Hadoop安装流程

Hadoop-2.7.5

- Remote host：140.120.13.242
- master帳號　：master100XX
- slave帳號　　：slave200XX
- 密　　碼　　：hadoop
- ssh port　　：帳號後面的數字
- Manager WebUI port：300XX

Hadoop(1/)

- 修改每台主機(包含master和slave)的hosts文件，讓主機可以直接認得該host的IP。
  - ifconfig　　// 查看該主機IP

hostname

```
master@master:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 02:42:ac:11:00:0c
          inet addr:172.17.0.12  Bcast:0.0.0.0  Mask:255.255.0.0
          inet6 addr: fe80::42:acff:fe11:c/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:96 errors:0 dropped:0 overruns:0 frame:0
          TX packets:59 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:12738 (12.7 KB)  TX bytes:14012 (14.0 KB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
```

  - **sudo vim /etc/hosts　// 新增slave和master的hostname讓他們認得彼此的IP。(master和slave都要)**

```
127.0.0.1       localhost
::1     localhost ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
172.17.0.12     master master
172.17.0.13     slave slave
~
```

hostname

master的hosts file

```
127.0.0.1       localhost
::1     localhost ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
172.17.0.13     slave slave
172.17.0.12     master master
```

hostname

slave的hosts file

Hadoop(2/)

- 修改完hosts文件後可以互ping看看是否正常。　　// Ctrl+C停止

```
PING slave (172.17.0.13) 56(84) bytes of data.
64 bytes from slave (172.17.0.13): icmp_seq=1 ttl=64 time=0.088 ms
64 bytes from slave (172.17.0.13): icmp_seq=2 ttl=64 time=0.046 ms
64 bytes from slave (172.17.0.13): icmp_seq=3 ttl=64 time=0.055 ms
64 bytes from slave (172.17.0.13): icmp_seq=4 ttl=64 time=0.046 ms
```

- 可以互通之後，接著設定SSH免密碼登入，先安裝Openssh server。

  sudo apt-get install openssh-server (目前使用的container已事先安裝)

```
master@master:~$ sudo apt-get install openssh-server
Reading package lists... Done
Building dependency tree
Reading state information... Done
openssh-server is already the newest version (1:7.2p2-4ubuntu2.2).
0 upgraded, 0 newly installed, 0 to remove and 8 not upgraded.
```

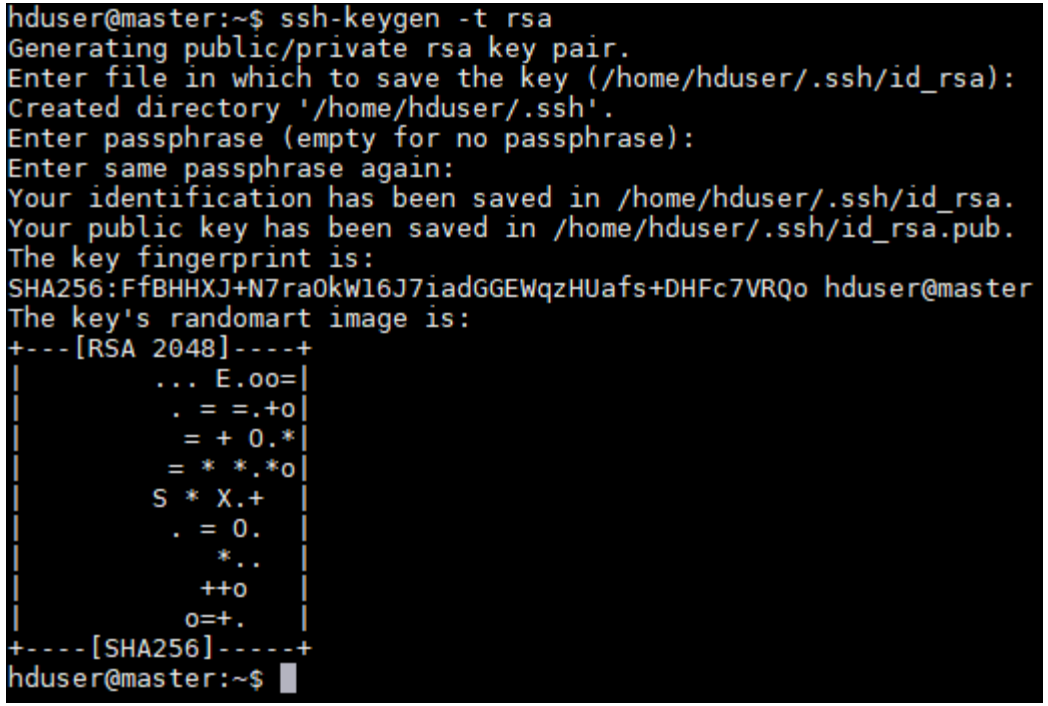
Hadoop(3/)

- **master和slave都要做**
- 新增hadoop用戶(這邊都用hduser方便後續進行)
  - sudo adduser hduser
- 給予 sudo 權限
  - sudo usermod -a -G sudo hduser
- 新增並加入hadoop 群組
  - sudo addgroup hadoop
  - sudo usermod -a -G hadoop hduser
- 切換此用戶
  - su - hduser
- groups　　// 確認是否有sudo權限及所屬群組

```
hduser@master:~$ groups
hduser sudo hadoop
hduser@master:~$
```

Hadoop(4/)

- 讓每台主機都產生公與私金鑰 **(master和slave都需要)**。
  - ssh-keygen -t rsa　　//過程一直按Enter直到結束，如下圖。



Hadoop(5/)

- 產生後在.ssh資料夾內會出現2個金鑰，將每台機器的公開金鑰都先傳給master
  由master集中所有機器的公開金鑰，在統一發送給所有機器。
  - cd ~/.ssh
  - ls

```
hduser@master:~$ cd ~/.ssh
hduser@master:~/.ssh$ ls
id_rsa  id_rsa.pub
```

- 將slave的公鑰傳送給master並在檔案尾端加上.slave1 **(在slave端下指令)**
  - scp ~/.ssh/id_rsa.pub hduser@master:~/.ssh/id_rsa.pub.slave1
- master會蒐集到所有的公鑰

```
hduser@master:~/.ssh$ ls
id_rsa  id_rsa.pub  id_rsa.pub.slave1
```

- master蒐集完成後將所有的公鑰統一集中到authorized_keys檔案中 **(在master端下指令)**
  - cat ~/.ssh/id_rsa.pub* >> ~/.ssh/authorized_keys
- 檢查authorized_keys是否有master與slave的key
  - cat ~/.ssh/authorized_keys
- 接著再將authorized_keys檔案**從master發送給所有slave**
  - scp ~/.ssh/authorized_keys hduser@slave:~/.ssh/

Hadoop(6/)

- 若成功則可以測試是否能夠無密碼切換不同主機，如下圖。
  - ssh master
  - ssh slave

```
hduser@master:~/.ssh$ ssh slave
Last login: Fri Dec 22 04:53:00 2017 from 172.17.0.12
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

hduser@slave:~$
```

  - exit　// 登出

```
hduser@slave:~$ exit
logout
Connection to slave closed.
hduser@master:~/.ssh$
```

Hadoop(7/)

- 接下來安裝java (目前使用的container已事先安裝)
  - sudo apt-get install -y default-jdk

```
hduser@master:~/.ssh$ sudo apt-get install -y default-jdk
[sudo] password for hduser:
Reading package lists... Done
Building dependency tree
Reading state information... Done
default-jdk is already the newest version (2:1.8-56ubuntu2).
0 upgraded, 0 newly installed, 0 to remove and 8 not upgraded.
hduser@master:~/.ssh$
```

- **在master主機**下載hadoop2.7.5版本
  - cd ~
  - sudo wget http://apache.stu.edu.tw/hadoop/common/hadoop-2.7.5/hadoop-2.7.5.tar.gz

```
hduser@master:~$ sudo wget http://apache.stu.edu.tw/hadoop/common/hadoop-2.7.5/hadoop-2.7.5.tar.gz
--2017-12-22 05:24:24--  http://apache.stu.edu.tw/hadoop/common/hadoop-2.7.5/hadoop-2.7.5.tar.gz
Resolving apache.stu.edu.tw (apache.stu.edu.tw)... 120.119.118.1, 2001:e10:c41:eeee::1
Connecting to apache.stu.edu.tw (apache.stu.edu.tw)|120.119.118.1|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 216929574 (207M) [application/x-gzip]
Saving to: 'hadoop-2.7.5.tar.gz'

hadoop-2.7.5.tar.gz                 100%[=============================================================================>] 206.88M  4.84MB/s    in 41s

2017-12-22 05:25:05 (5.10 MB/s) - 'hadoop-2.7.5.tar.gz' saved [216929574/216929574]

hduser@master:~$
```

Hadoop(8/)

- 下載完後解壓縮檔案
  - sudo tar -zxvf hadoop-2.7.5.tar.gz

```
hduser@master:~$ ls
hadoop-2.7.5  hadoop-2.7.5.tar.gz
hduser@master:~$
```

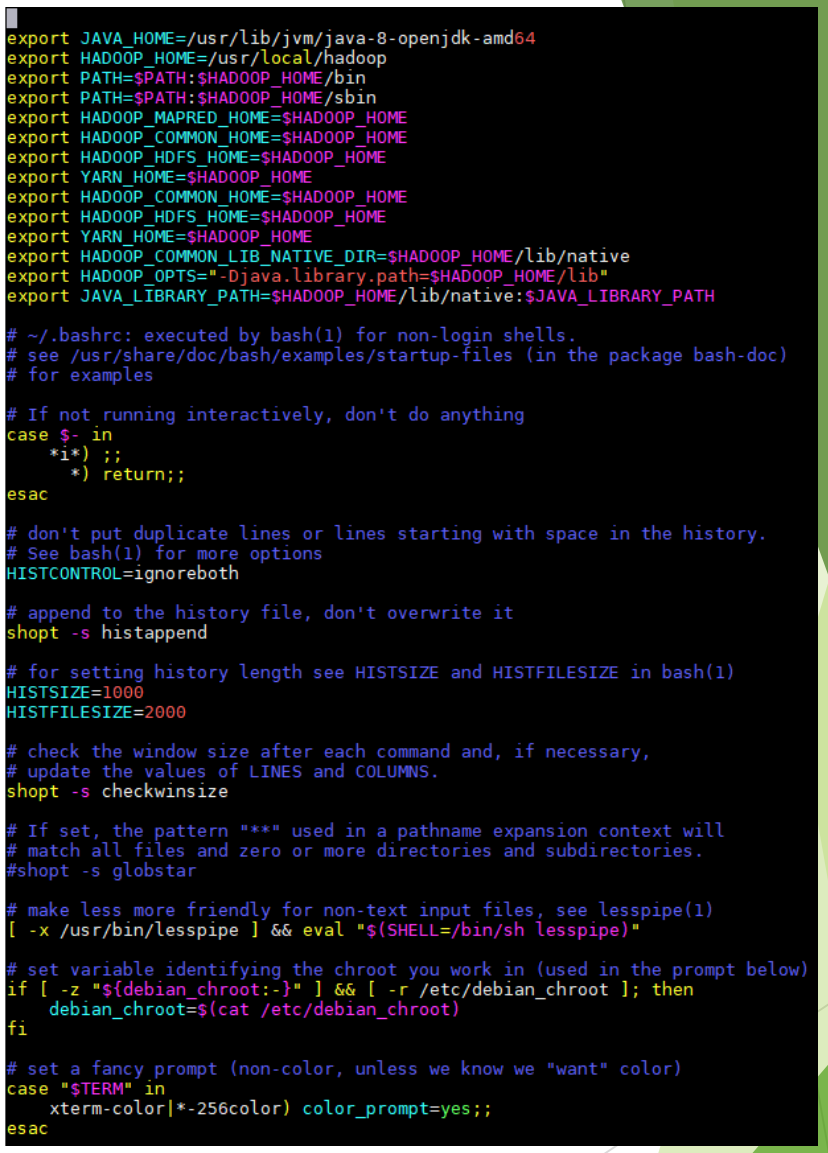
- 將hadoop-2.7.5移動到/usr/local/ 底下，並重新命名為hadoop。
  - sudo mv hadoop-2.7.5 /usr/local/hadoop
- 到/usr/local 底下查看hadoop資料夾是否正確移動
  - cd /usr/local/

```
hduser@master:/usr/local$ ls
bin  etc  games  hadoop  include  lib  man  sbin  share  src
hduser@master:/usr/local$
```

Hadoop(9/)

- 編輯 .bashrc file，方便我們打指令
  - cd ~
  - vim .bashrc
  - 增加以下環境變數設定

```
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_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
```



```
# ~/.bashrc: executed by bash(1) for non-login shells.
# see /usr/share/doc/bash/examples/startup-files (in the package bash-doc)
# for examples

# If not running interactively, don't do anything
case $- in
    *i*) ;;
      *) return;;
esac

# don't put duplicate lines or lines starting with space in the history.
# See bash(1) for more options
HISTCONTROL=ignoreboth

# append to the history file, don't overwrite it
shopt -s histappend

# for setting history length see HISTSIZE and HISTFILESIZE in bash(1)
HISTSIZE=1000
HISTFILESIZE=2000

# check the window size after each command and, if necessary,
# update the values of LINES and COLUMNS.
shopt -s checkwinsize

# If set, the pattern "**" used in a pathname expansion context will
# match all files and zero or more directories and subdirectories.
#shopt -s globstar

# make less more friendly for non-text input files, see lesspipe(1)
[ -x /usr/bin/lesspipe ] && eval "$(SHELL=/bin/sh lesspipe)"

# set variable identifying the chroot you work in (used in the prompt below)
if [ -z "${debian_chroot:-}" ] && [ -r /etc/debian_chroot ]; then
    debian_chroot=$(cat /etc/debian_chroot)
fi

# set a fancy prompt (non-color, unless we know we "want" color)
case "$TERM" in
    xterm-color|*-256color) color_prompt=yes;;
esac
```


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_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

Hadoop(10/)

- 使設定檔生效
  - source .bashrc
- 確認環境變數是否成功更改
  - hadoop

```
hduser@master:~$ hadoop
Usage: hadoop [--config confdir] [COMMAND | CLASSNAME]
  CLASSNAME            run the class named CLASSNAME
 or
  where COMMAND is one of:
  fs                   run a generic filesystem user client
  version              print the version
  jar <jar>            run a jar file
                       note: please use "yarn jar" to launch
                             YARN applications, not this command.
  checknative [-a|-h]  check native hadoop and compression libraries availability
  distcp <srcurl> <desturl> copy file or directories recursively
  archive -archiveName NAME -p <parent path> <src>* <dest> create a hadoop archive
  classpath            prints the class path needed to get the
  credential           interact with credential providers
                       Hadoop jar and the required libraries
  daemonlog            get/set the log level for each daemon
  trace                view and modify Hadoop tracing settings

Most commands print help when invoked w/o parameters.
hduser@master:~$
```

Hadoop(11/)

- 接下來的步驟都先在master完成，以下將要修改hadoop的文件，進入hadoop資料夾後需要設置的有七個檔案：hadoop-env.sh、yarn-env.sh，slaves、core-site.xml、hdfs-site.xml、maprd-site.xml、yarn-site.xml。
  - cd /usr/local/hadoop/etc/hadoop/

```
hduser@master:~$ cd /usr/local/hadoop/etc/hadoop/
hduser@master:/usr/local/hadoop/etc/hadoop$ ls
capacity-scheduler.xml  hadoop-env.cmd              hadoop-policy.xml       httpfs-signature.secret  kms-log4j.properties  mapred-env.sh              slaves                  yarn-env.sh
configuration.xsl       hadoop-env.sh               hdfs-site.xml           httpfs-site.xml          kms-site.xml          mapred-queues.xml.template ssl-client.xml.example  yarn-site.xml
container-executor.cfg  hadoop-metrics.properties   httpfs-env.sh           kms-acls.xml             log4j.properties      mapred-site.xml            ssl-server.xml.example
core-site.xml           hadoop-metrics2.properties  httpfs-log4j.properties kms-env.sh               mapred-env.cmd        mapred-site.xml.template   yarn-env.cmd
hduser@master:/usr/local/hadoop/etc/hadoop$
```

Hadoop(12/)

- hadoop-env.sh的修改
  - sudo vim hadoop-env.sh
  - export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64

```
# The java implementation to use.
export JAVA_HOME=${JAVA_HOME}

# 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}

export HADOOP_CONF_DIR=${HADOOP_CONF_DIR:-"/etc/hadoop"}

# Extra Java CLASSPATH elements.  Automatically insert capacity-scheduler.
for f in $HADOOP_HOME/contrib/capacity-scheduler/*.jar; do
  if [ "$HADOOP_CLASSPATH" ]; then
    export HADOOP_CLASSPATH=$HADOOP_CLASSPATH:$f
  else
    export HADOOP_CLASSPATH=$f
  fi
done
```

Hadoop(13/)

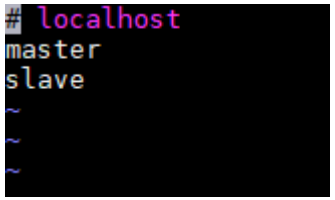
- yarn-env.sh的修改
  - sudo vim yarn-env.sh
  - export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64

```
# User for YARN daemons
export HADOOP_YARN_USER=${HADOOP_YARN_USER:-yarn}

# resolve links - $0 may be a softlink
export YARN_CONF_DIR="${YARN_CONF_DIR:-$HADOOP_YARN_HOME/conf}"

# some Java parameters
# export JAVA_HOME=/home/y/libexec/jdk1.6.0/
if [ "$JAVA_HOME" != "" ]; then
  #echo "run java in $JAVA_HOME"
  JAVA_HOME=$JAVA_HOME
fi

if [ "$JAVA_HOME" = "" ]; then
  echo "Error: JAVA_HOME is not set."
  exit 1
fi

JAVA=$JAVA_HOME/bin/java
JAVA_HEAP_MAX=-Xmx1000m

# For setting YARN specific HEAP sizes please use this
# Parameter and set appropriately
# YARN_HEAPSIZE=1000

# check envvars which might override default args
if [ "$YARN_HEAPSIZE" != "" ]; then
  JAVA_HEAP_MAX="-Xmx""$YARN_HEAPSIZE""m"
fi
```

修改這裡

Hadoop(14/)

- slaves的修改
- 這邊我們改成master和slave因為本次流程只有2台機器，我們讓他們都成為一個工作節點。
  - sudo vim slaves



Hadoop(15/)

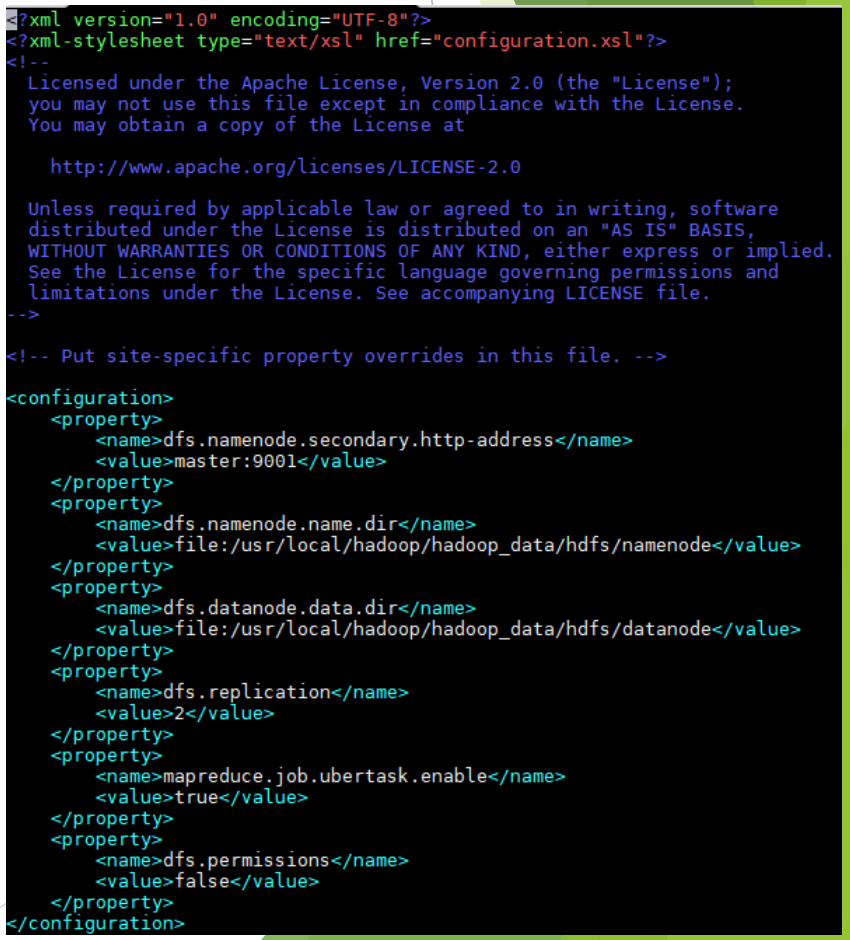
- core-site.xml的修改
  - sudo vim core-site.xml
  - 在configuration中加入property，切記路徑不要照抄，請查看自己的路徑。

```
<configuration>
    <property>
        <name>fs.defaultFS</name>
        <value>hdfs://master:9000/</value>
    </property>
    <property>
        <name>hadoop.tmp.dir</name>
        <value>file:/usr/local/hadoop/tmp</value>
    </property>
</configuration>
```

Hadoop(16/)

- hdfs-site.xml的修改
  - sudo vim hdfs-site.xml
  - 同樣在configuration中加入property，參數dfs.replication為資料備份的數量，因我們只有2個node，若設大於2，結果會出錯。

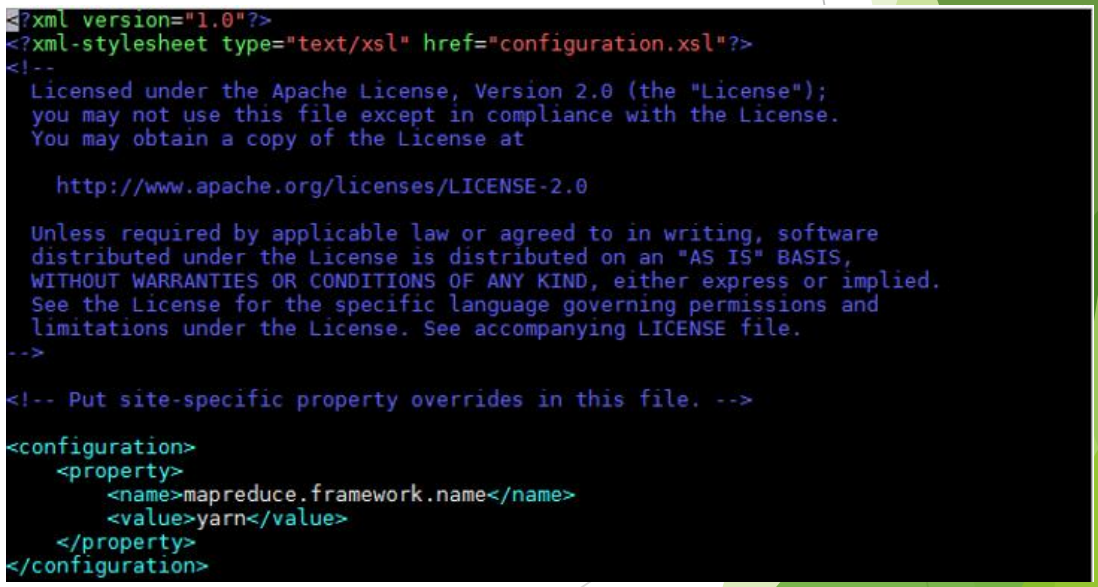
```
<configuration>
    <property>
        <name>dfs.namenode.secondary.http-address</name>
        <value>master:9001</value>
    </property>
    <property>
        <name>dfs.namenode.name.dir</name>
        <value>file:/usr/local/hadoop/hadoop_data/hdfs/namenode</value>
    </property>
    <property>
        <name>dfs.datanode.data.dir</name>
        <value>file:/usr/local/hadoop/hadoop_data/hdfs/datanode</value>
    </property>
    <property>
        <name>dfs.replication</name>
        <value>2</value>
    </property>
    <property>
        <name>mapreduce.job.ubertask.enable</name>
        <value>true</value>
    </property>
    <property>
        <name>dfs.permissions</name>
        <value>false</value>
    </property>
</configuration>
```



Hadoop(17/)

- mapred-site.xml的修改
- 若資料夾中沒有mapred-site.xml，請在資料夾中找到mapred-site.xml.template，並複製一份命名為mapred-site.xml來作修改。
  - sudo cp /usr/local/hadoop/etc/hadoop/mapred-site.xml.template /usr/local/hadoop/etc/hadoop/mapred-site.xml
  - sudo vim mapred-site.xml

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



Hadoop(18/)

- yarn-site.xml的修改
  - sudo vim yarn-site.xml

```
<configuration>
    <property>
        <name>yarn.nodemanager.resource.memory-mb</name>
        <value>2048</value>
    </property>
    <property>
        <name>yarn.nodemanager.resource.cpu-vcores</name>
        <value>2</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>
    <property>
        <name>yarn.resourcemanager.address</name>
        <value>master:8032</value>
    </property>
    <property>
        <name>yarn.resourcemanager.scheduler.address</name>
        <value>master:8030</value>
    </property>
    <property>
        <name>yarn.resourcemanager.resource-tracker.address</name>
        <value>master:8035</value>
    </property>
    <property>
        <name>yarn.resourcemanager.admin.address</name>
        <value>master:8033</value>
    </property>
    <property>
        <name>yarn.resourcemanager.webapp.address</name>
        <value>master:8088</value>
    </property>
</configuration>
```

Hadoop(19/)

- 以上步驟(Step7~Step18)在master端完成後，在slave端完成相同的修改後再繼續。

Hadoop(20/)

- 測試環境變數是否設定成功
  - hadoop

```
hduser@slave:~$ hadoop
Usage: hadoop [--config confdir] [COMMAND | CLASSNAME]
  CLASSNAME            run the class named CLASSNAME
 or
  where COMMAND is one of:
  fs                   run a generic filesystem user client
  version              print the version
  jar <jar>            run a jar file
                       note: please use "yarn jar" to launch
                             YARN applications, not this command.
  checknative [-a|-h]  check native hadoop and compression libraries availability
  distcp <srcurl> <desturl> copy file or directories recursively
  archive -archiveName NAME -p <parent path> <src>* <dest> create a hadoop archive
  classpath            prints the class path needed to get the
  credential           interact with credential providers
                       Hadoop jar and the required libraries
  daemonlog            get/set the log level for each daemon
  trace                view and modify Hadoop tracing settings

Most commands print help when invoked w/o parameters.
hduser@slave:~$
```

- 若出現hadoop: command not found

```
hduser@slave:~$ hadoop
-su: hadoop: command not found
hduser@slave:~$
```

- 則檢查hadoop資料夾是否在/usr/local底下以及是否記得運行Step9之步驟

Hadoop(21/)

- 都配置完成後到**master**上執行以下操作來啟動hadoop。
  - sudo mkdir -p /usr/local/hadoop/hadoop_data/hdfs/namenode
  - sudo mkdir -p /usr/local/hadoop/hadoop_data/hdfs/datanode
  - sudo chown hduser:sudo -R /usr/local/hadoop
  - hadoop namenode -format　　// 格式化namenode

```
17/12/22 06:21:53 INFO namenode.FSNamesystem: dfs.namenode.safemode.threshold-pct = 0.9990000128746033
17/12/22 06:21:53 INFO namenode.FSNamesystem: dfs.namenode.safemode.min.datanodes = 0
17/12/22 06:21:53 INFO namenode.FSNamesystem: dfs.namenode.safemode.extension     = 30000
17/12/22 06:21:53 INFO metrics.TopMetrics: NNTop conf: dfs.namenode.top.window.num.buckets = 10
17/12/22 06:21:53 INFO metrics.TopMetrics: NNTop conf: dfs.namenode.top.num.users = 10
17/12/22 06:21:53 INFO metrics.TopMetrics: NNTop conf: dfs.namenode.top.windows.minutes = 1,5,25
17/12/22 06:21:53 INFO namenode.FSNamesystem: Retry cache on namenode is enabled
17/12/22 06:21:53 INFO namenode.FSNamesystem: Retry cache will use 0.03 of total heap and retry cache entry expiry time is 600000 millis
17/12/22 06:21:53 INFO util.GSet: Computing capacity for map NameNodeRetryCache
17/12/22 06:21:53 INFO util.GSet: VM type       = 64-bit
17/12/22 06:21:53 INFO util.GSet: 0.029999999329447746% max memory 889 MB = 273.1 KB
17/12/22 06:21:53 INFO util.GSet: capacity      = 2^15 = 32768 entries
17/12/22 06:21:53 INFO namenode.FSImage: Allocated new BlockPoolId: BP-1912809608-172.17.0.12-1513923713353
17/12/22 06:21:53 INFO common.Storage: Storage directory /usr/local/hadoop/hadoop_data/hdfs/namenode has been successfully formatted.
17/12/22 06:21:53 INFO namenode.FSImageFormatProtobuf: Saving image file /usr/local/hadoop/hadoop_data/hdfs/namenode/current/fsimage.ckpt_0000000000000000000 using no compression
17/12/22 06:21:53 INFO namenode.FSImageFormatProtobuf: Image file /usr/local/hadoop/hadoop_data/hdfs/namenode/current/fsimage.ckpt_0000000000000000000 of size 323 bytes saved in 0 seconds.
17/12/22 06:21:53 INFO namenode.NNStorageRetentionManager: Going to retain 1 images with txid >= 0
17/12/22 06:21:53 INFO util.ExitUtil: Exiting with status 0
17/12/22 06:21:53 INFO namenode.NameNode: SHUTDOWN_MSG:
/************************************************************
SHUTDOWN_MSG: Shutting down NameNode at master/172.17.0.12
************************************************************/
```

Hadoop(22/)

- master 格式化後繼續下列指令啟動dfs和yarn。
  - start-all.sh

- 查看jps進程發現slave有的master都有，而master自己又額外多了一些進程，這是因為我們上面設定master除了負責管理以外自己也是一個slave。
  - jps

```
hduser@master:/usr/local/hadoop$ jps
1456 NodeManager
1072 SecondaryNameNode
849 DataNode
1893 Jps
695 NameNode
1325 ResourceManager
hduser@master:/usr/local/hadoop$
```

```
hduser@slave:/$ jps
240 DataNode
370 NodeManager
614 Jps
hduser@slave:/$
```

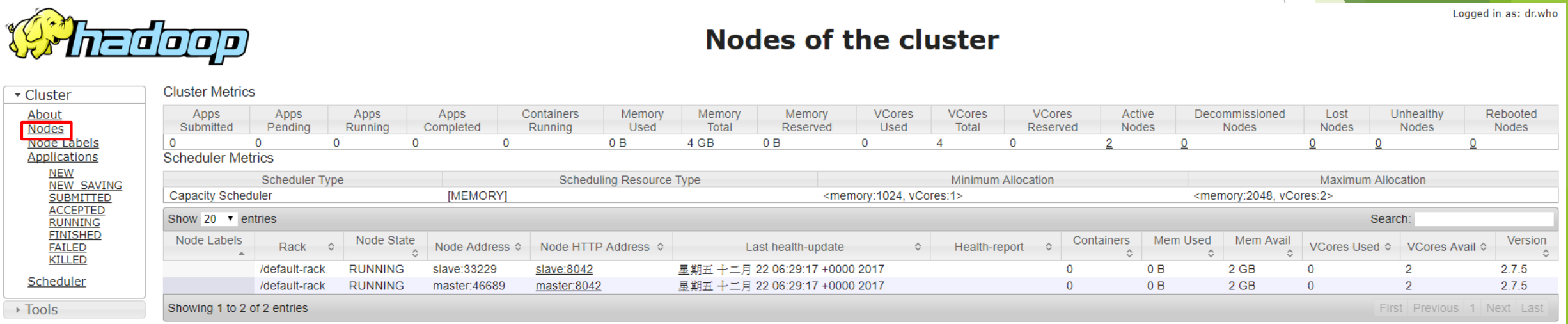
- 在hadoop上建立user目錄**(在master端)**
  - hadoop fs -mkdir -p /user/hduser

Hadoop(23/)

- 可以到瀏覽器上輸入http://140.120.13.242:**300XX**

  瀏覽Hadoop Resource Manager WebUI介面。

  - 可以看到2個node



Hadoop(24/)

- 若jps進程跟Hadoop(22/)的範例一模一樣的話，那大致上Hadoop已成功架設，若有不一樣的地方，你可以嘗試以下的方法。**(master和slave都要)**
  - stop-all.sh // 在master上關閉dfs和yarn
  - sudo rm -r /usr/local/hadoop/hadoop_data // 刪除hadoop_data資料夾
  - sudo rm -r /usr/local/hadoop/tmp // 刪除tmp資料夾
  - sudo rm -r /usr/local/hadoop/logs // 刪除logs資料夾
  - **並重新運行Step21~Step23之步驟**