Tutorial- Hadoop Multinode Cluster Setup on Ubuntu

Hadoop Multinode Cluster Setup for Ubuntu 12.04

Setting up a Hadoop cluster on multi node is as easy as reading this tutorial. This tutorial is a step by step guide for installation of a multi node cluster on Ubuntu 12.04.
Before setting up the cluster, let's first understand Hadoop and its modules.

What is Apache Hadoop?

Apache Hadoop is an open source java based programming framework that supports the processing of large data set in a distributed computing environment.

What are the different modules in Apache Hadoop?

Apache Hadoop framework is composed of following modules:

1. Hadoop common – collection of common utilities and libraries that support other Hadoop modules
2. Hadoop Distributed File System (HDFS) – Primary distributed storage system used by Hadoop applications to hold large volume of data. HDFS is scalable and fault-tolerant which works

management. It is an architectural center of Hadoop that allows multiple data processing engines to handle data stored in HDFS.

4. Hadoop MapReduce – A YARN based system for parallel processing of large data sets in a reliable manner.

Minimum two ubuntu machines to complete the multi node installation but it is advisable to use 3 machines for a balanced test environment. This article has used Hadoop version 2.5.2 with 3 ubuntu machines where one machine will serve as a **master** plus **slave**, and other 2 machines as **slaves**.

```
Machine 1: dzmnhdp01
IP address: 192.168.56.11
Machine 2: dzmnhdp02
IP address: 192.168.56.12
Machine 3: dzmnhdp03
IP address: 192.168.56.13
```

Hadoop daemons (perceive daemons as Windows services) are Java services which run their own JVM (Java Virtual Machine) and therefore require java installation on each machine. Secure shell (SSH) is also required to make remote login for operating securely over an unsecured network.

Install Java and SSH on all machines (nodes):

```
# Download packages and install Java and SSH
$ sudo apt-get update
$ sudo apt-get install openjdk-7-jdk
```

Add hostnames and their static IP addresses in /etc/hosts for host name resolution and comment the local host. This will help in avoiding errors of unreachable hosts.

```
$ sudo vi /etc/hosts
  192.168.56.11     dzmnhdp01
  192.168.56.12    dzmnhdp02
  192.168.56.13    dzmnhdp03
  # 127.0.0.1      localhost
```

Ping your machines for validating the host name resolution:

```
$ ping dzmnhdp01
$ ping dzmnhdp02
$ ping dzmnhdp03
```

Set up Hadoop user:

There must be a common user in all machines to administrate the cluster and this will help in making all nodes talking to each other with a password less connection under the guidance of a common user.

```
# Create Hadoop group
$ sudo addgroup hadoop

# Create a user inside “hadoop” group (enter/confirm
password and other fields can be left blank)
```

```
hduser      ALL=(ALL) ALL
```

Login as “hduser” and generate ssh key to enable a password less connection between the nodes. These steps must be performed on each node.

Alternate Option: Copying private and authorized key from on node to another also enable the password-less connection.

Generate SSH key:

```
# Login as hduser
$ su - hduser

# Generate ssh key
$ ssh-keygen -t rsa -P ""

# Enable the authorization
$ cp /home/hduser/.ssh/id_rsa.pub
/home/hduser/.ssh/authorized_keys

# Copy the public key from master to slaves and vice versa
# dzmnhdp01 to dzmnhdp02/03
# dzmnhdp02 to dzmnhdp01/03
# dzmnhdp03 to dzmnhdp01/02
$ ssh-copy-id -i /home/hduser/.ssh/id_rsa.pub
hduser@dzmnhdp02

# Modify permissions
$ sudo chmod 700 /home/hduser/.ssh ; sudo chmod 640
/home/hduser/.ssh/authorized_keys ; sudo chmod 600
/home/hduser/.ssh/id_rsa

# Check you connection between the nodes
```

Hadoop Installation:

Hadoop enables different distributed mode to run:

1. Standalone mode – Default mode of Hadoop which utilize local file system for input and output operations instead of HDFS and is mainly used for debugging purpose
2. Pseudo Distributed mode (Single node cluster) – Hadoop cluster will be set up on a single server running all Hadoop daemons on one node and is mainly used for real code to test in HDFS.
3. Fully Distributed mode (Multi node cluster) – Setting up Hadoop cluster on more than one server enabling a distributed environment for storage and processing which is mainly used for production phase.

This article objective is to set up a fully distributed Hadoop cluster on 3 servers.
Download Apache Hadoop 2.5.2 binary file on dzmnhdp01 from here or you can pick from other mirror site.

```
# Fix a base directory for Hadoop ecosystem
$ cd /usr/local

# Download the Hadoop 2.5.2 binary file
$ sudo wget
http://www.us.apache.org/dist/hadoop/core/hadoop-
2.5.2/hadoop-2.5.2.tar.gz
```

```
# Set up Hadoop environment variables
$ sudo vi ~/.bashrc
  export HADOOP_HOME=/usr/local/hadoop
  export PATH=$PATH:$HADOOP_HOME/bin:$HADOOP_HOME/sbin
$ source ~/.bashrc
```

Hadoop configuration files:

Basic configuration is the requirement of every software and therefore add below given parameters to seven important hadoop configuration files to make it run in safe environment:

Note: All child elements "property" in an xml configuration file must fall under parent element "configuration"

```
$ cd $HADOOP_HOME/etc/hadoop
```

1. **Hadoop-env.xml -** Contains the environment variables which is required by Hadoop such as log file location, java path, heap size, PIDs etc.

```
$ sudo vi hadoop-env.sh
      export   JAVA_HOME=/usr/lib/jvm/java-7-
openjdk-amd64/
  export HADOOP_HEAPSIZE=400
```

2. **core-site.xml –** Instructs the location of Namenode to Hadoop

fs.defaultFS

hdfs:*//dzmnhdp01:9000*

The name of the default file system (HDFS). A URI whose scheme

and authority determine the FileSystem implementation.

fs.trash.interval

30

Number of minutes after which the checkpoint gets deleted.

If zero, the trash feature is disabled

```
$ sudo vi hdfs-site.xml



    dfs.replication
    3

        Default block replication. The actual number of replications
        can be specified when the file is created. The default is used
        if replication is not specified in create time.



    dfs.datanode.data.dir
    file:///hdfs_storage/data

        Determines where on the local filesystem a DFS data node
        should store its blocks. If this is a comma-delimited list of
        directories, then data will be stored in all named directories,
        typically on different devices. Directories that do not exist
        are ignored.
```

4. **mapred-site.xml -** Configuration for MapReduce daemons and jobs but for Hadoop 2x it is used to point YARN framework.

```
# Create a copy of mapred file from its template
$ cp mapred-site.xml.template mapred-site.xml

# Edit the mapred file
$ sudo vi mapred-site.xml
```

```
mapred.job.tracker
dzmnhdp01:9001



mapreduce.framework.name
yarn

 The runtime framework for executing MapReduce jobs.
 Can be one of local, classic or yarn.
```

5. **yarn-site.xml -** Configuration for YARN daemons related parameters such as resource manager, node manager, container class, mapreduce class etc.

```
$ sudo vi yarn-site.xml



    yarn.nodemanager.aux-services
    mapreduce_shuffle


                                yarn.nodemanager.aux-services.mapreduce.shuffle.class
    org.apache.hadoop.mapred.ShuffleHandler



    yarn.resourcemanager.address
    dzmnhdp01:8032

        The address of applications manager interface in the RM
```

The address of scheduler interface in the RM

yarn.resourcemanager.resource-tracker.address
dzmnhdp01:8031

yarn.nodemanager.address
0.0.0.0:59392

The address of container manager interface in the RM

After updating the above given 5 configuration files, create "hdfs_storage" directory in all nodes and copy complete Hadoop 2.5.2 folder using SCP to other two nodes (dzmnhdp02 and dzmnhdp03).

```
## Node 1 (dzmnhdp01)
```

```
$ scp -r /usr/local/hadoop-2.5.2
hduser@dzmnhdp02:/home/hduser
$ scp -r /usr/local/hadoop-2.5.2
hduser@dzmnhdp03:/home/hduser

## Node 2 & 3 (dzmnhdp02 $ dzmnhdp03)

# Create hdfs directory and assign permissions
$ sudo mkdir /hdfs_storage
$ sudo mkdir /hdfs_storage/data
$ sudo chown -R hduser:hadoop /hdfs_storage

# Move the Hadoop folder to base directory, create soft
link and assign permissions
$ sudo mv /home/hduser/hadoop-2.5.2 /usr/local
$ cd /usr/local
$ sudo ln -s hadoop-2.5.2 hadoop
$ sudo chown -R hduser:hadoop hadoop
$ sudo chown -R hduser:hadoop hadoop-2.5.2

# Set up Hadoop environment variables
$ sudo vi ~/.bashrc
  export HADOOP_HOME=/usr/local/hadoop
  export PATH=$PATH:$HADOOP_HOME/bin:$HADOOP_HOME/sbin
$ source ~/.bashrc
```

Update the remaining two configuration files in the **master node** (dzmnhdp01):

- **slaves –** List of hosts, one per file, where Hadoop slave daemons will run.
- Keep it blank on other nodes

```
dzmnhdp02
dzmnhdp03
```

- **Masters –** List of hosts, one per file, where Secondary Namenode will run.
- Keep it blank on other nodes

```
# Add hosted node for secondary namenode in dzmnhdp01
$ sudo vi masters
dzmnhdp02
```


Update the parameters for Namenode directory and Secondary Namenode in node 1 & 2 (dzmnhdp01 & dzmnhdp02):

```
## Node 1 (dzmnhdp01)

$ sudo vi hdfs-site.xml

    dfs.namenode.name.dir
    file:///hdfs_storage/name

    Determines where on the local filesystem the DFS name node
```

dfs.secondary.http.address
dzmnhdp02:50090
The secondary namenode http server address and port.
If the port is 0 then the server will start on a free port.

```
## Node 2 (dzmnhdp01)

$ cd $HADOOP_HOME/etc/hadoop
$ sudo vi hdfs-site.xml
```

dfs.http.address
dzmnhdp01:50070
The address and the base port where the dfs namenode
web ui will listen on. If the port is 0 then the server will
start on a free port

dfs.secondary.http.address
dzmnhdp02:50090
The secondary namenode http server address and port.
If the port is 0 then the server will start on a free port.

fs.checkpoint.period
600
600 seconds when SNN will checkpoint NN for edits
and FSimage merge

```
    Secondary ame node should store the temporary images to
merge



    fs.checkpoint.edits.dir
    /hdfs_storage/snnedits
     Determines where on the local filesystem the DFS
     Secondary name node should store the temporary edits to merge
```

Create the secondary namenode FSimage and edits directories in node 2 (dzmnhdp02):

```
## Node 2 (dzmnhdp02)

# Create FSimage and edits directory and assign permissions
$ sudo mkdir /hdfs_storage/snnfsi
$ sudo mkdir /hdfs_storage/snnedits
$ sudo chown -R hduser:hadoop /hdfs_storage
```

Format Namenode and start DFS daemons:

```
# Format the Namenode on master node (dzmnhdp01)
$ hdfs namenode -format

# Confirm formatting by checking the VERSION
```

```
Namenode and Datanode
$ more /hdfs_storage/name/current/VERSION (on dzmnhdp01)
$ more /hdfs_storage/snnfsi/current/VERSION (on dzmnhdp02)
$ more /hdfs_storage/snnedits/current/VERSION (on
dzmnhdp02)
$ more /hdfs_storage/data/current/VERSION (on any node)

# Confirm the DFS daemons on each node
$ jps

## jps output

dzmnhdp01
2463 DataNode
2680 Jps
2241 NameNode


dzmnhdp02
2174 DataNode
2367 SecondaryNameNode
2436 Jps

dzmnhdp03
2209 DataNode
2277 Jps
```

Start YARN daemons:

```
# Start the YARN daemons (dzmnhdp01)
$ $HADOOP_HOME/sbin/start-yarn.sh

# Confirm the DFS + YARN daemons on each node
$ jps
```

```
state)
$ hdfs dfsadmin -safemode get

# Check the cluster report
$ hdfs dfsadmin -report
```

If safe mode is OFF and report display the clear picture of your cluster then you have set up a perfect Hadoop multi node cluster.

Test the environment with MapReduce:

Download an ebook to the local file system and copy it to the Hadoop file system (HDFS). By default, Hadoop folder include example jar files to help testing the environment.

```
# Download an ebook and rename it
$ cd ~
$ wget http://www.gutenberg.org/ebooks/4300.txt.utf-8
$ mv 4300.txt.utf-8 4300.txt

# Copy the book in HDFS under a new directory
$ hdfs dfs -mkdir /mn_test
$ hdfs dfs -put ~/4300.txt /mn_test/

# Verify the content
$ hdfs dfs -ls /mn_test
Found 1 items
-rw-r--r--   3 hduser supergroup    1573151 2015-12-29
13:00 /mn_test/4300.txt

# Run the wordcount mapreduce program on the downloaded
book
$ hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-
```

Track through Web Consoles:

By default Hadoop HTTP web-consoles allow access without any form of authentication.

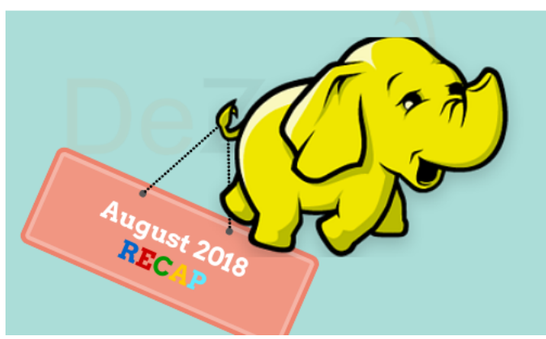
```
# NameNode
http://:50070
http://192.168.56.11:50070

# Resource Manager
http://192.168.56.11:8088

# Secondary NameNode
http://192.168.56.12:50090
```

Troubleshooting the environment:

By default, all logs for Hadoop gets stored in $HADOOP_HOME/logs. For any issue regarding installation, these logs will help to troubleshoot the cluster.

Recap of Hadoop News for September 2018

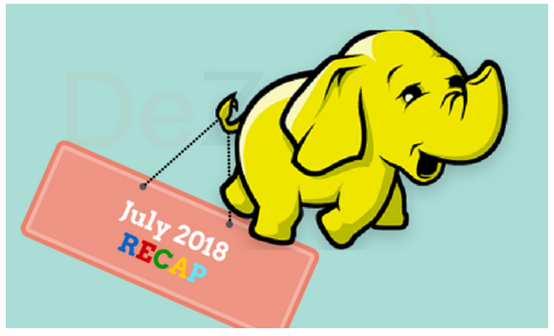
Introduction to TensorFlow for Deep Learning



Recap of Hadoop News for August 2018

AWS vs Azure-Who is the big winner in the cloud war?

Top 50 AWS Interview Questions and Answers for 2018



Recap of Hadoop News for July 2018

Other Tutorials

Big Data and Hadoop Training Courses in Popular Cities

- Hadoop Training in Edison
- Hadoop Training in Fremont
- Hadoop Training in San Jose
- Hadoop Training in Houston
- Hadoop Training in Virginia
- Hadoop Training in Washington