

## MSDS610 Week 2 Install Hadoop on a Cluster Assignment - Nathan Worsham

I started this assignment by reading through first to see what would be required. Seeing that we would be building 4 very similar machines it seemed to make more sense to try to create one, do as many steps that would save time and was not unique to each machine, and then since this is a VM environment, clone the system and then edit the unique settings on each machine.

From what I can tell the non-unique settings would be:

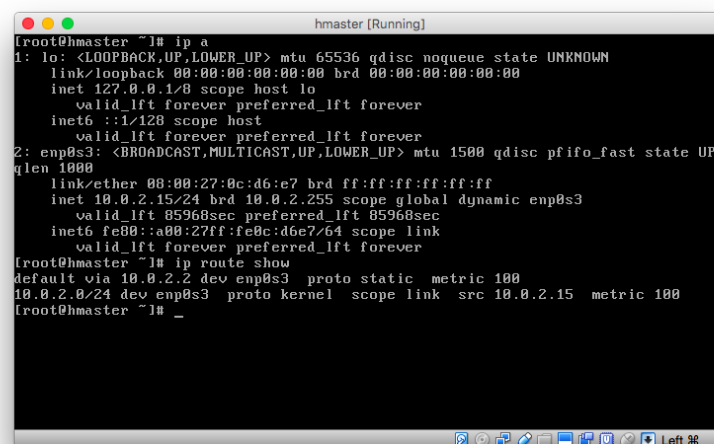
- create the VM
- update the VM
- create root and hadoop user during install
- if I can figure out ahead of time the IP scheme, the `/etc/hosts` file
- installing Java (by installing Java before coping the machine this also saves on having to run **alternatives** 4 times)
- environment variables in the `.bashrc` file
- installing hadoop
- making the datanode directory for hadoop
- hadoop config for namenode and datanode
- disable IPv6

And the unique settings would be:

- hostname
- ip address
- ssh key
- making the namenode directory for hmaster
- hadoop config for hmaster

So going with this plan I started by creating a single VM with very basic resources but going ahead and naming it my **hmaster** as I planned this machine to eventually be my master and the hostname would need to be changed on the other machines anyway. Using the option during install I went ahead and let the install create my hadoop user, did the reboot and logged in as root and then ran `yum update` to get the latest updates.

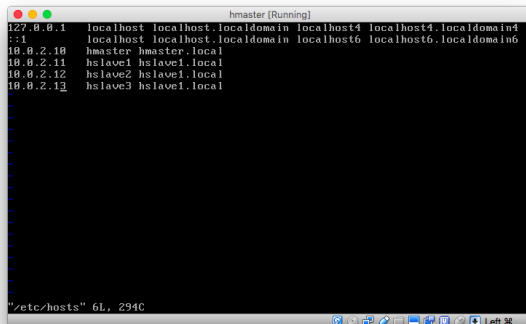
Next I decided to look at what IP address (10.0.2.15) and gateway (10.0.2.2) the machine was currently getting using the `ip a` (shortcut for `ip addr`) and `ip show route` commands.



```
root@hmaster ~]# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP
    qlen 1000
    link/ether 08:00:27:0c:d6:e7 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15/24 brd 10.0.2.255 scope global dynamic enp0s3
        valid_lft 85968sec preferred_lft 85968sec
    inet6 fe80::a00:27ff:fe0c:d6e7/64 scope link
        valid_lft forever preferred_lft forever
root@hmaster ~]# ip route show
default via 10.0.2.2 dev enp0s3 proto static metric 100
10.0.2.0/24 dev enp0s3 proto kernel scope link src 10.0.2.15 metric 100
root@hmaster ~]# _
```

Here I decided to setup an `/etc/hosts` file anyway because if I am wrong I would still need to touch the file on each host regardless. I was guessing that I likely was going to have to set a manual IP address on each host

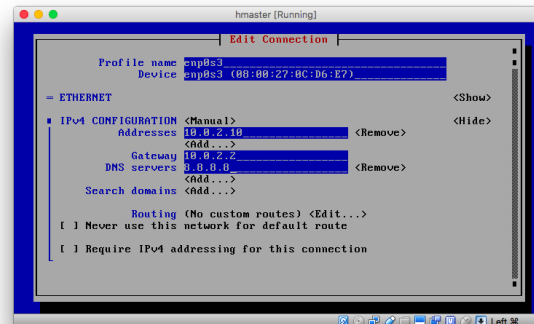
to get the `/etc/host` file working and stable. So I just picked a similar IP scheme and created my `/etc/hosts` file and then used `nmtui` to make the change to manual. Not knowing what the DNS was set to, I simply set it to my go-to-8.8.8.8, Google's DNS server address.



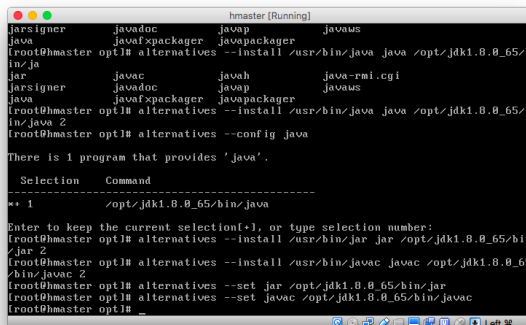
```

127.0.0.1 localhost localhost.localdomain localhost4 localhost4.localdomain4
::1 localhost localhost.localdomain localhost6 localhost6.localdomain6
10.0.2.10 hmaster hmaster.local
10.0.2.11 hslave1 hslave1.local
10.0.2.12 hslave2 hslave1.local
10.0.2.13 hslave3 hslave1.local

```



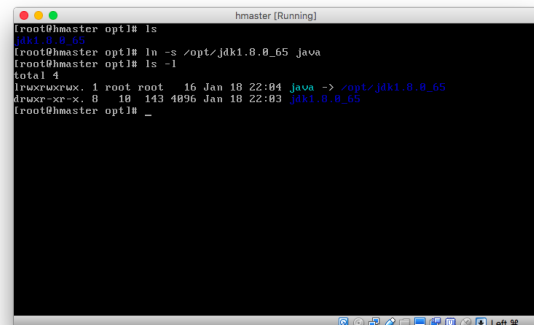
Trying to run the `wget` command, I realized I forgot to install `wget`. So I went ahead and installed it and then I was able to download Java and un-tar the package. Next I created symbolic links using the `alternatives` command. I also went ahead and created a link (using `ln -s` command) for `/opt/java` that redirects to the java directory. I am used to doing this sort of thing as it makes typos less likely and easier to remember.



```

jarsigner javadoc javap javaws
java javafxpackager javapackager
root@hmaster opt1# alternatives --install /usr/bin/java java /opt/jdk1.8.0_65/bin/java
root@hmaster opt1# alternatives --install /usr/bin/javac javac /opt/jdk1.8.0_65/bin/javac
root@hmaster opt1# alternatives --config java
There is 1 program that provides 'java'.
Selection Command
** 1 /opt/jdk1.8.0_65/bin/java
Enter to keep the current selection(+), or type selection number:
root@hmaster opt1# alternatives --install /usr/bin/jar jar /opt/jdk1.8.0_65/bin/jar
root@hmaster opt1# alternatives --install /usr/bin/javac javac /opt/jdk1.8.0_65/bin/javac
root@hmaster opt1# alternatives --set jar /opt/jdk1.8.0_65/bin/jar
root@hmaster opt1# alternatives --set javac /opt/jdk1.8.0_65/bin/javac
root@hmaster opt1#

```

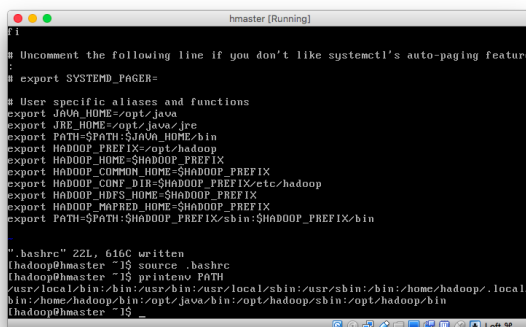


```

root@hmaster opt1# ls
jdk1.8.0_65
root@hmaster opt1# ln -s /opt/jdk1.8.0_65 java
root@hmaster opt1# ls -l
total 4
lrwxrwxrwx. 1 root root 16 Jan 18 22:04 java -> /opt/jdk1.8.0_65
drwxr-xr-x. 8 10 143 4096 Jan 18 22:03 jdk1.8.0_65
root@hmaster opt1#

```

By creating symbolic links, this made the environment variables slightly easier to type. I went ahead and downloaded and un-tar'd hadoop, again I made symbolic link to a simpler name.



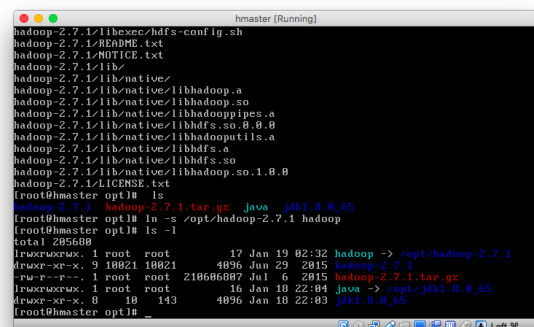
```

# Uncomment the following line if you don't like systemctl's auto-paging feature:
# export SYSTEMD_PAGER=

# User specific aliases and functions
export JAVA_HOME=/opt/java
export JRE_HOME=/opt/java/jre
export PATH=$PATH:$JAVA_HOME/bin
export HADOOP_PREFIX=/opt/hadoop
export HADOOP_HOME=$HADOOP_PREFIX
export HADOOP_COMMON_HOME=$HADOOP_PREFIX
export HADOOP_CONF_DIR=$HADOOP_PREFIX/etc/hadoop
export HADOOP_HDFS_HOME=$HADOOP_PREFIX
export HADOOP_MAPRED_HOME=$HADOOP_PREFIX
export PATH=$PATH:$HADOOP_PREFIX/sbin:$HADOOP_PREFIX/bin

".bashrc" 22L, 616C written
root@hmaster ~# source .bashrc
root@hmaster ~# printenv PATH
/usr/local/bin:/bin:/usr/bin:/usr/local/sbin:/usr/sbin:/bin:/home/hadoop/.local/bin:/home/hadoop/bin:/opt/java/bin:/opt/hadoop/sbin:/opt/hadoop/bin
root@hmaster ~#

```



```

hadoop-2.7.1/libexec/hdfs-config.sh
hadoop-2.7.1/README.txt
hadoop-2.7.1/NOTICE.txt
hadoop-2.7.1/lib/
hadoop-2.7.1/lib/native/
hadoop-2.7.1/lib/native/libhadoop.a
hadoop-2.7.1/lib/native/libhadoop.so
hadoop-2.7.1/lib/native/libhadooppipes.a
hadoop-2.7.1/lib/native/libhdfs.so.0.0.0
hadoop-2.7.1/lib/native/libhadooputils.a
hadoop-2.7.1/lib/native/libhdfs.a
hadoop-2.7.1/lib/native/libhdfs.so
hadoop-2.7.1/lib/native/libhadoop.so.1.0.0
hadoop-2.7.1/LICENCE.txt
root@hmaster opt1# ls
hadoop-2.7.1 hadoop-2.7.1.tar.gz java jdk1.8.0_65
root@hmaster opt1# ln -s /opt/hadoop-2.7.1 hadoop
root@hmaster opt1# ls -l
total 28568
lrwxrwxrwx. 1 root root 17 Jan 19 02:32 hadoop -> /opt/hadoop-2.7.1
drwxr-xr-x. 9 10021 10021 4096 Jun 29 2015 hadoop-2.7.1
-rw-r--r--. 1 root root 218686807 Jul 6 2015 hadoop-2.7.1.tar.gz
lrwxrwxrwx. 1 root root 16 Jan 18 22:04 java -> /opt/jdk1.8.0_65
drwxr-xr-x. 8 10 143 4096 Jan 18 22:03 jdk1.8.0_65
root@hmaster opt1#

```

I created the datanode folder and the tmp directory recommended from the class discussions in the hadoop's home directory. I then corrected the ownerships for those and the hadoop folder in `/opt`. I went ahead and made the same change to the symbolic link permissions as well.

```
hmaster [Running]
[root@hmaster home]# mkdir /home/hadoop/datanode
[root@hmaster home]# chown hadoop.hadoop /home/hadoop/datanode/
[root@hmaster home]# mkdir /home/hadoop/tmp
[root@hmaster home]# chown hadoop.hadoop /home/hadoop/tmp/
[root@hmaster home]# ls -l hadoop/
total 8
drwxr-xr-x. 2 hadoop hadoop 6 Jan 19 02:49 datanode
drwxr-xr-x. 2 hadoop hadoop 6 Jan 19 02:49 tmp
[root@hmaster home]# _
```

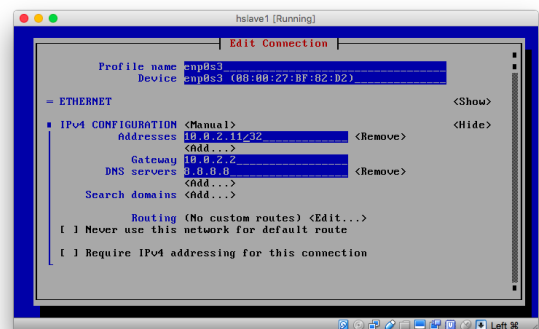
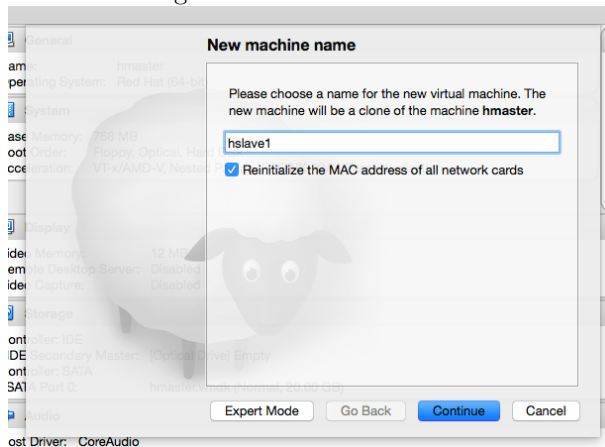
```
hmaster [Running]
drwxr-xr-x. 2 hadoop hadoop 6 Jan 19 02:38 datanode
[root@hmaster home]# ls -l /opt/
total 285688
lrwxrwxrwx. 1 root root 17 Jan 19 02:32 hadoop -> /opt/hadoop-2.7.1
drwxr-xr-x. 9 10021 10021 4096 Jun 29 2015 hadoop-2.7.1
-rw-r--r--. 1 root root 210606887 Jul 6 2015 hadoop-2.7.1.tar.gz
lrwxrwxrwx. 1 root root 16 Jan 18 22:04 java -> /opt/jdk1.8.0_65
drwxr-xr-x. 8 10 143 4096 Jan 18 22:03 jdk1.8.0_65
[root@hmaster home]# chown -R hadoop.hadoop /opt/hadoop
[root@hmaster home]# ls -l /opt/
total 285688
lrwxrwxrwx. 1 hadoop hadoop 17 Jan 19 02:32 hadoop -> /opt/hadoop-2.7.1
drwxr-xr-x. 9 hadoop hadoop 4096 Jun 29 2015 hadoop-2.7.1
-rw-r--r--. 1 root root 210606887 Jul 6 2015 hadoop-2.7.1.tar.gz
lrwxrwxrwx. 1 root root 16 Jan 18 22:04 java -> /opt/jdk1.8.0_65
drwxr-xr-x. 8 10 143 4096 Jan 18 22:03 jdk1.8.0_65
[root@hmaster home]# chown -R hadoop.hadoop /opt/hadoop-2.7.1
[root@hmaster home]# ls -l /opt/
total 285688
lrwxrwxrwx. 1 hadoop hadoop 17 Jan 19 02:32 hadoop -> /opt/hadoop-2.7.1
drwxr-xr-x. 9 hadoop hadoop 4096 Jun 29 2015 hadoop-2.7.1
-rw-r--r--. 1 root root 210606887 Jul 6 2015 hadoop-2.7.1.tar.gz
lrwxrwxrwx. 1 root root 16 Jan 18 22:04 java -> /opt/jdk1.8.0_65
drwxr-xr-x. 8 10 143 4096 Jan 18 22:03 jdk1.8.0_65
[root@hmaster home]# _
```

Next I created the configs that would need to be on all nodes of the cluster—core-site.xml and hdfs-site.xml.

```
hmaster [Running]
http://www.apache.org/licenses/LICENSE-2.0
Unless required by applicable law or agreed to in writing, software
distributed under the license is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the license for the specific language governing permissions and
limitations under the license. See accompanying LICENSE file.
-->
<!-- Put site-specific property overrides in this file. -->
<configuration>
<property>
<name>dfs.defaultFS</name>
<value>hdfs://hmaster:9000</value>
</property>
<property>
<name>hadoop.tmp.dir</name>
<value>/home/hadoop/tmp</value>
</property>
</configuration>
"/opt/hadoop-2.7.1/etc/hadoop/core-site.xml" 29L, 955C written
[root@hmaster home]# _
```

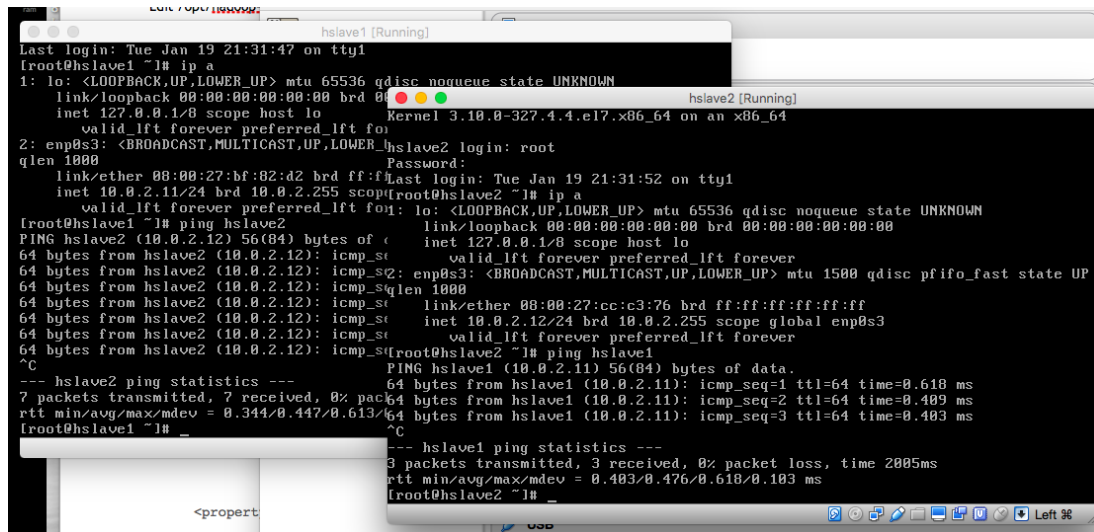
```
hmaster [Running]
Unless required by applicable law or agreed to in writing, software
distributed under the license is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the license for the specific language governing permissions and
limitations under the license. See accompanying LICENSE file.
-->
<!-- Put site-specific property overrides in this file. -->
<configuration>
<property>
<name>dfs.replication</name>
<value>3</value>
</property>
<property>
<name>dfs.permissions</name>
<value>false</value>
</property>
<property>
<name>dfs.datanode.data.dir</name>
<value>/home/hadoop/datanode</value>
</property>
</configuration>
"/opt/hadoop-2.7.1/etc/hadoop/hdfs-site.xml" 32L, 1023C written
[root@hmaster home]# _
```

Last, I disabled IPv6 by editing /etc/sysctl.conf. I also confirmed that firewalld was still not installed—something I learned from last week. Now I was ready to clone my VM, so I powered off the VM and from right-click menu on the VM I chose to clone. I labeled the new machine "hslave1" appropriately and then let it do its thing. After this was complete I started up the machine. I decided it would be best to leave the hmaster off until all 3 slaves can be brought up to change their IP addresses so there would not be a conflict. I then rebooted so that the changes would take place and I would remember that I was working on the slave not the master.



I was now ready to repeat the same steps for the remaining slaves, however I decided after building another slave to see if the two slaves could talk with each other. I tried pinging the hslave1 from hslave2 and received no response. Remembering that I was set to NAT networking mode on both VMs, I shut them both down and changed to "Internal Network" mode on both and started both machines. Even after this the VMs could still not ping each other, I researched a little on the internet to make sure I had chosen the right thing but then I realized I made a mistake when I set the IP address of each machine. Each machine was set to a subnet mask of /32, which is roughly the equivalent of one-hand clapping or a network of one.

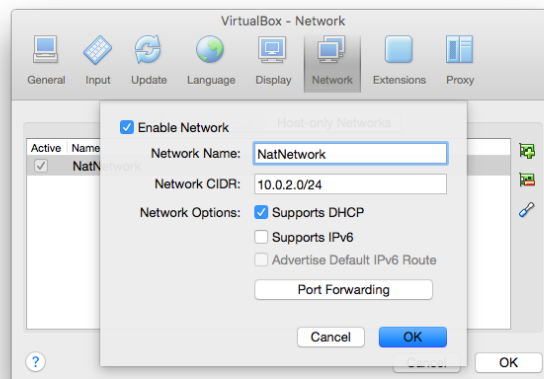
Correcting my mistake to make each IP instead use a subnet mask of /24, I could now ping each VM.



```
hslave1 [Running]
Last login: Tue Jan 19 21:31:47 on tty1
[root@hslave1 ~]# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP
    qlen 1000
    link/ether 08:00:27:bf:82:d2 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.11/24 brd 10.0.2.255 scope global enp0s3
        valid_lft forever preferred_lft forever
[root@hslave1 ~]# ping hslave2
PING hslave2 (10.0.2.12) 56(84) bytes of data:
64 bytes from hslave2 (10.0.2.12): icmp_seq=1 ttl=64 time=0.618 ms
64 bytes from hslave2 (10.0.2.12): icmp_seq=2 ttl=64 time=0.409 ms
64 bytes from hslave2 (10.0.2.12): icmp_seq=3 ttl=64 time=0.403 ms
--- hslave2 ping statistics ---
7 packets transmitted, 7 received, 0% packet loss, time 2005ms
rtt min/avg/max/mdev = 0.344/0.447/0.613/0.103 ms
[root@hslave1 ~]#

hslave2 [Running]
Last login: Tue Jan 19 21:31:52 on tty1
[root@hslave2 ~]# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP
    qlen 1000
    link/ether 08:00:27:cc:c3:76 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.12/24 brd 10.0.2.255 scope global enp0s3
        valid_lft forever preferred_lft forever
[root@hslave2 ~]# ping hslave1
PING hslave1 (10.0.2.11) 56(84) bytes of data:
64 bytes from hslave1 (10.0.2.11): icmp_seq=1 ttl=64 time=0.618 ms
64 bytes from hslave1 (10.0.2.11): icmp_seq=2 ttl=64 time=0.409 ms
64 bytes from hslave1 (10.0.2.11): icmp_seq=3 ttl=64 time=0.403 ms
--- hslave1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2005ms
rtt min/avg/max/mdev = 0.403/0.476/0.610/0.103 ms
[root@hslave2 ~]#
```

Now for each VM to be able to get out to the internet I would have to create another adapter on each with NAT mode, but while I was changing modes I noticed another mode called "NAT Network". I was curious if this was perhaps a hybrid that does the equivalent of both NAT and Internal Network so I decided to try it out. Sure enough, my thought was correct, it is the best of both worlds. In fact I don't understand why they don't make this the default. The only thing different I had to do first was define the NAT Network, which I just went with the default because it matched the network I was trying to make between the hadoop machines.



I was now ready to clone the final slave and then bring the master up. I had to fix the master's subnet mask but then I was ready to run the `ssh-keygen` command and copy to the slaves. I realized this would be easier to accomplish from an SSH session, so I shut down the host and added a Host only adapter so that my host machine could SSH to it and then continued with the copying of keys.

```

Desktop — hadoop@hmaster:~ — ssh root@192.168.56.101 — ttys007 — 160x51
splunk@inxspn... splunk@inxspn... splunk@inxspn... splunk@inxspn... hadoop@week1... splunk@inxspe... hadoop@hmaster...

[hadoop@hmaster ~]$ su hadoop
[hadoop@hmaster root]$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/hadoop/.ssh/id_rsa):
[Enter passphrase (empty for no passphrase):
[Enter same passphrase again:
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:
81:29:db:a3:85:09:ce:bb:94:1c:1b:00:16:91:da:69 hadoop@hmaster.local
The key's randomart image is:
+--[ RSA 2048 ]-----+
|          |
|          |
|          |
|          |
|          |
|          |
|          |
|          |
|          |
|          |
+-----+
[hadoop@hmaster root]$ ssh-copy-id -i ~/.ssh/id_rsa.pub hadoop@hmaster
The authenticity of host 'hmaster (10.0.2.10)' can't be established.
ECDSA key fingerprint is fd:62:bc:40:ef:fc:1a:9e:9b:30:f9:23:d9:cb:f4:de.
Are you sure you want to continue connecting (yes/no)? yes
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the new keys
hadoop@hmaster's password:
Number of key(s) added: 1
Now try logging into the machine, with: "ssh 'hadoop@hmaster'"
and check to make sure that only the key(s) you wanted were added.
[hadoop@hmaster root]$ ssh-copy-id -i ~/.ssh/id_rsa.pub hadoop@hslave1
The authenticity of host 'hslave1 (10.0.2.11)' can't be established.
ECDSA key fingerprint is fd:62:bc:40:ef:fc:1a:9e:9b:30:f9:23:d9:cb:f4:de.
Are you sure you want to continue connecting (yes/no)? yes
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the new keys
hadoop@hslave1's password:
Number of key(s) added: 1
Now try logging into the machine, with: "ssh 'hadoop@hslave1'"
and check to make sure that only the key(s) you wanted were added.
[hadoop@hmaster root]$

```

Last I was ready to complete the final config appropriate for the master.

```

[hadoop@hmaster root]$ mkdir /home/hadoop/namenode
[hadoop@hmaster root]$ chown hadoop.hadoop /home/hadoop/namenode
[hadoop@hmaster root]$ vi /opt/hadoop/etc/hadoop/hdfs-site.xml
[hadoop@hmaster root]$ tail /opt/hadoop/etc/hadoop/hdfs-site.xml
</property>
<property>
  <name>dfs.datanode.data.dir</name>
  <value>/home/hadoop/datanode</value>
</property>
<property>
  <name>dfs.namenode.data.dir</name>
  <value>/home/hadoop/namenode</value>
</property>
</configuration>
[hadoop@hmaster root]$ vi /opt/hadoop/etc/hadoop/mapred-site.xml.template
[hadoop@hmaster root]$ tail -5 /opt/hadoop/etc/hadoop/mapred-site.xml.template
<property>
  <name>mapreduce.framework.name</name>
  <value>yarn</value> <!-- and not local (l) -->
</property>
</configuration>
[hadoop@hmaster root]$ vi /opt/hadoop/etc/hadoop/yarn-site.xml
[hadoop@hmaster root]$ tail /opt/hadoop/etc/hadoop/yarn-site.xml
</property>
<property>
  <name>yarn.nodemanager.hostname</name>
  <value>hmaster</value> <!-- or hslave1, hslave2, hslave3 -->
</property>
<property>
  <name>yarn.nodemanager.aux-services</name>
  <value>mapreduce_shuffle</value>
</property>
</configuration>
[hadoop@hmaster root]$ vi /opt/hadoop/etc/hadoop/slaves
[hadoop@hmaster root]$ tail -5 /opt/hadoop/etc/hadoop/slaves
localhost
hmaster
hslave1
hslave2
hslave3
[hadoop@hmaster root]$

```

Now I was ready to format the hdfs using the `hdfs namenode -format` command, which ran as expected. When I ran the `start-dfs.sh` script, I received an error about localhost: Host key verification failed.. Earlier when I edited the slave file, I left "localhost" value in it, I suspect this is the cause of this error. So I removed localhost from the file and ran `stop-dfs.sh` and then re-ran the `start-dfs.sh` script.

```

[hadoop@hmaster root]$ hdfs namenode -format
16/01/20 11:47:38 INFO namenode.NameNode: STARTUP_MSG:
*****
STARTUP_MSG: Starting NameNode
STARTUP_MSG: host = hmaster/10.0.2.10
STARTUP_MSG: args = [-format]
STARTUP_MSG: version = 2.7.1
STARTUP_MSG: classpath = /opt/hadoop/etc/hadoop:/opt/hadoop/share/hadoop/common/lib/gson-2.2.4.jar:/opt/hadoop/share/hadoop/common/lib/jackson-jaxrs-1.9.13.jar:/opt/hadoop/share/hadoop/common/lib/jsp-api-2.1.jar:/opt/hadoop/share/hadoop/common/lib/apached-kerberos-codec-2.0.0-M15.jar:/opt/hadoop/share/hadoop/common/lib/httpclient-4.2.5.jar:/opt/hadoop/share/hadoop/common/lib/incubating-jar:/opt/hadoop/share/hadoop/common/lib/jets3t-0.9.0.jar:/opt/hadoop/hadoop-auth-2.7.1.jar:/opt/hadoop/share/hadoop/common/lib/java-xmlbuilder-0.4.1.jar:/opt/hadoop/share/hadoop/common/lib/slf4j-api-1.7.10.jar:/opt/hadoop/share/hadoop/common/lib/jersey-servlet/hadoop/share/hadoop/common/lib/avro-1.7.4.jar:/opt/hadoop/share/hadoop/common/lib/curator-recipes-2.7.1.jar:/opt/hadoop/share/hadoop/common/lib/netty-3.6.2.Final.jar:/opt/hadoop/share/hadoop/common/lib/com.google.protobuf-java-2.5.0.jar:/opt/hadoop/share/hadoop/common/lib/commons-compress-1.4.1.jar:/opt/hadoop/share/hadoop-common-2.7.1.jar:/opt/hadoop/share/hadoop-common/lib/jersey-core-1.9.jar:/opt/hadoop
16/01/20 11:47:40 INFO namenode.NameNode: SHUTDOWN_MSG:
/*****
SHUTDOWN_MSG: Shutting down NameNode at hmaster/10.0.2.10
*****
[hadoop@hmaster root]$ start-dfs.sh
Starting namenodes on [hmaster]
hmaster: starting namenode, logging to /opt/hadoop/logs/hadoop-hadoop-namenode-hmaster.
The authenticity of host 'localhost (127.0.0.1)' can't be established.
ECDSA key fingerprint is fd:62:bc:40:ef:fc:1a:9e:9b:30:f9:23:d9:cb:f4:4e.
Are you sure you want to continue connecting (yes/no)? hslave2: starting datanode, logging to /opt/hadoop/logs/hadoop-hadoop-datanode-hslave1.
hslave1: starting datanode, logging to /opt/hadoop/logs/hadoop-hadoop-datanode-hslave1.
hmaster: starting datanode, logging to /opt/hadoop/logs/hadoop-hadoop-datanode-hmaster.
hslave3: starting datanode, logging to /opt/hadoop/logs/hadoop-hadoop-datanode-hslave3.
localhost: Host key verification failed
Starting secondary namenodes [0.0.0.0]
The authenticity of host '0.0.0.0 (0.0.0.0)' can't be established.
ECDSA key fingerprint is fd:62:bc:40:ef:fc:1a:9e:9b:30:f9:23:d9:cb:f4:4e.
Are you sure you want to continue connecting (yes/no)? yes
0.0.0.0: Warning: Permanently added '0.0.0.0' (ECDSA) to the list of known hosts.
0.0.0.0: starting secondarynamenode, logging to /opt/hadoop/logs/hadoop-hadoop-secondary
16/01/20 11:47:40 INFO namenode.NameNode: SHUTDOWN_MSG:
/*****
SHUTDOWN_MSG: Shutting down NameNode at hmaster/10.0.2.10
*****

```

After re-running the `start-dfs.sh` script again, this time it worked. The instructions for the assignment say "At this point I see the NameNode, DataNode, and SecondaryNameNode running on hmaster – using `jps`. On the slaves I see NameNode and DataNode running." But when I checked `jps` on each machine, I found that hmaster had the correct services running, but I only had DataNode running on the slaves not NameNode as the instructions said. However later in the assignment it shows the output of `start-dfs.sh` and that output only shows NameNode being started on hmaster so I assume I have the correct services running in the right places. At this point I had to get back to work so I paused my VMs to work on them later, when I was able to get back to work on the assignment and had unpaused the machines I now only had DataNode running on hslave1. I then tried to stop and start the services but still ended up with the same issue. I ended up finding a Stackoverflow.com (2013) post that led me to believe I needed to run the `hdfs namenode -format` command again. Once I did this, then DataNode was not starting on any of the hosts. Back to Stackoverflow.com (2015) and realized in the logs that it was complaining that the clusterID for the namenode and datanode did not match:

```

2016-01-20 22:18:24,722 INFO org.apache.hadoop.hdfs.server.common.Storage: Lock on /home/hadoop/datanode/in_use.lock acquired by nodename 1214@hmaster
2016-01-20 22:18:24,735 WARN org.apache.hadoop.hdfs.server.common.Storage: java.io.IOException: Incompatible clusterIDs in /home/hadoop/datanode: namenode clusterID = CID-5941a2e9-2c70-40c1-a985-f88247714cf
c; datanode clusterID = CID-aeeff11ab-85c1-4a1c-beld-da489b8619bc
2016-01-20 22:18:24,736 FATAL org.apache.hadoop.hdfs.server.datanode.DataNode: Initialization failed for Block pool <registering> (Datanode Uuid unassigned) service to hmaster/10.0.2.10:9000. Exiting.
java.io.IOException: All specified directories are failed to load.
at org.apache.hadoop.hdfs.server.datanode.DataStorage.recoverTransitionRead(DataStorage.java:477)

```

So after editing the clusterID line in `/home/hadoop/datanode/current/VERSION` on hmaster and hslave1 (the other slaves lacked the file) and restarting this time I was back to running:

```

[hadoop@hmaster current]$ start-dfs.sh
Starting namenodes on [hmaster]
hmaster: starting namenode, logging to /opt/hadoop/logs/hadoop-hadoop-namenode-hmaster.local.out
hslave1: starting datanode, logging to /opt/hadoop/logs/hadoop-hadoop-datanode-hslave1.local.out
hmaster: starting datanode, logging to /opt/hadoop/logs/hadoop-hadoop-datanode-hmaster.local.out
hslave2: starting datanode, logging to /opt/hadoop/logs/hadoop-hadoop-datanode-hslave2.local.out
hslave3: starting datanode, logging to /opt/hadoop/logs/hadoop-hadoop-datanode-hslave3.local.out
Starting secondary namenodes [0.0.0.0]
0.0.0.0: starting secondarynamenode, logging to /opt/hadoop/logs/hadoop-hadoop-secondarynamenode-hmaster.local.out
[hadoop@hmaster current]$ for line in $(cat /opt/hadoop/etc/hadoop/slaves);do echo "Host: $line";ssh $line jps;echo " ";done;
Host: hmaster
1094 NameNode
1511 Jps
1199 DataNode
1359 SecondaryNameNode
Host: hslave1
1010 Jps
941 DataNode
Host: hslave2
976 Jps
937 DataNode
Host: hslave3
914 DataNode
954 Jps

```

Finishing the assignment off by starting yarn and running the wordcount exercise. Using my one liner to check `jps` on all machines again, I noticed that similarly to before DataNode had stopped running on hslave2 and 3 but NodeManager was running.

```
[hadoop@hmaster ~]$ ls
datanode namenode shakespeare.txt tmp
[hadoop@hmaster ~]$ start-yarn.sh
starting yarn daemons
starting resourcemanager, logging to /opt/hadoop/logs/yarn-hadoop-resourcemanager-hmaster.local.out
hslave1: starting nodemanager, logging to /opt/hadoop/logs/yarn-hadoop-nodemanager-hslave1.local.out
hslave2: starting nodemanager, logging to /opt/hadoop/logs/yarn-hadoop-nodemanager-hslave2.local.out
hslave3: starting nodemanager, logging to /opt/hadoop/logs/yarn-hadoop-nodemanager-hslave3.local.out
hmaster: starting nodemanager, logging to /opt/hadoop/logs/yarn-hadoop-nodemanager-hmaster.local.out
[hadoop@hmaster ~]$ for line in $(cat /opt/hadoop/etc/hadoop/slaves);do echo "Host: $line";ssh $line jps;echo " "; done;
Host: hmaster
1712 ResourceManager
1894 NameNode
1815 NodeManager
2107 Jps
1199 DataNode
1359 SecondaryNameNode
Round 1 items
Host: hslave1
1139 Jps
941 DataNode
1885 NodeManager
Host: hslave2
1842 NodeManager
1878 Jps
Host: hslave3
1819 NodeManager
1855 Jps

[hadoop@hmaster ~]$ hadoop fs -ls
ls: '.': No such file or directory
[hadoop@hmaster ~]$ hadoop fs -mkdir -p /user/hadoop
[hadoop@hmaster ~]$ hadoop fs -ls
[hadoop@hmaster ~]$ hadoop fs -mkdir shakespeare
[hadoop@hmaster ~]$ hadoop fs -mkdir shakespeare/input
[hadoop@hmaster ~]$ hadoop fs -copyFromLocal ~/shakespeare.txt shakespeare/input
[hadoop@hmaster ~]$ hadoop fs -ls shakespeare/input
Round 1 items
-rw-r--r-- 3 hadoop supergroup 4538523 2016-01-21 11:42 shakespeare/input/shakespeare.txt
[hadoop@hmaster ~]$ hadoop jar /opt/hadoop/share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.1.jar wordcount shakespeare/input shakespeare/output
16/01/21 11:45:07 INFO Configuration.deprecation: session.id is deprecated. Instead, use dfs.metrics.session-id
16/01/21 11:45:07 INFO jvm.JvmMetrics: Initializing JVM Metrics with processName=JobTracker, sessionId=
16/01/21 11:45:08 INFO input.FileInputFormat: Total input paths to process : 1
16/01/21 11:45:08 INFO mapreduce.JobSubmitter: number of splits:1
16/01/21 11:45:09 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_local1616995523_0001
16/01/21 11:45:09 INFO mapreduce.Job: The url to track the job: http://localhost:8080/
16/01/21 11:45:09 INFO mapreduce.Job: Running job: job_local1616995523_0001
16/01/21 11:45:09 INFO mapred.LocalJobRunner: OutputCommitter set in config null
```

I went ahead and continued the assignment and was able to complete the wordcount exercise.

```

16/01/21 11:45:16 INFO mapreduce.Job: Job job_local1616095523_0001 completed successfully
16/01/21 11:45:16 INFO mapreduce.Job: Counters: 35
File System Counters
  FILE: Number of bytes read=1515012
  FILE: Number of bytes written=2555164
  FILE: Number of read operations=6
  FILE: Number of large read operations=0
  FILE: Number of write operations=0
  HDFS: Number of bytes read=9077046
  HDFS: Number of bytes written=356409
  HDFS: Number of read operations=13
  HDFS: Number of large read operations=0
  HDFS: Number of write operations=4
Map-Reduce Framework
  Map input records=129107
  Map output records=980637
  Map output bytes=8406347
  Map output materialized bytes=483860
  Input split bytes=198
  Combine input records=980637
  Combine output records=33505
  Reduce input groups=33505
  Reduce shuffle bytes=483860
  Reduce input records=33505
  Reduce output records=33505
  Spilled Records=67010
  Shuffled Maps=1
  Failed Shuffles=0
  Merged Map outputs=1
  GC time elapsed (ms)=142
  Total committed heap usage (bytes)=240033792
Shuffle Errors
  BAD_ID=0
  CONNECTION=0
  IO_ERROR=0
  WRONG_LENGTH=0
  WRONG_MAP=0
  WRONG_REDUCE=0
File Input Format Counters
  Bytes Read=4530523
File Output Format Counters
  Bytes Written=356409
[hadoop@hmaster ~]$ hadoop fs -ls shakespeare/output
Found 2 items
-rw-r--r--   3 hadoop supergroup          0 2016-01-21 11:45 shakespeare/output/_SUCCESS
-rw-r--r--   3 hadoop supergroup    256409 2016-01-21 11:45 shakespeare/output/part-r-000000 ~/shakespeare/output.txt
16/01/21 11:46:59 WARN hdfs.DFSClient: DFSInputStream has been closed already
[hadoop@hmaster ~]$ ls
datanode namenode shakespeareout.txt shakespeare.txt tmp
[hadoop@hmaster ~]$ head shakespeareout.txt
1
10526
183
1
By 1
1
1'twas 1
1
1
1>About 1
1
1All 1
1As 1
1
1As 1
1

```

I then checked jps on all machines again and found that again nothing was running on hslave2 or 3. I realized that the assignment did not recommend to keep checking jps, so if I hadn't I would not even had known that



```

STARTUP_MSG: java -Xlog:0:65
2016-01-21 11:15:20.095 INFO org.apache.hadoop.hdfs.server.datanode.DataNode: registered UNIX signal handlers for [TERM, HUP, INT]
2016-01-21 11:16:00.928 FATAL org.apache.hadoop.hdfs.server.datanode.DataNode: Exception in secureMain
java.net.UnknownHostException: hlsave2.local: hlsave2.local: unknown error
    at java.net.InetAddress.getLocalHost(InetAddress.java:1558)
    at org.apache.hadoop.security.SecurityUtil.(SecurityUtil.java:198)
    at org.apache.hadoop.security.SecurityUtil.login(SecurityUtil.java:218)
    at org.apache.hadoop.hdfs.server.datanode.DataNode.(DataNode.java:2258)
    at org.apache.hadoop.hdfs.server.datanode.DataNode.createDataNode(DataNode.java:1387)
    at org.apache.hadoop.hdfs.server.datanode.DataNode.secureMain(DataNode.java:2484)
    at org.apache.hadoop.hdfs.server.datanode.DataNode.main(DataNode.java:2588)
Caused by: java.net.UnknownHostException: hlsave2.local: unknown error
    at java.net.Inet4AddressImpl.lookupAllHostAddr(Native Method)
    at java.net.Inet4AddressImpl.lookupAllHostAddr(InetAddress.java:928)
    at java.net.InetAddress.getAddressesFromNameService(InetAddress.java:1323)
    at java.net.InetAddress.getLocalHost(InetAddress.java:1368)
    ... 6 more
2016-01-21 11:16:00.935 INFO org.apache.hadoop.util.ExitUtil: Exiting with status 1
2016-01-21 11:16:00.946 INFO org.apache.hadoop.hdfs.server.datanode.DataNode: SHUTDOWN_MSG:
*****
SHUTDOWN_MSG: Shutting down DataNode at java.net.UnknownHostException: hlsave2.local: hlsave2.local: unknown error

```

```

Total Committed heap usage (bytes)=242499472
Shuffle Errors
  BAD_ID=0
  CONNECTION=0
  IO_EXCEPTION=0
  WRONG_LENGTH=0
  WRONG_MAP=0
  WRONG_REDUCE=0
File Input Format Counters
  Bytes Read=153552
File Output Format Counters
  Bytes Written=356489
[hadoop@hmaster ~]$ hadoop fs -ls /shakespeare/output
Found 2 items
-rw-r--r-- 3 hadoop supergroup      0 2016-01-22 11:18 /shakespeare/output/_SUCCESS
-rw-r--r-- 3 hadoop supergroup  356489 2016-01-22 11:18 /shakespeare/output/part-r-00000
[hadoop@hmaster ~]$ mv /shakespeare/output.txt /shakespeare/output.txt.old
[hadoop@hmaster ~]$ hadoop fs -copyToLocal /shakespeare/output/part-r-00000 ~/shakespeare/output.txt
2016/01/22 11:19:17 WARN hdfs.DFSClient: DFSInputStream has been closed already
[hadoop@hmaster ~]$ head /shakespeare/output.txt
1      18526
2      163
copy 1
txt 1
rtweet 1
about 1
all 1
aks 1
do 1
[hadoop@hmaster ~]$ diff /shakespeare/output.txt /shakespeare/output.txt.old
[hadoop@hmaster ~]$ for line in $(cat /opt/hadoop/etc/hadoop/slaves); do echo
Host: hmaster
1360 SecondaryNameNode
1093 NameNode
1703 NodeManager
1592 ResourceManager
2381 Jps
1198 DataNode
Host: hslave1
1073 NodeManager
1194 Jps
942 DataNode
Host: hslave2
937 DataNode
1194 Jps
1069 NodeManager
Host: hslave3
961 DataNode
1217 Jps
1092 NodeManager

```

Stackoverflow.com, 2013. Retrieved from <http://stackoverflow.com/questions/19085978/data-node-does-not-start>

Stackoverflow.com, 2015. Retrieved from <http://stackoverflow.com/questions/30521474/hadoop-hdfs-formatting-gets-error-failed-for-block-pool>