/.ssh/id_rsa):

Enter passphrase (empty for no passphrase):

Enter same passphrase again:

Your identification has been saved in /Users/alumi5566/.ssh/id_rsa.

Your public key has been saved in /Users/alumi5566/.ssh/id_rsa.pub.

The key fingerprint is:

SHA256:f3G/tp2TyvIR+VAAovP+8PYSYi738KVlhBjKMZXN9o

alumi5566@ucrwp-1-7-10-25-26-210.wnet.ucr.edu

The key's randomart image is:

+---[RSA 2048]-----+

```
|      . ... |  
|      . . . |  
|      + . o . |  
|      . B = . o |  
|      + oSB E.+ |  
|      . o o+o. o+. |  
|      o=o oo .o|  
|      ..oB=+ .++|  
|      o.o*==o++|
```

+---[SHA256]-----+

安裝成功的話

\$ ssh localhost

Enter passphrase for key '/Users/alumi5566/.ssh/id_rsa':

Last login: Mon Jun 4 13:01:03 2018 from ::1

4. 安裝Hadoop

\$ brew install hadoop

Hadoop安裝在 /usr/local/Cellar/hadoop/3.1.0/ 之下 (版本號可能不同)

(a) 打開 /usr/local/Cellar/hadoop/3.1.0/libexec/etc/hadoop/hadoop-env.sh, 把export HADOOP_OPTS="-Djava.net.preferIPv4Stack=true"

修改成

export HADOOP_OPTS="\$HADOOP_OPTS -Djava.net.preferIPv4Stack=true

-Djava.security.krb5.realm= -Djava.security.krb5.kdc="

export JAVA_HOME="/Library/Java/JavaVirtualMachines/jdk-10.0.1.jdk/Contents/Home"

(b) 打開 /usr/local/Cellar/hadoop/3.1.0/libexec/etc/hadoop/core-site.xml

在<configuration>中間加入 :

<property>

<name>hadoop.tmp.dir</name>

```

    <value>/usr/local/Cellar/hadoop/hdfs/tmp</value>
    <description>A base for other temporary directories.</description>
</property>
<property>
    <name>fs.default.name</name>
    <value>hdfs://localhost:8020</value>
</property>

```

(c) 打開 /usr/local/Cellar/hadoop/3.1.0/libexec/etc/hadoop/mapred-site.xml
在<configuration>中間加入

```

<property>
    <name>mapred.job.tracker</name>
    <value>localhost:8021</value>
</property>

```

(d) 設置hdfs的默認備份方式，在偽分布式系統中，需要修改為1
打開 /usr/local/Cellar/hadoop/3.1.0/libexec/etc/hadoop/hdfs-site.xml
在<configuration>中間加入

```

<property>
    <name>dfs.replication</name>
    <value>1</value>
</property>

```

(e) 格式化新安裝的HDFS, 並通過創建存儲目錄和初始化元數據創新空的文件系統
在/usr/local/Cellar/hadoop/3.1.0/libexec/etc/hadoop/底下

\$hdfs namenode -format

5. 啟動Hadoop (script都放在sbin底下)

./start-dfs.sh //啟動HDFS

./stop-dfs.sh //停止HDFS

用sudo啟動時遇到錯誤訊息 (無解)

```

ucrwpa-1-7-10-25-26-210:sbin alum15566$ sudo ./start-dfs.sh
Password:
Starting namenodes on [localhost]
ERROR: Attempting to operate on hdfs namenode as root
ERROR: but there is no HDFS_NAMENODE_USER defined. Aborting operation.
Starting datanodes
ERROR: Attempting to operate on hdfs datanode as root
ERROR: but there is no HDFS_DATANODE_USER defined. Aborting operation.
Starting secondary namenodes [ucrwpa-1-7-10-25-26-210.wnet.ucr.edu]
ERROR: Attempting to operate on hdfs secondarynamenode as root
ERROR: but there is no HDFS_SECONDARYNAMENODE_USER defined. Aborting operation.
2018-06-04 15:34:04,148 WARN util.NativeCodeLoader: Unable to load native
-hadoop library for your platform... using builtin-java classes where applicable
ucrwpa-1-7-10-25-26-210:sbin alum15566$

```

一般啟動時遇到錯誤訊息

Starting namenodes on [localhost]

localhost: U@localhost: Permission denied (publickey,password,keyboard-interactive).

Starting datanodes

解法：

Generate new keygen.

\$ssh-keygen -t rsa -P "" -f ~/.ssh/id_rsa

Register key gen:

\$cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys

如果成功開啟，另外再用browser連接

http://localhost:9870

Overview 'localhost:8020' (active)

Started:	Mon Jun 04 15:54:29 -0700 2018
Version:	3.1.0, r16b70619a24cdc5d3b0fc4b58ca77238ccbe6d
Compiled:	Thu Mar 29 17:00:00 -0700 2018 by centos from branch-3.1.0
Cluster ID:	CID-bf28b6dc-9d46-4446-af21-8b414b702d10
Block Pool ID:	BP-1352234239-10.25.26.210-1528152397334

Summary

Security is off.

Safemode is off.

1 files and directories, 0 blocks (0 replicated blocks, 0 erasure coded block groups) = 1 total filesystem object(s).

Heap Memory used 66.48 MB of 156 MB Heap Memory. Max Heap Memory is 2 GB.

Non Heap Memory used 56.2 MB of 60.14 MB Committed Non Heap Memory. Max Non Heap Memory is <unbounded>.

Configured Capacity:	112.8 GB
Configured Remote Capacity:	0 B

- 啟動yarn (mapreduce framework) (script都放在sbin底下)
 - `./start-yarn.sh` //启动yarn, 一个MapReduce框架
 - `./stop-yarn.sh` //停止yarn
- 也可以一鍵啟動全部
 - `./start-all.sh` ##启动Hadoop
 - `./stop-all.sh` ##停止Hadoop

用mahout來implement kmeans

```
$brew install mahout
```

裝在/usr/local/Cellar/mahout/0.13.0

編輯環境變數\$vim ~/.bash_profile

```
export MAHOUT_HOME=/usr/local/Cellar/mahout/0.13.0/libexec
MAHOUT_CONF_DIR=$MAHOUT_HOME/
export PATH=$MAHOUT_HOME/bin:$PATH
$source ~/.bash_profile
```

然後然後

```
$time bin/hadoop jar /usr/local/Cellar/mahout/0.13.0/libexec/mahout-examples-0.13.0-job.jar
org.apache.mahout.clustering.syntheticcontrol.kmeans.Job
```

Spark

- 安裝scala, 確認在/usr/local下面有scala的資料夾 (我們放在/usr/local/Cellar/scala/2.12.6 底下)
 - `$brew install scala`
 - 修改環境變量
 - `$sudo vim /etc/profile`

```
export SCALA_HOME=/usr/local/Cellar/scala/2.12.6
```

```
export PATH=$PATH:$SCALA_HOME/bin
```

記得用 **\$source /etc/profile** 讓修改生效

(輸入\$scala測試一下)

2. 從apache下載spark安裝包 (記得選對應的版本)



Download Apache Spark™

1. Choose a Spark release: 2.3.0 (Feb 28 2018)

2. Choose a package type: Pre-built for Apache Hadoop 2.7 and later

3. Download Spark: [spark-2.3.0-bin-hadoop2.7.tgz](#)

4. Verify this release using the [2.3.0 signatures and checksums](#) and [project release KEYS](#).

Note: Starting version 2.0, Spark is built with Scala 2.11 by default. Scala 2.10 users should download the Spark source package and build with Scala 2.10 support.

解壓縮之後放到 /usr/local 底下，並改名成/spark

一樣要去修改環境變量 **\$sudo vim /etc/profile**

```
export SPARK_HOME=/usr/local/spark
```

```
export PATH=$PATH:$SPARK_HOME/bin
```

3. 把/usr/local/spark/conf/spark-env.sh.template 複製一份在同樣資料夾在，名稱為spark-env.sh

在/usr/local/spark/conf/spark-env.sh裡面加入以下內容

```
export SCALA_HOME=/usr/local/Cellar/scala/2.12.6
```

```
export SPARK_MASTER_IP=localhost
```

```
export SPARK_WORKER_MEMORY=4g
```

4. 跑**\$spark-shell** 時出現很多錯誤訊息，

改用scala-2.11.12 (手動下載到/usr/local/Cellar/scala)

還是很多錯誤訊息，從jdk下手

5. 改裝jdk1.7

(<http://www.oracle.com/technetwork/java/javase/downloads/java-archive-downloads-javase7-521261.html>)

```
$vim ~/.bash_profile
```

```
JAVA_HOME=/Library/Java/JavaVirtualMachines/jdk1.7.0_80.jdk/Contents/Home
```

6. 再把scala 改成 scala-2.11.8

一樣下載之後解壓縮，放到 /usr/local/scala

```
$sudo vim /etc/profile
```

```
export SCALA_HOME=/usr/local/scala
```

還是有錯誤

7. 最後用這個

<https://stackoverflow.com/questions/46436879/spark-shell-failed-to-initialize-compiler-error-on-a-mac>

```
$ brew cask install java
```

```
$ brew install scala
```

```
$ brew install apache-spark
```

然後 **\$sudo spark-shell**

```
Spark context available as 'sc' (master = lo  
Spark session available as 'spark'.  
Welcome to
```



version 2.3.1


(如果之後要手動安裝, scala 2.11 + jdk 1.7 或1.8可能比較好, 部落主試過可以)

用scala太麻煩了，想辦法裝pyspark

1. 去把 `/etc/profile` 裡面的 `SPARK_HOME=/usr/local/spark` 改成我們用brew裝的那個版本 (`/usr/local/Cellar/apache-spark/2.3.1/bin`)

或是乾脆著解掉 (brew會自動幫我們加路徑)

```
$sudo pyspark
```

```
ting port 4041.  
Welcome to  
 version 2.3.1  
  
Using Python version 2.7.10 (default, Oct 6 2017 22:29:07)  
SparkSession available as 'spark'.  
>>>
```

Storm

需要zookeeper 和python

1. 下載apache-storm的release版本 (<http://storm.apache.org/downloads.html>)

我們下載1.22版並把資料夾放到 /usr/local/storm 底下

一樣去改環境變數

```
$sudo vim /etc/profile
```

```
export STORM_HOME=/usr/local/storm
```

```
export PATH=$STORM_HOME/bin:$PATH
```

`$source /etc/profile` 讓它生效

2. 安裝zookeeper (<https://zookeeper.apache.org/releases.html#download>) 我們下載3.4.10

一樣下載，放到 /usr/local/zookeeper

把/usr/local/zookeeper/conf/zoo_sample.cfg 複製一份到 /usr/local/zookeeper/conf/zoo.cfg

一樣去改環境變數

```
$sudo vim /etc/profile
```

```
export ZOOKEEPER_HOME=/usr/local/zookeeper
```

```
export PATH=$PATH:$ZOOKEEPER_HOME/bin
```

`$source /etc/profile` 讓它生效

3. 後來查到OSX另外有dependency: zeromq

```
$brew install zeromq
```

4. 然後依序開啟 zookeeper,

```
$bin/zkServer.sh start
```

(用\$bin/zkServer.sh status 看有沒有開起來)

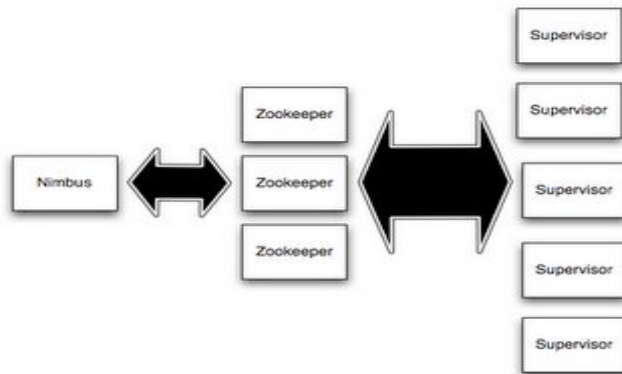

```
ucrwpa-1-7-10-25-27-11:storm-starter alumi5566$ zkServer.sh status
ZooKeeper JMX enabled by default
Using config: /usr/local/zookeeper/bin/../conf/zoo.cfg
Mode: standalone
```

5. 開啟storm的supervisor

\$bin/storm nimbus & (master node)

\$bin/storm supervisor & (slave node)

\$bin/storm ui &



(這三者的關係，spout和bolt是處理data flow的分工)

6. 成功的話用 <http://localhost:8080/index.html> 檢查



Storm UI

Cluster Summary

Nimbus Summary

Cluster Resources

Topology Summary

Supervisor Summary

Nimbus Configuration

1. 為了開發，安裝maven

\$brew install maven

用 **\$mvn -version** 檢查

```
ucrwpa-1-7-10-25-27-11:storm-starter alumi5566$ mvn -version
Apache Maven 3.5.3 (3383c37e1f9e9b3bc3df5050c29c8aff9f295297; 2018-02-24T1
Maven home: /usr/local/Cellar/maven/3.5.3/libexec
Java version: 10.0.1, vendor: Oracle Corporation
Java home: /Library/Java/JavaVirtualMachines/jdk-10.0.1.jdk/Contents/Home
Default locale: zh_TW_#Hant, platform encoding: Big5_Solaris
OS name: "mac os x", version: "10.13.4", arch: "x86_64", family: "mac"
ucrwpa-1-7-10-25-27-11:storm-starter alumi5566$
```

2. 去到 /usr/local/storm/examples/storm-starter

\$ mvn clean install -DskipTest

Fail, 在build的時候一直出現

Could not find artifact jdk.tools:jdk.tools:jar:1.7 at specified path mac

不知道為什麼maven一直去找jdk 10 的 tools.jar，這個jar在jdk 9之後消失了
最後在pom.xml裡面手動指定

```

<dependency>
<groupId>jdk.tools</groupId>
<artifactId>jdk.tools</artifactId>
<version>1.7</version>
<scope>system</scope>
<systemPath>/Library/Java/JavaVirtualMachines/jdk1.7.0_80.jdk/Contents/Home/lib/tools.jar</systemPath>
</dependency>

```

```

[INFO] Installing /usr/local/Cellar/storm/1.2.2/libexec/examples/storm-starter/dependencies.xml to /var/root/.m2/repository/org/apache/storm/storm-starter/1.2.2/storm-starter-1.2.2-libs.jar
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 01:44 min
[INFO] Finished at: 2018-06-10T23:23:34-07:00
[INFO] -----
ucrwpa-1-7-10-25-27-11:storm-starter alumi5566$

```

3. `$sudo mvn compile exec:java -Dstorm.topology=storm.starter.WordCountTopology`
還是失敗，改裝java 8就可以了

Storm的架構：data flow由spout和bolt組成

以word count為例：

核心组件包括：一个spout，两个bolt，一个Topology。

spout读文件，然后readLine，向bolt发射，一个文件处理完毕后，重命名，以不再重复处理。

第一个bolt将从spout接收到的字符串tuple按空格split，产生word，发射给下一个bolt。

第二个bolt接收到word后，统计、计数，放到HashMap容器中。

Useful Link

[1] Hadoop cmd

<http://hadoopspark.blogspot.com/2015/09/6-hadoop-hdfs.html>

[] Hadoop IO performance

<https://blog.csdn.net/bhq2010/article/details/8740154>

Spark 投影片

https://www.slideshare.net/imac-cloud/spark-61970801?next_slideshow=1

Zookeeper

<https://blog.csdn.net/liuxinghao/article/details/42747625>

[2] Storm wordcount 架構

<https://blog.csdn.net/wuliusir/article/details/49910873>

[] wordcount and other code

<https://github.com/storm-book>