VirtualBox+Ubuntu+Hadoop 單節點安裝步驟教學

1. 下載、安裝 VirtualBox windows，版本網址：https://www.virtualbox.org/wiki/Downloads
   檔案名稱：VirtualBox-6.0.12-133076-Win.exe

2. 下載 Ubuntu 網址：http://www.ubuntu-tw.org/modules/tinyd0/
   檔案名稱：ubuntu-16.04.6-desktop-amd64.iso
   ※選擇國內大學載點速度較快

3. 開啟 VirtualBox
   (1) 新增虛擬機
   (2) 啟動虛擬機

4. 登入 Ubuntu
   (1) 點選"裝置"，按下"插入 Guest Additions CD 映像"
   (2) 設定共用剪貼簿
   點選"裝置"，按下"共用剪貼簿"，按下"雙向"
   (3) 重新啟動

5. 安裝 JDK
   java -version
   sudo apt-get update
   sudo apt-get install default-jdk
   java -version
   update-alternatives --display java

6. 安裝、設定 SSH 無密碼登入
   安裝 ssh
   sudo apt-get install ssh
   安裝 rsync
   sudo apt-get install rsync
   產生 SSH key
   ssh-keygen -t dsa -P '' -f ~/.ssh/id_dsa
   查看 SSH key
   ll ~/.ssh
   將 SSH key 放到授權檔案中
   cat ~/.ssh/id_dsa.pub >> ~/.ssh/authorized_keys

   ※測試本機 ssh 是否正常
   ssh master // 若有詢問 Are you sure you want to continue connecting (yes/no)?回答 yes
   exit
   ssh master // 再連線一次！若沒有任何訊息或者要求密碼，而能自動連線，表示 ssh 功能正常
   exit
   若 ssh master 時一直詢問密碼，則改以 RSA 編碼
   ssh-keygen -t rsa -P '' -f ~/.ssh/id_rsa
   cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys

7. 下載、安裝 Hadoop
   到 Hadoop 官網下載頁面，複製檔案下載連結
   wget https://archive.apache.org/dist/hadoop/common/hadoop-2.6.4/hadoop-2.6.4.tar.gz

sudo tar -zxvf hadoop-2.6.4.tar.gz
sudo mv hadoop-2.6.4 /usr/local/hadoop
ll /usr/local/hadoop

8. 設定 Hadoop 環境變數
修改~/.bashrc
sudo gedit ~/.bashrc
輸入下列內容
export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
export HADOOP_HOME=/usr/local/hadoop
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_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/lib/native
export HADOOP_OPTS="-Djava.library.path=$HADOOP_HOME/lib"
export JAVA_LIBRARY_PATH=$HADOOP_HOME/lib/native:$JAVA_LIBRARY_PATH
讓~/.bashrc 修改生效
source ~/.bashrc
9. 修改 Hadoop 組態設定檔
修改 hadoop-env.sh
sudo gedit /usr/local/hadoop/etc/hadoop/hadoop-env.sh
輸入下列內容:
export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
修改 core-site.xml
sudo gedit /usr/local/hadoop/etc/hadoop/core-site.xml
在<configuration></configuration>之間，輸入下列內容:

```
<property>
  <name>fs.default.name</name>
  <value>hdfs://localhost:9000</value>
</property>
```

修改 yarn-site.xml
sudo gedit /usr/local/hadoop/etc/hadoop/yarn-site.xml
在<configuration></configuration>之間，輸入下列內容:

```
<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>
```

修改 mapred-site.xml
sudo cp /usr/local/hadoop/etc/hadoop/mapred-site.xml.template /usr/local/hadoop/etc/hadoop/mapred-site.xml
sudo gedit /usr/local/hadoop/etc/hadoop/mapred-site.xml
在<configuration></configuration>之間，輸入下列內容:

```
<property>
  <name>mapreduce.framework.name</name>
  <value>yarn</value>
```

```
    </property>
```
修改 hdfs-site.xml

sudo gedit /usr/local/hadoop/etc/hadoop/hdfs-site.xml

在<configuration></configuration>之間，輸入下列內容:

```
    <property>
      <name>dfs.replication</name>
      <value>3</value>
    </property>
    <property>
      <name>dfs.namenode.name.dir</name>
      <value> file:/usr/local/hadoop/hadoop_data/hdfs/namenode</value>
    </property>
    <property>
      <name>dfs.datanode.data.dir</name>
      <value> file:/usr/local/hadoop/hadoop_data/hdfs/datanode</value>
    </property>
```

10. 建立與格式化 HDFS 目錄

```
sudo mkdir -p /usr/local/hadoop/hadoop_data/hdfs/namenode
sudo mkdir -p /usr/local/hadoop/hadoop_data/hdfs/datanode
sudo chown hduser:hduser -R /usr/local/hadoop
hadoop namenode -format
```

11. 啟動 Hadoop

啟動 start-dfs.sh，再啟動 start-yarn.sh

```
start-dfs.sh
start-yarn.sh
```

或啟動全部

```
start-all.sh
```

12. 查看目前所執行的行程

```
jps
```

13. 驗證 Hadoop 是否安裝成功

(1) 開啟 Hadoop Resource Manager Web 介面

Hadoop Resource Manager Web 介面網址：http://localhost:8088/

(2) 開啟 NameNode HDFS Web 介面

開啟 HDFS Web UI 網址：http://localhost:50070/

執行 **WordCount.java** 範例程式

1. 在下列網址 Hadoop 說明文件中有 WordCount.java v1.0的程式碼：
   http://hadoop.apache.org/docs/current/hadoop-mapreduce-client/hadoop-mapreduce-client-core/MapReduceTutorial.html
2. 編輯 WordCount.java
   建立 wordcount 目錄
   ```
   mkdir -p ~/wordcount/input
   cd  ~/wordcount
   ```
   編輯 WordCount.java
   ```
   gedit WordCount.java
   ```

在 gedit 輸入 WordCount.java 完整程式碼

```
import java.io.IOException;
import java.util.StringTokenizer;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

public class WordCount {

 public static class TokenizerMapper
      extends Mapper<Object, Text, Text, IntWritable>{

   private final static IntWritable one = new IntWritable(1);
   private Text word = new Text();

   public void map(Object key, Text value, Context context
                   ) throws IOException, InterruptedException {
     StringTokenizer itr = new StringTokenizer(value.toString());
     while (itr.hasMoreTokens()) {
       word.set(itr.nextToken());
       context.write(word, one);
     }
   }
 }

 public static class IntSumReducer
      extends Reducer<Text,IntWritable,Text,IntWritable> {
   private IntWritable result = new IntWritable();

   public void reduce(Text key, Iterable<IntWritable> values,
                      Context context
                      ) throws IOException, InterruptedException {
     int sum = 0;
     for (IntWritable val : values) {
       sum += val.get();
     }
     result.set(sum);
     context.write(key, result);
   }
 }

 public static void main(String[] args) throws Exception {
   Configuration conf = new Configuration();
   Job job = Job.getInstance(conf, "word count");
   job.setJarByClass(WordCount.class);
   job.setMapperClass(TokenizerMapper.class);
   job.setCombinerClass(IntSumReducer.class);
   job.setReducerClass(IntSumReducer.class);
   job.setOutputKeyClass(Text.class);
   job.setOutputValueClass(IntWritable.class);
   FileInputFormat.addInputPath(job, new Path(args[0]));
   FileOutputFormat.setOutputPath(job, new Path(args[1]));
   System.exit(job.waitForCompletion(true) ? 0 : 1);
 }
}
```

3. 編譯 wordCount.java

sudo gedit ~/.bashrc

輸入下列內容

```
export PATH=${JAVA_HOME}/bin:${PATH}
export HADOOP_CLASSPATH=${JAVA_HOME}/lib/tools.jar
```

讓 ~/.bashrc 修改的設定值生效

```
source ~/.bashrc
```

開始編譯

```
hadoop com.sun.tools.javac.Main WordCount.java
```

jar cf wc.jar WordCount*.class
ll

4. 建立測試文字檔
cp /usr/local/hadoop/LICENSE.txt ~/wordcount/input
ll ~/wordcount/input
start-all.sh
hadoop fs -mkdir -p /user/hduser/wordcount/input
cd ~/wordcount/input
hadoop fs -copyFromLocal LICENSE.txt /user/hduser/wordcount/input
hadoop fs -ls /user/hduser/wordcount/input
5. 執行 wordCount.java
cd ~/wordcount
hadoop jar wc.jar WordCount /user/hduser/wordcount/input/LICENSE.txt /user/hduser/wordcount/output
6. 查看執行結果
hadoop fs -ls /user/hduser/wordcount/output
hadoop fs -cat /user/hduser/wordcount/output/part-r-00000
7. 再次執行前先刪除執行結果
hadoop fs -rm -R /user/hduser/wordcount/output