Hadoop 2.2.0 installation

Chun-Chen Tu
timtu@umich.edu

Before installation

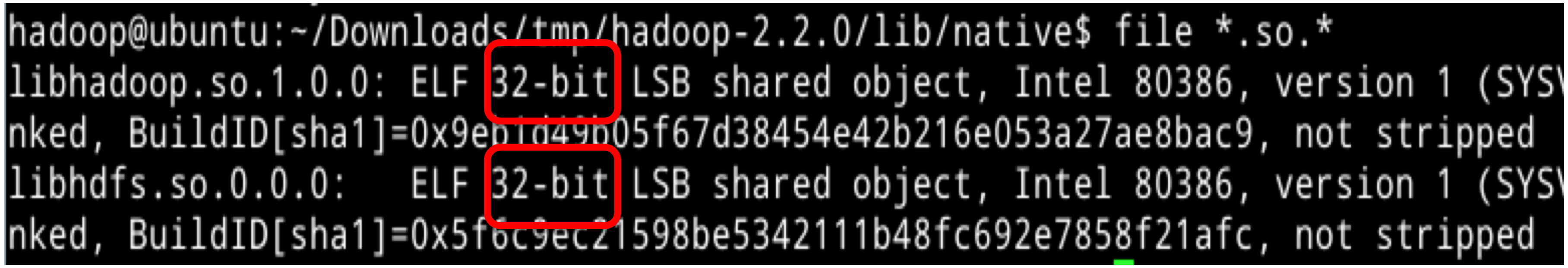
- Where to get hadoop 2.2.0
  - http://apache.stu.edu.tw/hadoop/common/hadoop-2.2.0/
  - ftp://hadoop:hahahadoop@140.113.114.104/hadoop_2.2.0_cluster.tar.gz
- GUI mode may help for typing commands.
- In this ppt, commands will be shown in italic and purple color
  - *mkdir hadoop*
- The content in file will be label as green in a square
  - Hello, Hadoop

About Hadoop

- If you download from official website, you should check the library files

  cd hadoop/lib/native

  *file **



```
hadoop@ubuntu:~/Downloads/tmp/hadoop-2.2.0/lib/native$ file *.so.*
libhadoop.so.1.0.0: ELF 32-bit LSB shared object, Intel 80386, version 1 (SYSV
nked, BuildID[sha1]=0x9eb1d49b05f67d38454e42b216e053a27ae8bac9, not stripped
libhdfs.so.0.0.0:   ELF 32-bit LSB shared object, Intel 80386, version 1 (SYSV
nked, BuildID[sha1]=0x5f6c9ec21598be5342111b48fc692e7858f21afc, not stripped
```

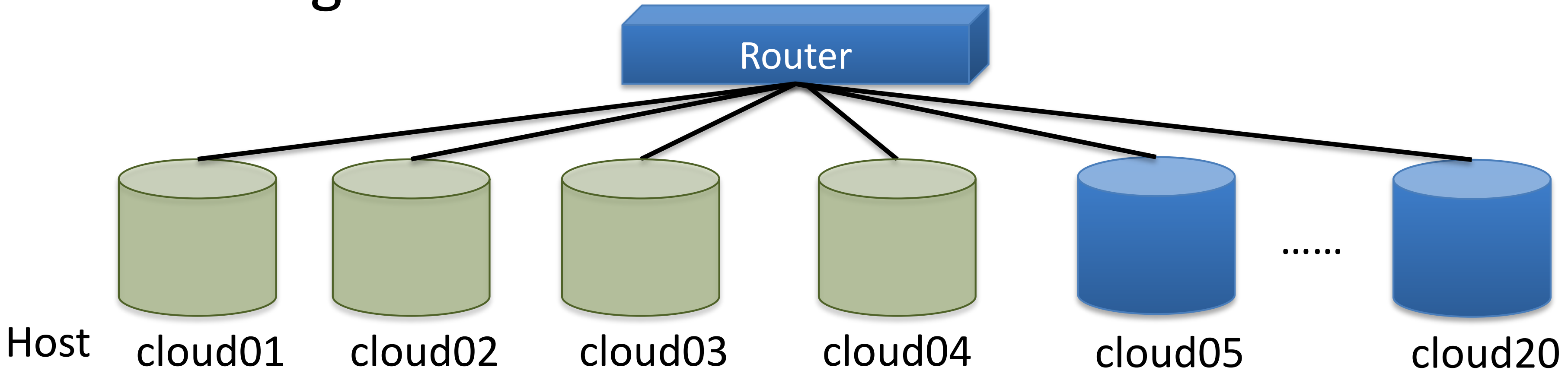
- The default library is designed for 32-bit system. There will be some warnings but Hadoop can still work. To fix it, re-compile Hadoop or get a 64-bit library.

Brief introduction

- Goal : setting up cluster of Hadoop 2.2.0
- Workflow:
  - Download and install required material.
  - Install JAVA
  - Install Hadoop
  - Cluster configuration
  - Start and test

Topology setting

- 1 machine running namenode
- 1 machine running secondary namenode
- 1 resource manager
- 1 client running job history server
- other 16 machines running datanode and node manager.



NOTE: You can make up host yourselves.

Hostname	IP	Role
cloud01	192.168.1.101	Namenode
cloud02	192.168.1.102	Secondary Namenode
cloud03	192.168.1.103	Resource manager
cloud04	192.168.1.104	Client, Job history server
cloud05	192.168.1.105	Datanode, Nodemanager
cloud20	192.168.1.120	

Notes

- We create a user called “hadoop” and we want to install hadoop under this user.
- Some settings should be exactly same for all machines. We will use shell scripts to make things easy.
- First, we will set up Namenode and copy the settings to other machines.
- Let’s start with cloud01 !!

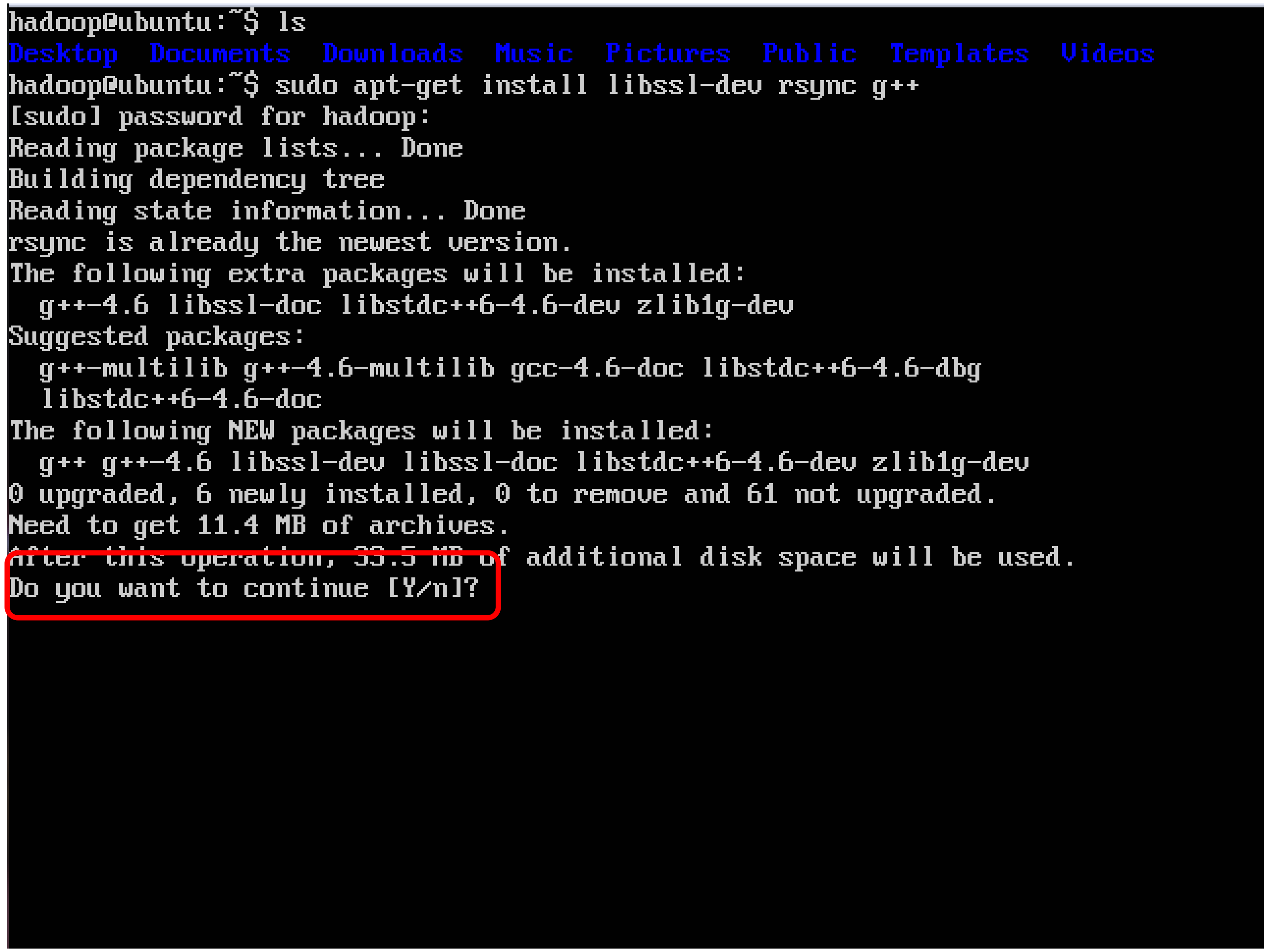
After login, install required packages first

sudo apt-get install libssl-dev rsync g++

type "*y*" when asked

Note: If you get message like "Package not found" type

sudo apt-get update



```
hadoop@ubuntu:~$ ls
Desktop  Documents  Downloads  Music  Pictures  Public  Templates  Videos
hadoop@ubuntu:~$ sudo apt-get install libssl-dev rsync g++
[sudo] password for hadoop:
Reading package lists... Done
Building dependency tree
Reading state information... Done
rsync is already the newest version.
The following extra packages will be installed:
  g++-4.6 libssl-doc libstdc++6-4.6-dev zlib1g-dev
Suggested packages:
  g++-multilib g++-4.6-multilib gcc-4.6-doc libstdc++6-4.6-dbg
  libstdc++6-4.6-doc
The following NEW packages will be installed:
  g++ g++-4.6 libssl-dev libssl-doc libstdc++6-4.6-dev zlib1g-dev
0 upgraded, 6 newly installed, 0 to remove and 61 not upgraded.
Need to get 11.4 MB of archives.
After this operation, 33.5 MB of additional disk space will be used.
Do you want to continue [Y/n]?
```

Edit hosts files

sudo vim /etc/hosts

Add the hostname and ip for each machines in the cluster

DELETE two lines of 127.x.x.x

Thus, it will look like this

```
192.168.1.101        cloud01
192.168.1.102        cloud02
192.168.1.103        cloud03
192.168.1.104        cloud04
192.168.1.105        cloud05
192.168.1.106        cloud06
192.168.1.107        cloud07
192.168.1.108        cloud08
192.168.1.109        cloud09
192.168.1.110        cloud10
192.168.1.111        cloud11
192.168.1.112        cloud12
192.168.1.113        cloud13
192.168.1.114        cloud14
192.168.1.115        cloud15
192.168.1.116        cloud16
192.168.1.117        cloud17
192.168.1.118        cloud18
192.168.1.119        cloud19
192.168.1.120        cloud20
```

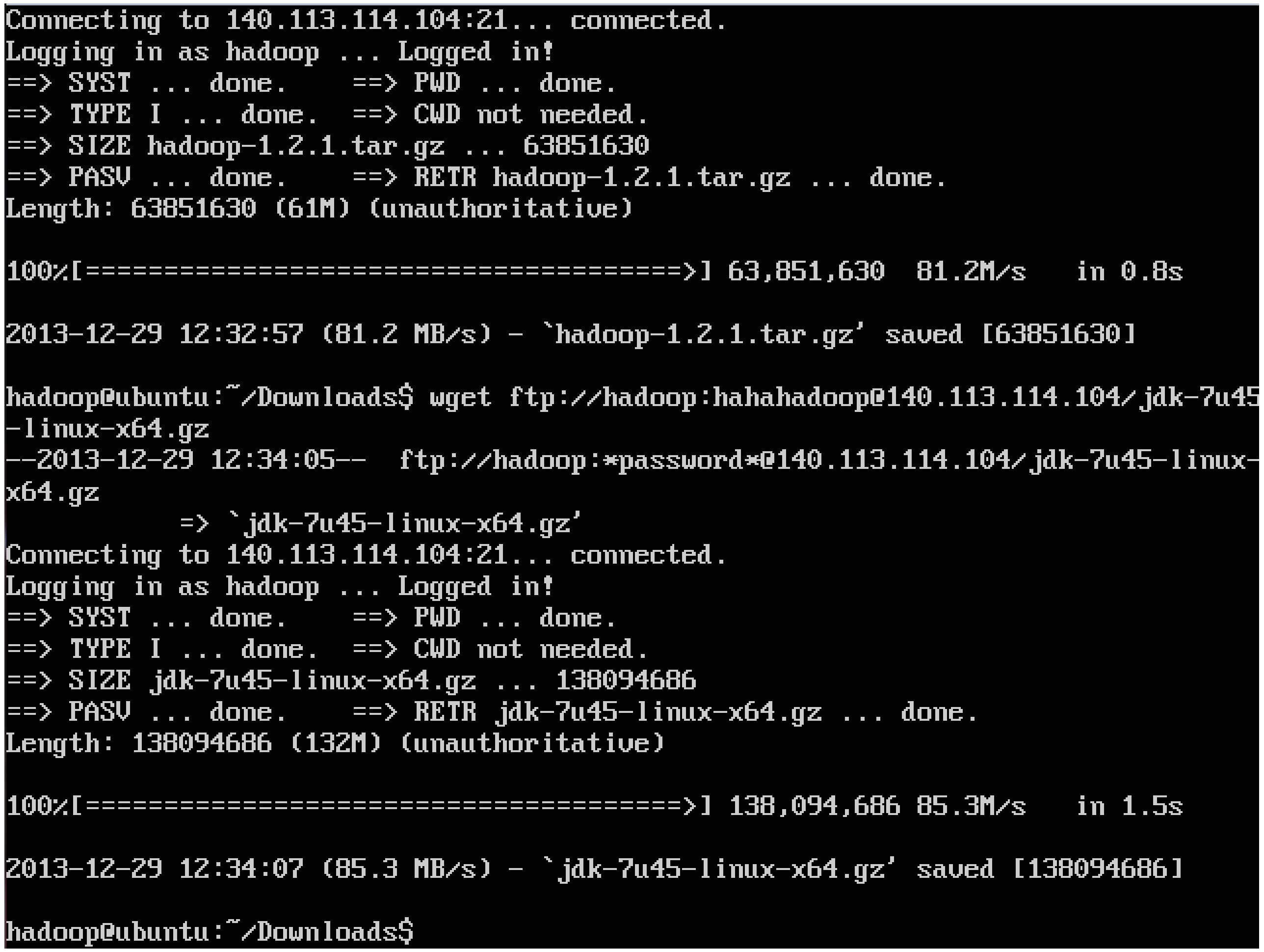
$\}$ You should add this information on all machines.

Download files:

cd ~/

wget ftp://hadoop:hahahadoop@140.113.114.104/hadoop_2.2.0_cluster.tar.gz

wget ftp://hadoop:hahahadoop@140.113.114.104/jdk-7u45-linux-x64.gz



```
Connecting to 140.113.114.104:21... connected.
Logging in as hadoop ... Logged in!
==> SYST ... done.    ==> PWD ... done.
==> TYPE I ... done.  ==> CWD not needed.
==> SIZE hadoop-1.2.1.tar.gz ... 63851630
==> PASV ... done.    ==> RETR hadoop-1.2.1.tar.gz ... done.
Length: 63851630 (61M) (unauthoritative)

100%[======================================>] 63,851,630  81.2M/s   in 0.8s

2013-12-29 12:32:57 (81.2 MB/s) - `hadoop-1.2.1.tar.gz' saved [63851630]

hadoop@ubuntu:~/Downloads$ wget ftp://hadoop:hahahadoop@140.113.114.104/jdk-7u45
-linux-x64.gz
--2013-12-29 12:34:05--  ftp://hadoop:*password*@140.113.114.104/jdk-7u45-linux-
x64.gz
           => `jdk-7u45-linux-x64.gz'
Connecting to 140.113.114.104:21... connected.
Logging in as hadoop ... Logged in!
==> SYST ... done.    ==> PWD ... done.
==> TYPE I ... done.  ==> CWD not needed.
==> SIZE jdk-7u45-linux-x64.gz ... 138094686
==> PASV ... done.    ==> RETR jdk-7u45-linux-x64.gz ... done.
Length: 138094686 (132M) (unauthoritative)

100%[======================================>] 138,094,686 85.3M/s   in 1.5s

2013-12-29 12:34:07 (85.3 MB/s) - `jdk-7u45-linux-x64.gz' saved [138094686]

hadoop@ubuntu:~/Downloads$
```

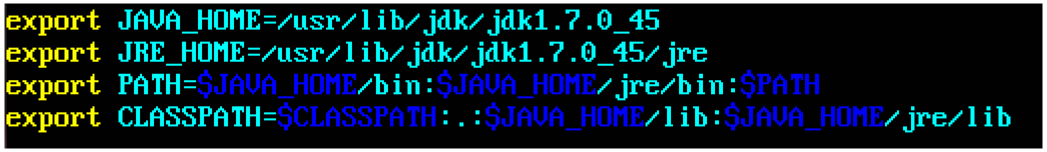
Install java : reference website
(Under Downloads folder)

tar -zxvf jdk-7u45-linux-x64.gz
sudo mkdir /usr/lib/jdk
sudo cp -r jdk1.7.0_45 /usr/lib/jdk/

Edit profile:

sudo vim ~/.bashrc

(add four lines in at the top of .bashrc)



```
export JAVA_HOME=/usr/lib/jdk/jdk1.7.0_45
export JRE_HOME=/usr/lib/jdk/jdk1.7.0_45/jre
export PATH=$JAVA_HOME/bin:$JAVA_HOME/jre/bin:$PATH
export CLASSPATH=$CLASSPATH:.:$JAVA_HOME/lib:$JAVA_HOME/jre/lib
```

```
export JAVA_HOME=/usr/lib/jdk/jdk1.7.0_45
export JRE_HOME=/usr/lib/jdk/jdk1.7.0_45/jre
export PATH=$JAVA_HOME/bin:$JAVA_HOME/jre/bin:$PATH
export CLASSPATH=$CLASSPATH:.:$JAVA_HOME/lib:$JAVA_HOME/jre/lib
```

source ~/.bashrc

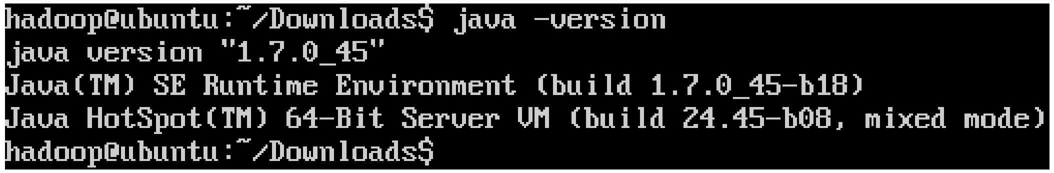
Config java:

sudo update-alternatives --install /usr/bin/java java /usr/lib/jdk/jdk1.7.0_45/bin/java 300
sudo update-alternatives --install /usr/bin/javac javac /usr/lib/jdk/jdk1.7.0_45/bin/javac 300
sudo update-alternatives --config java
sudo update-alternatives --config javac

Test it with version

java –version

You will see the version information if success.



```
hadoop@ubuntu:~/Downloads$ java -version
java version "1.7.0_45"
Java(TM) SE Runtime Environment (build 1.7.0_45-b18)
Java HotSpot(TM) 64-Bit Server VM (build 24.45-b08, mixed mode)
hadoop@ubuntu:~/Downloads$
```

SSH setting: SSH setting is optional but is recommended if you don't want to enter password every time.

Generate RSA key

ssh-keygen -t rsa -P '' -f ~/.ssh/id_rsa

put public key on current machine

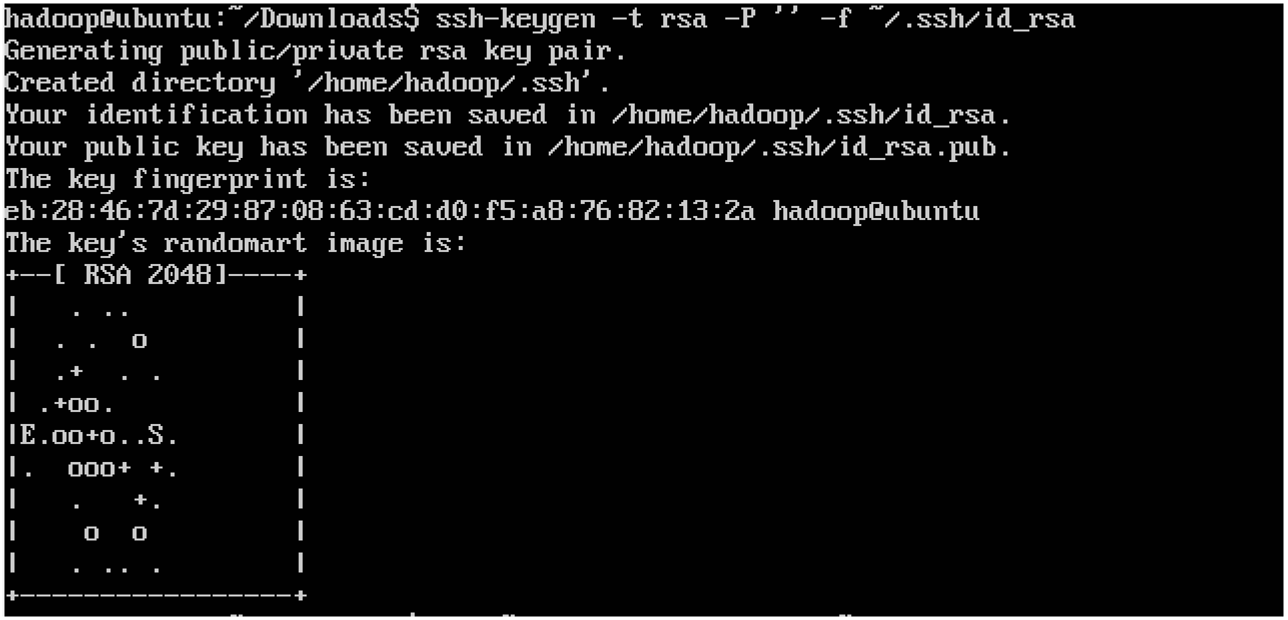
cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys

Copy public key to other machines

ssh hadoop@cloud02 "mkdir ~/.ssh"

scp ~/.ssh/id_rsa.pub hadoop@cloud02:~/.ssh/keys_from_hosts

ssh hadoop@cloud02 "cat ~/.ssh/keys_from_hosts >> ~/.ssh/authorized_keys"



```
hadoop@ubuntu:~/Downloads$ ssh-keygen -t rsa -P '' -f ~/.ssh/id_rsa
Generating public/private rsa key pair.
Created directory '/home/hadoop/.ssh'.
Your identification has been saved in /home/hadoop/.ssh/id_rsa.
Your public key has been saved in /home/hadoop/.ssh/id_rsa.pub.
The key fingerprint is:
eb:28:46:7d:29:87:08:63:cd:d0:f5:a8:76:82:13:2a hadoop@ubuntu
The key's randomart image is:
+--[ RSA 2048]----+
|    . ..         |
|   . .  o        |
|   .+  . .       |
|  .+oo.          |
|E.oo+o..S.       |
|.  ooo+ +.       |
|    .   +.       |
|     o  o        |
|    . .. .       |
+-----------------+
```

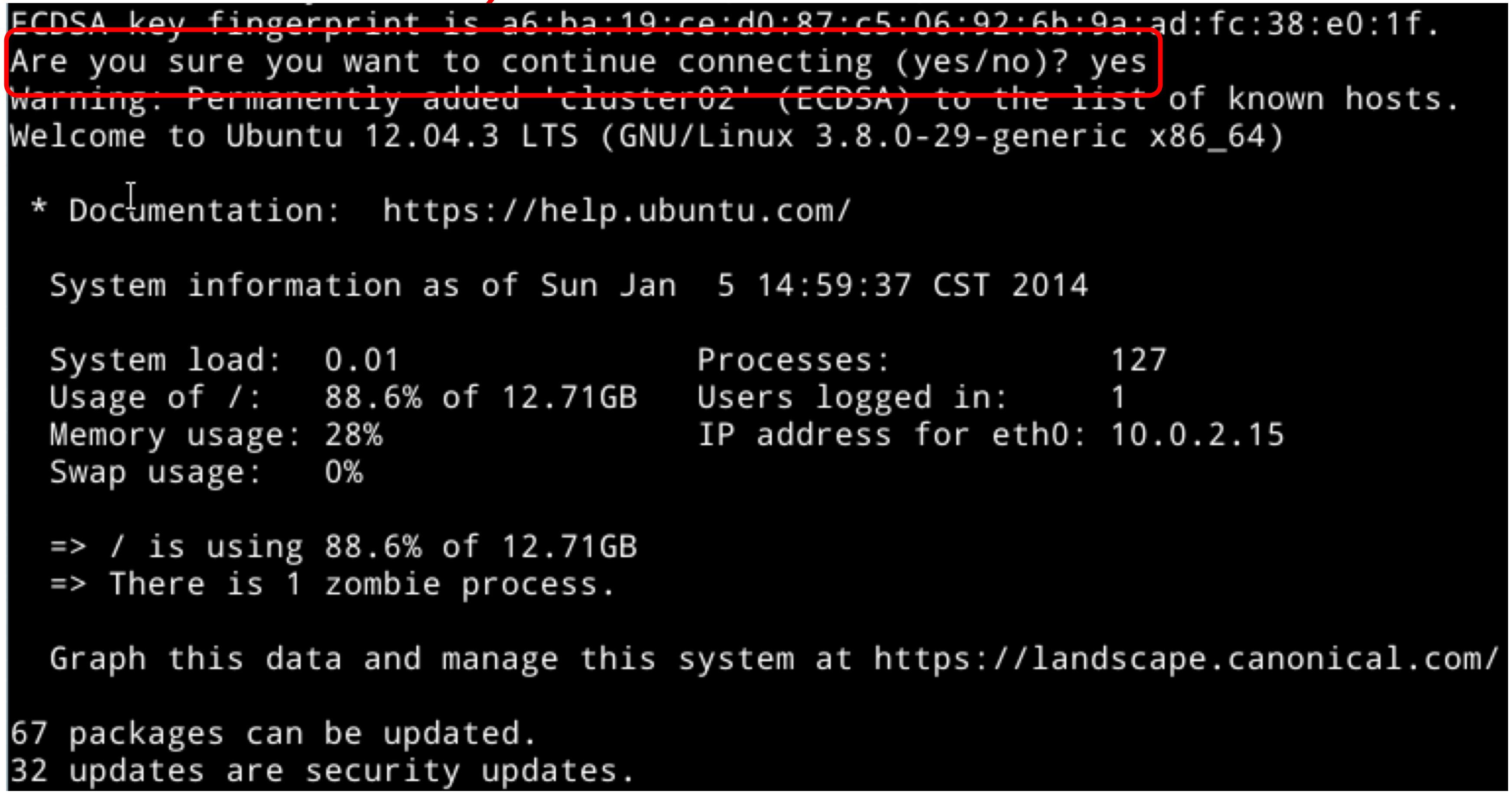
SSH test:

ssh hadoop@cloud02

remember to exit

exit

You will be asked for the authenticity for the first time. After this connection, no more inquiring.



```
ECDSA key fingerprint is a6:ba:19:ce:d0:87:c5:06:92:6b:9a:ad:fc:38:e0:1f.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'cluster02' (ECDSA) to the list of known hosts.
Welcome to Ubuntu 12.04.3 LTS (GNU/Linux 3.8.0-29-generic x86_64)

 * Documentation:  https://help.ubuntu.com/

  System information as of Sun Jan  5 14:59:37 CST 2014

  System load:  0.01              Processes:           127
  Usage of /:   88.6% of 12.71GB  Users logged in:     1
  Memory usage: 28%               IP address for eth0: 10.0.2.15
  Swap usage:   0%

  => / is using 88.6% of 12.71GB
  => There is 1 zombie process.

  Graph this data and manage this system at https://landscape.canonical.com/

67 packages can be updated.
32 updates are security updates.
```

If you fail the setting, you will need to enter password.

Set up ssh of all the machines in cluster using shell scripts

cd ~/hadoop/scripts (This directory will appear after tar hadoop_1.2.1_cluster.tar.gz See next slide)

1. List of machines

vim machines

```
cloud02
cloud03
cloud04
cloud05
cloud06
cloud07
cloud08
cloud09
cloud10
cloud11
cloud12
cloud13
cloud14
cloud15
cloud16
cloud17
cloud18
cloud19
cloud20
```

2. And then we create a shell script SetSSH.sh to do jobs according to the list of machines

vim SetSSH.sh

```
#!/bin/bash
HOST_FILES=/home/hadoop/hadoop/scriptes/machines
seq=1
while read line
do
        lines[$seq]=$line
        ((seq++))
done < $HOST_FILES

for ((i=1;i<=${#lines[@]};i++))
do
        echo "Set keys to ${lines[i]}"
        ssh ${lines[i]} "mkdir ~/.ssh"
        scp -r ~/.ssh/id_rsa.pub ${lines[i]}:~/.ssh/keys_from_hosts
        ssh ${lines[i]} "cat ~/.ssh/keys_from_hosts >> ~/.ssh/authorized_keys"
done
```

3. Finally change scripts to executable and run it

chmod 755 setSSH.sh

./setSSH.sh

Install hadoop:

```
tar -zxvf hadoop_2.2.0_cluster.tar.gz
mv hadoop ~/hadoop
```

move it under home directory for convenience

```
vim ~/.bashrc
```

```
export HADOOP_HOME=/home/hadoop/hadoop
export HADOOP_PREFIX=/home/hadoop/hadoop
export HADOOP_COMMON_HOME=/home/hadoop/hadoop
export HADOOP_MAPRED_HOME=/home/hadoop/hadoop
export HADOOP_CONF_DIR=/home/hadoop/hadoop/etc/hadoop
export HADOOP_HDFS_HOME=/home/hadoop/hadoop
export HADOOP_YARN_HOME=/home/hadoop/hadoop
export YARN_CONF_DIR=/home/hadoop/hadoop/etc/hadoop

export HADOOP_COMMON_LIB_NATIVE_DIR=/home/hadoop/hadoop/lib/native
export HADOOP_OPTS="-Djava.library.path=/home/hadoop/hadoop/lib/native"
```

```
source ~/.bashrc
```

Configure for cluster: you should set up 4 files in hadoop/etc/hadoop/
core-site.xml hdfs-site.xml mapred-site.xml yarn-site.xml

core-site.xml : parameter website

```
<configuration>
            <property>
                        <name>fs.defaultFS</name>
                         <value>hdfs://cloud01:9000</value>
            </property>
            <property>
                        <name>io.file.buffer.size</name>
                        <value>131072</value>
             </property>
             <property>
                         <name>hadoop.tmp.dir</name>
                         <value>/home/hadoop/tmp</value>
            </property>
            <property>
                          <name>hadoop.proxyuser.hadoop.hosts</name>
                          <value>*</value>
             </property>
            <property>
                           <name>hadoop.proxyuser.hadoop.groups</name>
                           <value>*</value>
            </property>
</configuration>
```

hdfs-site.xml : parameter website

```
<configuration>
        <property>
                <name>dfs.namenode.secondary.http-address</name>
                <value>cloud02:9001</value>
        </property>
        <property>
                <name>dfs.namenode.name.dir</name>
                <value>/home/hadoop/dfs/name</value>
        </property>
        <property>
                <name>dfs.datanode.data.dir</name>
                <value>/home/hadoop/dfs/data</value>
        </property>
        <property>
                <name>dfs.replication</name>
                <value>3</value>
        </property>
        <property>
                <name>dfs.webhdfs.enabled</name>
                <value>true</value>
        </property>
</configuration>
```

mapred-site.xml : parameter website

```
<configuration>
        <property>
                <name>mapreduce.framework.name</name>
                <value>yarn</value>
        </property>
        <property>
                <name>mapreduce.jobhistory.address</name>
                <value>cloud04:10020</value>
        </property>
        <property>
                <name>mapreduce.jobhistory.webapp.address</name>
                <value>cloud04:19888</value>
        </property>
</configuration>
```

yarn-site.xml : parameter website

```
<configuration>
		<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>cloud03:8032</value>
		</property>
		<property>
				<name>yarn.resourcemanager.scheduler.address</name>
				<value>cloud03:8030</value>
		</property>
		<property>
				<name>yarn.resourcemanager.resource-tracker.address</name>
				<value>cloud03:8031</value>
		 </property>
		<property>
				<name>yarn.resourcemanager.admin.address</name>
				<value>cloud03:8033</value>
		</property>
		<property>
				<name>yarn.resourcemanager.webapp.address</name>
				<value>cloud03:8088</value>
		</property>
 </configuration>
```

Similarly under hadoop/etc/hadoop/
edit slaves and masters

In general, masters contain namenode, secondary namenode, resourcemanager and job history server.

In our case, in masters we add

```
cloud01
cloud02
cloud03
cloud04
```

slaves will include datanode and nodemanager
in slaves we add

```
cloud05
cloud06
   ⋮
cloud20
```

Tar the current hadoop directory and copy files to other machine using shell scripts cpHadoop.sh

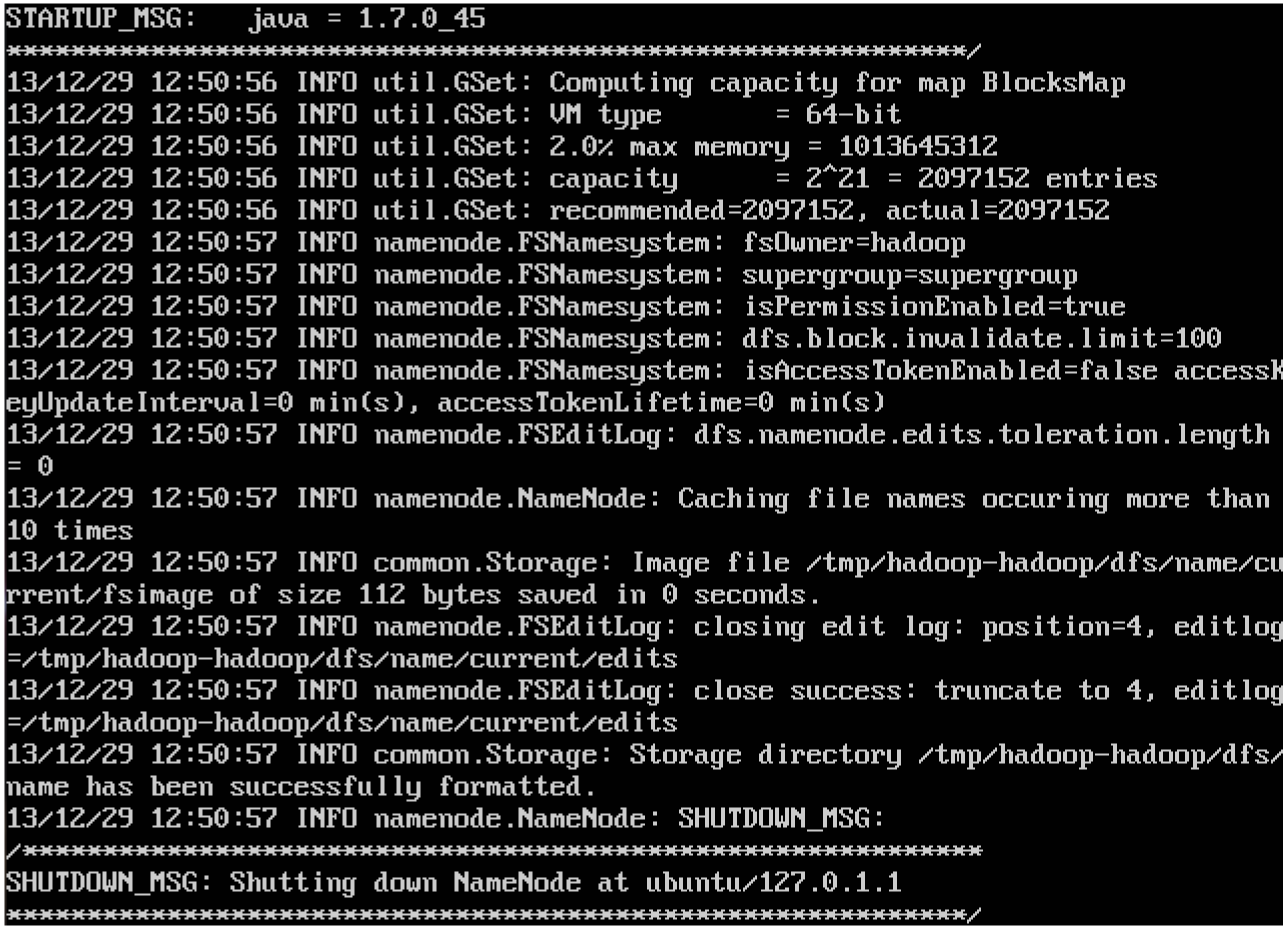
```
cd ~/
rm hadoop_2.2.0_cluster.tar.gz
tar –czvf hadoop_2.2.0_cluster.tar.gz hadoop
cd ~/hadoop/scripts/
```

```
dir=/home/hadoop/hadoop/etc/hadoop
HOST_FILES=/home/hadoop/hadoop/scripts/machines
seq=1
while read line
do
        lines[$seq]=$line
        ((seq++))
done < $HOST_FILES

for ((i=1;i<=${#lines[@]};i++))
do
       scp ~/Downloads/hadoop_2.2.0_cluster.tar.gz ${lines[$i]}:~/
       ssh ${lines[$i]} “tar –zxf hadoop_2.2.0_cluster.tar.gz”
       echo "Copy file: .bashrc to: ${lines[$i]}"
       scp ~/.bashrc ${lines[$i]}:~/.bashrc
for FILE in slaves masters core-site.xml hdfs-site.xml mapred-site.xml yarn-site.xml
do
        echo "Copy file: $FILE to: ${lines[$i]}"
        scp $dir/$FILE ${lines[$i]}:$dir/$FILE
done
done
```

HDFS format:

hdfs namenode -format



```
STARTUP_MSG:   java = 1.7.0_45
************************************************************/
13/12/29 12:50:56 INFO util.GSet: Computing capacity for map BlocksMap
13/12/29 12:50:56 INFO util.GSet: VM type       = 64-bit
13/12/29 12:50:56 INFO util.GSet: 2.0% max memory = 1013645312
13/12/29 12:50:56 INFO util.GSet: capacity      = 2^21 = 2097152 entries
13/12/29 12:50:56 INFO util.GSet: recommended=2097152, actual=2097152
13/12/29 12:50:57 INFO namenode.FSNamesystem: fsOwner=hadoop
13/12/29 12:50:57 INFO namenode.FSNamesystem: supergroup=supergroup
13/12/29 12:50:57 INFO namenode.FSNamesystem: isPermissionEnabled=true
13/12/29 12:50:57 INFO namenode.FSNamesystem: dfs.block.invalidate.limit=100
13/12/29 12:50:57 INFO namenode.FSNamesystem: isAccessTokenEnabled=false accessK
eyUpdateInterval=0 min(s), accessTokenLifetime=0 min(s)
13/12/29 12:50:57 INFO namenode.FSEditLog: dfs.namenode.edits.toleration.length
= 0
13/12/29 12:50:57 INFO namenode.NameNode: Caching file names occuring more than
10 times
13/12/29 12:50:57 INFO common.Storage: Image file /tmp/hadoop-hadoop/dfs/name/cu
rrent/fsimage of size 112 bytes saved in 0 seconds.
13/12/29 12:50:57 INFO namenode.FSEditLog: closing edit log: position=4, editlog
=/tmp/hadoop-hadoop/dfs/name/current/edits
13/12/29 12:50:57 INFO namenode.FSEditLog: close success: truncate to 4, editlog
=/tmp/hadoop-hadoop/dfs/name/current/edits
13/12/29 12:50:57 INFO common.Storage: Storage directory /tmp/hadoop-hadoop/dfs/
name has been successfully formatted.
13/12/29 12:50:57 INFO namenode.NameNode: SHUTDOWN_MSG:
/************************************************************
SHUTDOWN_MSG: Shutting down NameNode at ubuntu/127.0.1.1
************************************************************/
```

Start hadoop
cd ~/hadoop/sbin
./start-all.sh

jps : see what’s working on current machine.
hdfs dfsadmin –report : see the information of DFS.
more commands on this website

Start resource manager
ssh cloud03 “~/hadoop/sbin/yarn-start.sh start resourcemanager”
check if it start
ssh cloud03 “jps”

Start job history server
ssh cloud04 “~/hadoop/sbin/mr-jobhistory-daemon.sh start historyserver”
check if it start
ssh cloud04 “jps”

Let’s run an example! There is an example jar file under
hadoop/share/hadoop/mapreduce : hadoop-mapreduce-examples-2.2.0.jar

hadoop jar hadoop-mapreduce-examples-2.2.0.jar : to get more information

Now suppose we want to run the wordcount example.
First, put the input data on HDFS (you have to create your own input.txt first)
hdfs dfs –put input.txt /input.txt

Next, execute the wordcount example
hadoop jar hadoop-mapreduce-examples-2.2.0.jar wordcount /input.txt /test_out

Finally, get the results
hadoop dfs –get /test_out test_out

The result file part-r-00000 show up in the directory test_out