This document will guide you regarding how to install multinode cloudera hadoop cluster cdh5.4.0 without Cloudera manager.

In this tutorial I have used 2 Centos 6.6 virtual machines viz. master.hadoop.com & slave.hadoop.com.

Prerequisites:

CentOS 6.X

Open JDK 1.7:-

sudo yum install java-1.7.0-openjdk-devel

Master machine – master.hadoop.com (ip of first machine)

Daemons that we are going to install on master are :

Namenode

HistoryServer

Slave machine – slave.hadoop.com (ip of second machine)

Daemons that we are going to install on master are :

Resource Manager (Yarn)

Node-manager

Secondary Namenode

Datanode

Important configuration before proceeding further: please add both the hostname and ip information to /etc/hosts file on each host.

```
[root@master ~]# cat /etc/hosts
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6
192.168.111.130         master.hadoop.com
192.168.111.131         slave.hadoop.com
```

```
[root@slave ~]# cat /etc/hosts
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6
192.168.111.130         master.hadoop.com
192.168.111.131        slave.hadoop.com
```

Please verify that both the hosts are ping'able from each other 

Also,

Please stop the firewall and disable the selinux.

To stop firewall in centos :

```
service iptables stop && chkconfig iptables off
```

To disable selinux :

```
vim /etc/selinux/config
```

once file is opened, please verify that “SELINUX=disabled” is set.

1.Date should be in sync

Please make sure that master and slave machine’s date is in sync, if not please do it so by configuring NTP.

2.Passwordless ssh must be setup from master –> slave

To setup passwordless ssh follow below procedure :

2a. Generate rsa key pair using ssh-keygen command

```
[root@master conf]# ssh-keygen
Generating public/private rsa key pair.
```

```
Enter file in which to save the key (/root/.ssh/id_rsa):
/root/.ssh/id_rsa already exists.
Overwrite (y/n)?
```

2b. Copy generated public key to slave.hadoop.com

```
[root@master conf]# ssh-copy-id -i ~/.ssh/id_rsa.pub root@slave.hadoop.com
```

Now try logging into the machine, with “ssh ‘root@slave.hadoop.com'”, and check in:

.ssh/authorized_keys

to make sure we haven’t added extra keys that you weren’t expecting.

2c. Now try connecting to slave.hadoop.com using ssh

```
[root@master conf]# ssh root@slave.hadoop.com
Last login: Fri Apr 24 14:20:43 2015 from master.hadoop.com
[root@slave ~]# logout
Connection to slave.hadoop.com closed.
[root@master conf]#
```

That’s it! You have successfully configured passwordless ssh between master and slave node.

3. Internet connection

Please make sure that you have working internet connection, as we are going to download CDH packages in next steps.

4. Install cdh repo

4a. download cdh repo rpm

```
[root@master ~]# wget http://archive.cloudera.com/cdh5/one-click-install/redhat/6/x86_64/cloudera-cdh-5-0.x86_64.rpm
```

4b. install cdh repo downloaded in above step

```
[root@master ~]# yum --nogpgcheck localinstall cloudera-cdh-5-0.x86_64.rpm
Loaded plugins: fastestmirror, refresh-packagekit, security
setting up Local Package Process
....
Complete!
```

4c. do the same steps on slave node

```
[root@slave ~]# wget http://archive.cloudera.com/cdh5/one-click-install/redhat/6/x86_64/cloudera-cdh-5-0.x86_64.rpm
```

```
[root@slave ~]# yum --nogpgcheck localinstall cloudera-cdh-5-0.x86_64.rpm
Loaded plugins: fastestmirror, refresh-packagekit, security
Setting up Local Package Process
……
Complete!
```

5. Install and deploy ZooKeeper.

```
[root@master ~]# yum -y install zookeeper-server
Loaded plugins: fastestmirror, refresh-packagekit, security
Setting up Install Process
…..
Complete!
```

5a. create zookeeper dir and apply permissions

```
[root@master ~]# mkdir -p /var/lib/zookeeper
[root@master ~]# chown -R zookeeper /var/lib/zookeeper/
```

5b. Init zookeeper and start the service

```
[root@master ~]# service zookeeper-server init
No myid provided, be sure to specify it in /var/lib/zookeeper/myid if using non-standalone
```

```
[root@master ~]# service zookeeper-server start
JMX enabled by default
Using config: /etc/zookeeper/conf/zoo.cfg
Starting zookeeper ... STARTED
```

6. Install namenode on master machine

```
yum -y install hadoop-hdfs-namenode
```

7. Install secondary namenode on slave machine

```
yum -y install hadoop-hdfs-secondarynamenode
```

8. Install resource manager on slave machine

```
yum -y install hadoop-yarn-resourcemanager
```

9. Install nodemanager, datanode & mapreduce on slave node

```
yum -y install hadoop-yarn-nodemanager hadoop-hdfs-datanode hadoop-mapreduce
```

10. Install history server and yarn proxyserver on master machine

```
yum -y install hadoop-mapreduce-historyserver hadoop-yarn-proxyserver
```

11. On both the machine you can install hadoop-client package

```
yum -y install hadoop-client
```

Now we are done with the installation, it's time to deploy HDFS!

1. On each node, execute below commands :

```
[root@master ~]# cp -r /etc/hadoop/conf.empty /etc/hadoop/conf.my_cluster
[root@master ~]# alternatives --install /etc/hadoop/conf hadoop-conf /etc/hadoop/conf.my_cluster 50
[root@master ~]# alternatives --set hadoop-conf /etc/hadoop/conf.my_cluster
```

```
[root@slave ~]# cp -r /etc/hadoop/conf.empty /etc/hadoop/conf.my_cluster
[root@slave ~]# alternatives --install /etc/hadoop/conf hadoop-conf /etc/hadoop/conf.my_cluster 50
```

```
[root@slave ~]# alternatives --set hadoop-conf /etc/hadoop/conf.my_cluster
```

2. Let's configure hdfs properties now :

Goto /etc/hadoop/conf/ dir on master node and edit below property files:

2a. vim /etc/hadoop/conf/core-site.xml

Add below lines in it under <configuration> tag

```
<property>
<name>fs.defaultFS</name>
<value>hdfs://master.hadoop.com:8020</value>
</property>
```

2b. vim /etc/hadoop/conf/hdfs-site.xml

```
<property>
<name>dfs.permissions.superusergroup</name>
<value>hadoop</value>
</property>
<property>
<name>dfs.namenode.name.dir</name>
<value>file:///data/1/dfs/nn,file:///nfsmount/dfs/nn</value>
```

```
</property>
<property>
<name>dfs.datanode.data.dir</name>
<value>file:///data/1/dfs/dn,file:///data/2/dfs/dn,file:///data/3/dfs/dn,file:///data/4/dfs/dn</value
></property>
<property>
<name>dfs.namenode.http-address</name>
<value>192.168.111.130:50070</value>
<description>
The address and the base port on which the dfs NameNode Web UI will listen.
</description>
</property>
```

3. scp core-site.xml and hdfs-site.xml to slave.hadoop.com at /etc/hadoop/conf/

```
[root@master conf]# scp core-site.xml hdfs-site.xml slave.hadoop.com:/etc/hadoop/conf/
core-site.xml                                 100% 1
001     1.0KB/s   00:00
hdfs-site.xml                                 100% 1
669     1.6KB/s   00:00
[root@master conf]#
```

4. Create local directories:

On master host run below commands:

```
mkdir -p /data/1/dfs/nn /nfsmount/dfs/nn
chown -R hdfs:hdfs /data/1/dfs/nn /nfsmount/dfs/nn
chmod 700 /data/1/dfs/nn /nfsmount/dfs/nn
chmod go-rx /data/1/dfs/nn /nfsmount/dfs/nn
```

On slave host run below commands:

```
mkdir -p /data/1/dfs/dn /data/2/dfs/dn /data/3/dfs/dn /data/4/dfs/dn
chown -R hdfs:hdfs /data/1/dfs/dn /data/2/dfs/dn /data/3/dfs/dn /data/4/dfs/dn
```

5. Format the namenode :

```
sudo -u hdfs hdfs namenode -format
```

6. Start hdfs services

Run below commands on master and slave

```
for x in `cd /etc/init.d ; ls hadoop-hdfs-*` ; do service $x start ; done
```

7. Create hdfs tmp dir

Run on any of the hadoop node

```
[root@slave ~]# sudo -u hdfs hadoop fs -mkdir /tmp
[root@slave ~]# sudo -u hdfs hadoop fs -chmod -R 1777 /tmp
```

Congratulations! You have deployed hdfs successfully 

Deploy Yarn

1. Prepare yarn configuration properties

replace your /etc/hadoop/conf/mapred-site.xml with below contents on master host

[root@master conf]# cat mapred-site.xml

```
<configuration>
<property>
<name>mapreduce.framework.name</name>
<value>yarn</value>
</property>
<property>
<name>yarn.app.mapreduce.am.staging-dir</name>
```

```
<value>/user</value>
</property>
</configuration>
```

2. Replace your /etc/hadoop/conf/yarn-site.xml with below contents on master host

[root@master conf]# cat yarn-site.xml

```
<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.log-aggregation-enable</name>
<value>true</value>
</property>
<property>
<description>List of directories to store localized files in.</description>
<name>yarn.nodemanager.local-dirs</name>
<value>file:///var/lib/hadoop-yarn/cache/${user.name}/nm-local-dir</value>
</property>
<property>
<description>Where to store container logs.</description>
<name>yarn.nodemanager.log-dirs</name>
```

```
<value>file:///var/log/hadoop-yarn/containers</value>
</property>
<property>
<description>Where to aggregate logs to.</description>
<name>yarn.nodemanager.remote-app-log-dir</name>
<value>hdfs://master.hadoop.com:8020/var/log/hadoop-yarn/apps</value>
</property>
<property>
<description>Classpath for typical applications.</description>
<name>yarn.application.classpath</name>
<value>
$HADOOP_CONF_DIR,
$HADOOP_COMMON_HOME/*,$HADOOP_COMMON_HOME/lib/*,
$HADOOP_HDFS_HOME/*,$HADOOP_HDFS_HOME/lib/*,
$HADOOP_MAPRED_HOME/*,$HADOOP_MAPRED_HOME/lib/*,
$HADOOP_YARN_HOME/*,$HADOOP_YARN_HOME/lib/*
</value>
</property>
<property>
<name>yarn.resourcemanager.hostname</name>
<value>slave.hadoop.com</value>
</property>
<property>
<name>yarn.nodemanager.local-dirs</name>
<value>file:///data/1/yarn/local,file:///data/2/yarn/local,file:///data/3/yarn/local</value>
</property>
<property>
<name>yarn.nodemanager.log-dirs</name>
<value>file:///data/1/yarn/logs,file:///data/2/yarn/logs,file:///data/3/yarn/logs</value>
</property>
</configuration>
```

3. Copy modified files to slave machine.

```
[root@master conf]# scp mapred-site.xml yarn-site.xml slave.hadoop.com:/etc/hadoop/conf/
mapred-site.xml                                                     100%
1086     1.1KB/s   00:00
yarn-site.xml                                                       100%
2787     2.7KB/s   00:00
[root@master conf]#
```

4. Configure local directories for yarn

To be done on yarn machine i.e. slave.hadoop.com in our case

```
[root@slave ~]# mkdir -p /data/1/yarn/local /data/2/yarn/local /data/3/yarn/local /data/4/yarn/loc
al
[root@slave ~]# mkdir -p /data/1/yarn/logs /data/2/yarn/logs /data/3/yarn/logs /data/4/yarn/logs
[root@slave ~]# chown -R yarn:yarn /data/1/yarn/local /data/2/yarn/local /data/3/yarn/local /data/
4/yarn/local
[root@slave ~]# chown -R yarn:yarn /data/1/yarn/logs /data/2/yarn/logs /data/3/yarn/logs /data/4/
yarn/logs
```

5. Configure the history server.

Add below properties in mapred-site.xml

```
<property>
<name>mapreduce.jobhistory.address</name>
<value>master.hadoop.com:10020</value>
</property>
<property>
<name>mapreduce.jobhistory.webapp.address</name>
<value>master.hadoop.com:19888</value>
</property>
```

6. Configure proxy settings for history server

Add below properties in /etc/hadoop/conf/core-site.xml

```
<property>
<name>hadoop.proxyuser.mapred.groups</name>
<value>*</value>
</property>
<property>
<name>hadoop.proxyuser.mapred.hosts</name>
<value>*</value>
</property>
```

7. Copy modified files to slave.hadoop.com

```
[root@master conf]# scp mapred-site.xml core-site.xml slave.hadoop.com:/etc/hadoop/conf/
mapred-site.xml                                                     100
% 1299     1.3KB/s   00:00
core-site.xml                                                       100% 1
174     1.2KB/s   00:00
[root@master conf]#
```

8. Create history directories and set permissions

```
[root@master conf]# sudo -u hdfs hadoop fs -mkdir -p /user/history
[root@master conf]# sudo -u hdfs hadoop fs -chmod -R 1777 /user/history
[root@master conf]# sudo -u hdfs hadoop fs -chown mapred:hadoop /user/history
```

9. Create log directories and set permissions

```
[root@master conf]# sudo -u hdfs hadoop fs -mkdir -p /var/log/hadoop-yarn
[root@master conf]# sudo -u hdfs hadoop fs -chown yarn:mapred /var/log/hadoop-yarn
```

10. Verify hdfs file structure

```
[root@master conf]# sudo -u hdfs hadoop fs -ls -R /
```

```
drwxrwxrwt   - hdfs hadoop         0 2015-04-25 01:16 /tmp
drwxr-xr-x   - hdfs hadoop         0 2015-04-25 02:52 /user
drwxrwxrwt   - mapred hadoop         0 2015-04-25 02:52 /user/history
drwxr-xr-x   - hdfs   hadoop         0 2015-04-25 02:53 /var
drwxr-xr-x   - hdfs   hadoop         0 2015-04-25 02:53 /var/log
drwxr-xr-x   - yarn   mapred         0 2015-04-25 02:53 /var/log/hadoop-yarn
[root@master conf]#
```

11. Start yarn and Jobhistory server

On slave.hadoop.com

```
[root@slave ~]# sudo service hadoop-yarn-resourcemanager start
starting resourcemanager, logging to /var/log/hadoop-yarn/yarn-yarn-resourcemanager-slave.hadoop.com.out
Started Hadoop resourcemanager:                            [ OK ]
[root@slave ~]#
```

```
[root@slave ~]# sudo service hadoop-yarn-nodemanager start
starting nodemanager, logging to /var/log/hadoop-yarn/yarn-yarn-nodemanager-slave.hadoop.com.out
Started Hadoop nodemanager:                            [ OK ]
[root@slave ~]#
```

On master.hadoop.com

```
[root@master conf]# sudo service hadoop-mapreduce-historyserver start

starting historyserver, logging to /var/log/hadoop-mapreduce/mapred-mapred-historyserver-master
.hadoop.com.out

15/04/25 02:56:01 INFO hs.JobHistoryServer: STARTUP_MSG:

/************************************************************

STARTUP_MSG: Starting JobHistoryServer

STARTUP_MSG:   host = master.hadoop.com/192.168.111.130

STARTUP_MSG:   args = []

STARTUP_MSG:   version = 2.6.0-cdh5.4.0

STARTUP_MSG:   classpath =

STARTUP_MSG:   build = http://github.com/cloudera/hadoop -r c788a14a5de9ecd968d1e2666e8765
c5f018c271; compiled by 'jenkins' on 2015-04-21T19:18Z

STARTUP_MSG:   java = 1.7.0_79

-

-

-

************************************************************/

Started Hadoop historyserver:                  [ OK ]

[root@master conf]#
```

12. Create user for running mapreduce jobs

```
[root@master conf]# sudo -u hdfs hadoop fs -mkdir /user/gaurav

[root@master conf]# sudo -u hdfs hadoop fs -chown kuldeep /user/gaurav
```

13. Important: Don’t forget to set core hadoop services to auto start when OS boot ups.

On master.hadoop.com

```
[root@master conf]# sudo chkconfig hadoop-hdfs-namenode on
[root@master conf]# sudo chkconfig hadoop-mapreduce-historyserver on
```

On slave.hadoop.com

```
[root@slave ~]# sudo chkconfig hadoop-yarn-resourcemanager on
[root@slave ~]# sudo chkconfig hadoop-hdfs-secondarynamenode on
[root@slave ~]# sudo chkconfig hadoop-yarn-nodemanager on
[root@slave ~]# sudo chkconfig hadoop-hdfs-datanode on
```

Final step : check UIs