

## Notes on Hadoop ✓

Do not use Jdk, you must use sun Java in order to use hadoop

There are packages for sun Java on the ubuntu repository. you can download them directly using apt-get (Java 6)

Your JAVA\_HOME line in hadoop-env.sh should look like:  
export JAVA\_HOME=/usr/lib/jvm/java-6-sun

see:

[http://wiki.apache.org/hadoop/Running-Hadoop-On-Ubuntu-Linux-\(Single-Node-Cluster\)](http://wiki.apache.org/hadoop/Running-Hadoop-On-Ubuntu-Linux-(Single-Node-Cluster))  
(On node0.localhost)

Step 1... ✓

install sun Java 6 ...  
sudo apt-get install sun-java6-jdk

Step 2... ✓

added /usr/lib/jvm/java-6-sun-1.6.0.10 to  
/etc/jvm.cfg ✓  
(and /usr to bottom).

Step 3...

Add a dedicated Hadoop system user...

sudo addgroup hadoop ✓

sudo adduser --ingroup hadoop hadoop ✓

This will add user hadoop and the group hadoop to local machine. (give it your password)  
(add to sudo group)

• use "" for password -X

#### Step 4...

Configure SSH...

Setup an SSH server...

```
sudo apt-get install openssh-server ✓
```

```
su - hadoop ✓ (generate ssh key) (go to hadoop@node0)
```

```
ssh-keygen -t rsa -P "" (-P) ✓
```

Enable ssh access to your local machine with this newly created key...

```
cat $HOME/.ssh/id_rsa.pub >> $HOME/.ssh/authorized-keys (authorized_keys) ✓
```

#### Step 5... (from hadoop@node0 ~\$)

Test ssh setup by connecting to your local machine with hadoop user. The step is also needed to save your local machine's host key fingerprint to the hadoop user's known-hosts file. If you have special ssh configuration for your machine like non-standard ssh port

you can define host specific ssh options in `$HOME/.ssh/config` (see `man ssh-config` for more info).

```
ssh localhost ✓  
... (answer yes) ✓
```

```
... ✓
```

#### Step 6...

Disable IPv6...

To disable IPv6 on Ubuntu Linux open `/etc/modprobe.d/blacklist` and add the following lines to the end of the file...

```
# disable IPv6 ✓  
blacklist ipv6 ✓
```

### Step 7...

Hadoop Installation...

Download hadoop and unpack it in /usr/local  
cd /usr/local ✓

sudo tar xzf hadoop-0.19.0.tar.gz ✓

sudo mv hadoop-0.19.0 hadoop ✓

sudo chown -R hadoop:hadoop hadoop ✓

... Then I changed the owner of the files  
to hadoop user and group

### Step 8...

Configuration... Edit... /conf/hadoop-env.sh  
added export JAVA\_HOME=/usr/lib/jvm/java-6-sun  
to <HADOOP\_INSTALL>/conf/hadoop-env.sh ✓

Edit... conf/hadoop-site.xml

hadoop.tmp.dir variable must be changed to the  
directory of choice (/usr/local/hadoop-datastore/  
hadoop- $\{\text{\$user.name}\}$ ). Hadoop will expand  $\{\text{\$user.name}\}$   
to the system user which is running Hadoop, so  
in our case this will be hadoop and thus the  
final path will be /usr/local/hadoop-datastore/hadoop-hadoop.

Create hadoop-data store manually and then  
change ownership...

✓ sudo mkdir /usr/local/hadoop-datastore

✓ sudo chown hadoop:hadoop /usr/local/hadoop-datastore

### Step 9...

Format the name node

(hadoop@node0) ~~\$ /usr/local/hadoop namenode -format~~  
" /usr/local/hadoop/bin/hadoop namenode -format" ✓

## Step 10...

Start your Single-Node Cluster...

```
(hadoop@ubuntu: ~$) usr/local/hadoop  
usr/local/hadoop/bin/start-all.sh ✓
```

check expected processes...

```
usr/local/hadoop $ jps (Just jps) ✓
```

check if hadoop is listening on configured ports...  
sudo netstat -plten | grep java ✓

If you have errors check ~~usr/local~~

<HADOOP\_INSTALL>/logs directory...

Stop your Single-Node Cluster...

```
✓ hadoop@node0: ~$ usr/local/hadoop/bin/stop-all.sh
```

## Step 11... ✓

run tutorials...

(replace hadoop-0.19.0-examples.jar for  
hadoop-0.14.2-examples.jar in wordcount  
example.)

see links at end of tutorial on web.

The End... ✓✓✓✓

Commands...

```
usr/local/hadoop/bin/start-all.sh
```

```
usr/local/hadoop/bin/stop-all.sh
```

Running Hadoop On Ubuntu Linux (Multi-Node Cluster)  
[http://www.michael-noll.com/wiki/Running-Hadoop-On-Ubuntu-Linux-\(Multi-Node-Cluster\)](http://www.michael-noll.com/wiki/Running-Hadoop-On-Ubuntu-Linux-(Multi-Node-Cluster))

Tutorial approach and structure...

From two single-node clusters to a multi-node cluster -

Step 1... Install, configure and test a "local" Hadoop setup for each of the two ubuntu boxes and in the second step "merge" these two single-node clusters into one multi-node cluster in which one Ubuntu box will become the designated master (but also act as a slave with regards to data storage and processing) and the other box will become only a slave. It is much easier to track down any problems you might encounter due to the ~~lack of~~ reduced complexity of doing a single-node cluster first on each machine.

## Step 2...

### Networking...

Both machines must be able to reach each other over the network. The easiest is to put both machines in the same network with regards to hardware and software configuration, for example connect both machines via hub or switch and configure the network interface to use a common network such as  $192.168.0.x/24$

To make it simple we will assign the IP address  $192.168.0.1$  to node0 and  $192.168.0.2$  to node1. Update `/etc/hosts` on both machines with the following lines

```
# /etc/hosts (for node0 And node1)
192.168.0.1 node0 master (192.168.0.1 node0)
192.168.0.2 node1 slave (192.168.0.2 node1)
```

## Step 3...

### SSH Access

The hadoop user on node0 must be able to connect a. to its own user account on node0. ie `ssh node0` in this context and not necessarily `ssh localhost`. and b) to the hadoop user account on node1 (aka `hadoop@node1`).

If you followed this tutorial then just add the `hadoop@node0`'s public key (which should be `$HOME/.ssh/id_rsa.pub`) to the `authorized_keys` file of `hadoop@node1` (in this users `$HOME/.ssh/authorized_keys`).

The final step is to test the SSH setup by connecting with user `hadoop` from `node0` to the users account `hadoop` on `node1`. The step is also needed to save `node1`'s host key fingerprint to `hadoop@node0: ~/.ssh/known-hosts` file.

So connecting from `node0` to `node0`...  
`ssh node0.local` and `ssh node1.local` gives the desired result.

If using a cable/switch set Network Manager Applet → Edit Connections → Wired (Tab) → Auto eth0 → Edit → IPv4 Settings → Method to Link-Local Only.

Then `node0.local` to `node1.local`...

Then `node1.local` to `node0.local`...

all `ssh` using the `hadoop` account.

#### Step 4...

##### Hadoop...

Cluster Overview (aka the goal)...

The next section will describe how to configure one Ubuntu box as a master node and the other Ubuntu box as the slave node. The master node will also act as a slave because we only have two machines available in our cluster but still want to spread data and storage and processing to multiple machines.

The master node will run "master" daemons for each layer: `namenode` for the HDFS storage layer, and `jobtracker` for the ~~map~~ MapReduce processing layer. Both machines will run the "slave" daemons: `datanode` for the HDFS layer, and `tasktracker` for the MapReduce processing layer.

Basically, the "master" daemons are responsible for coordination and management of the "slave" daemons while the latter will do the actual data storage and data processing work.

### Configuration

conf/masters (master only)

conf/masters file defines the master nodes of our multi-node cluster. In our case, this is just the master machine.

- On master (node0) update /usr/local/hadoop/conf/masters to look like this...

master (node0.local) removed "localhost"

- conf/slaves (master only)

The conf/slaves file lists the hosts, one per line, where the Hadoop slave daemons (datanodes and tasktrackers) will run. We want both the master box (node0.local) and the slave box (node1.local) to act as Hadoop slaves because we want both of them to store and process data.

On master, update /usr/local/hadoop/conf/slaves to look like this

master (node0.local) (removed "localhost")  
slave (node1.local)

conf/hadoop-site.xml (all machines) ...

Assuming you have configured conf/hadoop-site.xml on each machine as described in single-node-cluster tutorial you will only have to change a few variables.

Important!: You must change `conf/hadoop-site.xml` on ALL machines as follows.

First we have to change the `fs.default.name` variable which specifies the NameNode (the HDFS master) host and port. In our case this is the master machine first...

changed `localhost` to `node0.local` for `fs.default.name`.

second....

changed `localhost` to `node0.local` for `mapred.job.tracker`.

third...

changed `dfs.replication` value to 2.

on both all machines

Step 5...-

Formatting the namenode

Before we start our new multi-node cluster we have to format Hadoop's distributed filesystem (HDFS) for the namenode. You need to do this the first time you setup a hadoop cluster

To format the filesystem which initializes the directory specified by the `dfs.name.dir` var. on the namenode run the command...

```
hadoop@node0: /usr/local/hadoop$ bin/hadoop namenode -format
```

Step 6...

Starting the multi-node cluster...

Starting the cluster is done in two steps. First the HDFS daemons are started: the namenode daemon is started on node0, and datanode daemons are started on all slaves (here node0.local and node1.local)

HDFS daemons

Run `/usr/local/hadoop/bin/start-dfs.sh` on the machine you want to run the namenode on. This will bring up the HDFS with the namenode running on the machine you ran the previous command on, and datanodes on the machines listed in `conf/slaves` file.

~~In our case we ~~start~~ run `bin/start-dfs.sh` on `node0.local`~~

On IBM T60 Thinkpad, ubuntu 8.10

Datanodes do not start unless `/usr/local/hadoop-datastore` is deleted, re-made & reformatted. I believe this is required everytime you add a node to the cluster. This problem is a known bug HADOOP-1212.

otherwise we can start the cluster by running...

```
/usr/local/hadoop/bin/start-all.sh  
jps (shows processes running)  
/usr/local/hadoop/bin/stop-all.sh
```

Thu Feb 10 2009

Successfully connected node0.local & node1.local in ad-hoc mode. Node1.local had to be connected twice to FI Cloud (my ad-hoc network) (FI)

Before starting hadoop, check for a ping both ways & check trace route to be sure they're on FI.

then start ~~the~~ the cluster, check processes with jps & shut them down

```
/usr/local/hadoop/bin/start-all.sh ...
```

```
jps ...
```

```
/usr/local/hadoop/bin/stop-all.sh
```

Settings <sup>(only)</sup> for Wireless FI Cloud (FI) adhoc network...

SSID: FI Cloud

Mode: Ad-Hoc

MTU: automatic

Wireless Security: ~~None~~ 128 bit passphrase

IPv4 Settings: Link-Local Only

... For all nodes... -

Issues...

```
/usr/local/hadoop/logs/hadoop-hadoop-datanode-node1.log  
is empty.
```

```
/usr/local/hadoop/logs/hadoop-hadoop-datanode-node0.log  
shows successful (and clean) startup &  
shutdown.
```

20 Feb, 2009

Starting 2 node cluster FI nodes 0 & 1

- Starting node 0 ✓ & log in attempted to connect to "Hidden Wireless Network" FI ✓
- start node 1 & log in ✓ attempted to connect to FI Cloud visible and labeled "available network" ✓

Check for network integrity visa-visa ping & trace route

First on node 0...

System → Admin → Net. Tools ✓  
set Net device to wlan0 ✓ then

ping node0.local ✓

ping node1.local ✓

use trace route to check path to node1 (had to reconnect to FI) (Try to ping at some time)

trace route shows only node 0 & 1 on network for both machines (as user hadoop)

Start Cluster...

`./usr/local/hadoop/bin/start-all.sh`

check processes on both nodes ✓ (jps)

Stop Cluster...

`./usr/local/hadoop/bin/stop-all.sh`

check that processes stopped on both nodes  
jps ✓

Note: System is unaffected with lid closed on slave with screen set to blank when it is closed

## Starting the Multi-Node Cluster

HDFS daemons.....

This is done in 2 steps

Step 1... HDFS daemons are started, namenode daemon is started on master, and datanode daemons are started on all slaves

Step 2... MapReduce daemons are started on master, and task tracker daemons are started on all slaves

## Starting the Multi-Node Cluster...

In practice.....

```
/usr/local/hadoop/bin/start-dfs.sh  
/usr/local/hadoop/bin/start-mapred.sh
```

## Stopping the multi-node cluster...

```
/usr/local/hadoop/bin/stop-mapred.sh  
/usr/local/hadoop/bin/stop-dfs.sh
```

4 March 2009

Hadoop Multi Node Cluster Log...

Today I will test the FI using WEP  
Wireless security...

... and text recommended from multinode  
cluster example by m. roll.

Step 1. download text files as us-ascii.

Step 2. copy them into /tmp/gutenberg

Step 3. copy data into HDFS...

```
usr/local/hadoop/bin/hadoop dfs -copyFromLocal  
/tmp/gutenberg gutenberg
```