







ABOUT PRIVACY POLICY DONATE



Tech Support Says
For those who seek help in different areas of software and hardware platform.

Search

[HOME](#)

[Big Data](#) , [CentOS 7](#) , [CentOS 8](#) , [HA Cluster](#) , [Hadoop](#) , [High Availability](#) , [RHEL 7](#) , [RHEL 8](#)

Set Up a Highly Available Multi-node Hadoop Cluster on CentOS/RHEL 7/8

Muhammad Anwar March 10, 2020

This tutorial will show you how to set up a highly available multi-node hadoop cluster on CentOS/RHEL server. Please note that, this guide is specifically written for CentOS and RHEL release 7 and 8.

Prerequisites

To follow this tutorial, you will need three (physical or virtual) machines installed with CentOS/RHEL release 7 or 8 having sudo non-root user

Search



Video Tutorials



113

privileges.

We will use following three servers for this guide:

Name	IP	Purpose
master-node	192.168.10.1	Master Node
worker-node1	192.168.10.2	Worker Node1
worker-node2	192.168.10.3	Worker Node2

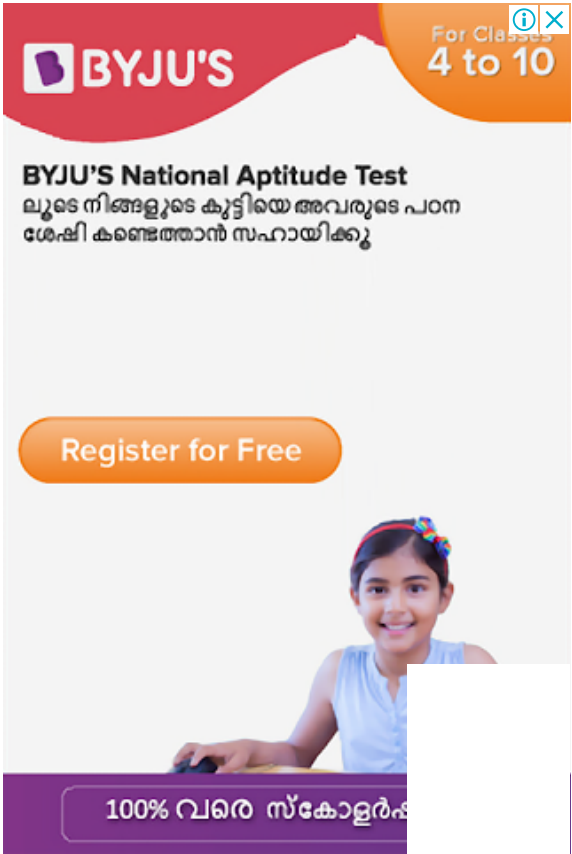
You should set hostname on each node using the below command but make sure you replace highlighted text with yours:

```
sudo hostnamectl set-hostname master-node

sudo hostnamectl set-hostname worker-node1
sudo hostnamectl set-hostname worker-node2
```

Also, set the correct timezone on each node using the below command:

```
sudo timedatectl set-timezone Asia/Karachi
```



Update Hosts File

For each node to communicate with each other by name, you need to map the ip addresses against their name.

Edit the `/etc/hosts` file on each node:

```
sudo vi /etc/hosts
```

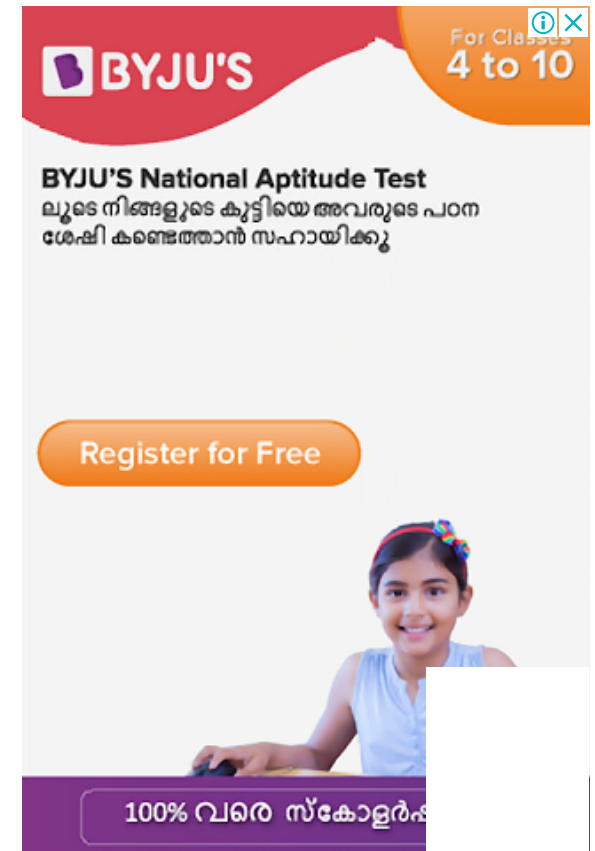
Remove everything, add your nodes ip and name like below:

```
192.168.10.1    master-node
192.168.10.2    worker-node1
192.168.10.3    worker-node2
```

Save and close file when you are finished.

Adding Hadoop User

You need a user with sudo privileges for hadoop installation and



configuration on each node.

Type below command to create a user called hadoop:

```
sudo adduser -m hadoop -G wheel
```

Set hadoop user password with below command:

```
sudo passwd hadoop
```

This will prompt you for new password and confirm password.

Make sure you create the same user on each node before moving to next step.

SSH Key-Pair Authentication

The master node in hadoop cluster will use an SSH connection to connect to other nodes with key-pair authentication to actively manage the cluster. For this, we need to set up key-pair ssh authentication on each node.

Login to your master-node as the hadoop user, and generate an SSH key like below:

```
ssh-keygen
```

This will prompt you for passphrase, make sure you leave the fields blank.

Repeat the same step on each worker node as the hadoop user. When you are finished generating ssh key-pair on all nodes, move to next step.



Dedicated Server Hosting from eWebGuru

- ✓ Fully Managed
- ✓ Tier 4 Datacenter
- ✓ Premium Bandwidth
- ✓ Premium Hardware
- ✓ India Location

www.ewebguru.com

TRY NOW



Information icon (i) and Close icon (X) are visible in the top right corner of the advertisement.

Now you need to copy `id_rsa.pub` contents to `authorized_keys` file and then transfer `authorized_keys` to remote node like below:

```
ssh-copy-id -i ~/.ssh/id_rsa.pub localhost  
scp ~/.ssh/authorized_keys worker-node1:~/.ssh/
```

Next, login to worker-node1 as the hadoop user, copy `id_rsa.pub` contents to `authorized_keys` and then transfer to remove node like below:

```
ssh-copy-id -i ~/.ssh/id_rsa.pub localhost  
  
scp ~/.ssh/authorized_keys worker-node2:~/.ssh/
```

Next, login to worker-node2, copy `id_rsa.pub` contents to `authorized_keys` and then transfer to remote node like below:

```
ssh-copy-id -i ~/.ssh/id_rsa.pub localhost  
scp ~/.ssh/id_rsa.pub master-node:~/.ssh/  
scp ~/.ssh/id_rsa.pub worker-node1:~/.ssh/
```

If everything setup correctly as described, you will be able to connect to each other node via ssh with key-pair authentication without providing password.

Installing Java

Hadoop comes with code and scripts that need java to run, you can install latest version of java on each node with below command:

```
sudo dnf -y install java-latest-openjdk java-latest-openjdk-devel
```

If you are on CentOS/RHEL 7, install latest java with yum package manager:

```
sudo yum -y install java-latest-openjdk java-latest-openjdk-devel
```

Set Java Home Environment

Hadoop comes with code and configuration that references the JAVA_HOME environment variable. This variable points to the java binary file, allowing them to run java code.

You can set up JAVA_HOME variable on each node like below:

```
echo "JAVA_HOME=$(which java)" | sudo tee -a /etc/environment
```

Reload your system's environment variables with below command:

```
source /etc/environment
```

Verify that variable was set correctly:

```
echo $JAVA_HOME
```

This should return the path to the java binary. Make sure you repeat the same step on each worker node as well.

You need to manually set hadoop binaries location into system path so that default environment understand where to look for hadoop commands.

edit /home/hadoop/.bashrc like below:

```
vi /home/hadoop/.bashrc
```

add following lines at the end of the file:

```
export HADOOP_HOME=/home/hadoop/hadoop  
export PATH=${PATH}:${HADOOP_HOME}/bin:${HADOOP_HOME}/sbin
```


Save and close.

Next, edit /home/hadoop/.bash_profile:

```
vi ~/.bash_profile
```

add following line at the end of the file:

```
PATH=/home/hadoop/hadoop/bin:/home/hadoop/hadoop/sbin:$PATH
```

Save and close file.

Make sure you repeat the same step on each worker node as well.

Download Hadoop

At the time of writing this article, hadoop 3.1.3 was the most latest available release.

Login to master-node as the hadoop user, download the Hadoop tarball file, and unzip it:

```
cd ~  
wget http://apache.cs.utah.edu/hadoop/common/current/hadoop-3.1.3.t  
tar -xzf hadoop-3.1.3.tar.gz  
mv hadoop-3.1.3 hadoop
```

Configure Hadoop

At this stage, we'll configure hadoop on master-node first, then replicate the configuration to worker nodes later.

On master-node, type below command to find java installation path:

```
update-alternatives --display java
```

Take the value of the (link currently points to) and remove the trailing /bin/java. For example on CentOS or RHEL, the link is /usr/lib/jvm/java-11-openjdk-11.0.5.10-2.el8_1.x86_64/bin/java, so JAVA_HOME should be /usr/lib/jvm/java-11-openjdk-11.0.5.10-2.el8_1.x86_64.

Edit hadoop-env.sh like below:

```
cd ~  
vi hadoop/etc/hadoop/hadoop-env.sh
```

Uncomment by removing # and update **JAVA_HOME** line like below:

```
export JAVA_HOME=/usr/lib/jvm/java-11-openjdk-amd64
```

Save and close when you are finished.

Next, edit core-site.xml file to set the NameNode location to master-node on port 9000:

```
vi hadoop/etc/hadoop/core-site.xml
```

add the following code, make sure you replace master-node with yours:

```
<?xml version="1.0" encoding="UTF-8"?>  
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>  
<!--  
    Licensed under the Apache License, Version 2.0 (the "License");  
    you may not use this file except in compliance with the License.  
    You may obtain a copy of the License at
```

```
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>fs.default.name</name>
```

```
<value>hdfs://master-node:9000</value>
```

```
</property>
```

```
</configuration>
```

Save and close.

Next, edit `hdfs-site.conf` to resemble the following configuration:

```
vi hadoop/etc/hadoop/hdfs-site.xml
```

add following code:

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<!--
  Licensed under the Apache License, Version 2.0 (the "License");
  you may not use this file except in compliance with the License.
  You may obtain a copy of the License at

    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.namenode.name.dir</name>
    <value>/home/hadoop/data/nameNode</value>
</property>
```

```
<property>
    <name>dfs.datanode.data.dir</name>
    <value>/home/hadoop/data/dataNode</value>
</property>
<property>
    <name>dfs.replication</name>
    <value>2</value>
</property>
</configuration>
```

Note that the last property string `dfs.replication`, indicates how many times data is replicated in the cluster. We set **2** to have all the data duplicated on the two of our worker nodes. If you have only one worker node, enter 1, if you have three, enter 3 but don't enter a value higher than the actual number of worker nodes you have.

Save and close file when you are finished.

Next, edit the `mapred-site.xml` file, setting YARN as the default framework for MapReduce operations:

```
vi hadoop/etc/hadoop/mapred-site.xml
```

add following code:

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<!--
  Licensed under the Apache License, Version 2.0 (the "License");
  you may not use this file except in compliance with the License.
  You may obtain a copy of the License at

    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>mapreduce.framework.name</name>
    <value>yarn</value>
</property>
<property>
```

```
        <name>yarn.app.mapreduce.am.env</name>
        <value>HADOOP_MAPRED_HOME=$HADOOP_HOME</value>
    </property>
    <property>
        <name>mapreduce.map.env</name>
        <value>HADOOP_MAPRED_HOME=$HADOOP_HOME</value>
    </property>
    <property>
        <name>mapreduce.reduce.env</name>
        <value>HADOOP_MAPRED_HOME=$HADOOP_HOME</value>
    </property>
    <property>
        <name>yarn.app.mapreduce.am.resource.mb</name>
        <value>512</value>
    </property>
    <property>
        <name>mapreduce.map.memory.mb</name>
        <value>256</value>
    </property>
    <property>
        <name>mapreduce.reduce.memory.mb</name>
        <value>256</value>
    </property>
</configuration>
```

Save and close.

Next, edit `yarn-site.xml`, which contains the configuration options for YARN.

```
vi hadoop/etc/hadoop/yarn-site.xml
```

add below code, make sure you replace 192.168.10.1 with the your master-node's ip address:

```
<?xml version="1.0"?>
<!--
Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

    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.
-->
<configuration>
```

```
<!-- Site specific YARN configuration properties -->
<property>
    <name>yarn.acl.enable</name>
    <value>0</value>
</property>
<property>
    <name>yarn.resourcemanager.hostname</name>
    <value>192.168.10.1</value>
</property>
<property>
    <name>yarn.nodemanager.aux-services</name>
    <value>mapreduce_shuffle</value>
</property>
<property>
    <name>yarn.nodemanager.resource.memory-mb</name>
    <value>2048</value>
</property>
<property>
    <name>yarn.scheduler.maximum-allocation-mb</name>
    <value>2048</value>
</property>
<property>
    <name>yarn.scheduler.minimum-allocation-mb</name>
    <value>1024</value>
</property>
<property>
```

```
<name>yarn.nodemanager.vmem-check-enabled</name>  
<value>>false</value>  
</property>  
</configuration>
```

The last property disables virtual-memory checking which can prevent containers from being allocated properly with openjdk if enabled.

Note: Memory allocation can be tricky on low RAM nodes because default values are not suitable for nodes with less than 8GB of RAM. We have manually set memory allocation for MapReduce jobs, and provide a sample configuration for 4GB RAM nodes.

Save and close.

Next, edit workers file to include both of the worker nodes (worker-node1, worker-node2) in our case:

```
vi hadoop/etc/hadoop/workers
```

Remove **localhost** if exists, add your worker nodes like below:

```
worker-node1  
worker-node2
```

Save and close.

The workers file is used by hadoop startup scripts to start required daemons on all nodes.

At this stage, we have completed hadoop configuration on master-node. In the next step we will duplicate hadoop configuration on worker nodes.

Configure Worker Nodes

This section will show you how to duplicate hadoop configuration from master-node to all work nodes.

First copy the hadoop tarball file from master-node to worker nodes like below:

```
cd ~  
scp hadoop-*.tar.gz worker-node1:/home/hadoop/  
scp hadoop-*.tar.gz worker-node2:/home/hadoop/
```

Next, login to each worker node as the hadoop user via SSH and unzip the hadoop archive, rename the directory then exit from worker nodes to get

back on the master-node:

```
ssh worker-node1

tar -xzf hadoop-3.1.2.tar.gz
mv hadoop-3.1.3 hadoop
exit
```

Repeat the same step on worker-node2.

From the master-node, duplicate the Hadoop configuration files to all worker nodes using command below:

```
for node in worker-node1 worker-node2; do

scp ~/hadoop/etc/hadoop/* $node:/home/hadoop/hadoop/etc/hadoop/;
done
```

Make sure you replace worker-node1, worker-node2 with your worker nodes name.

Next, on master-node as the hadoop user, type the below command to

format hadoop file system:

```
hdfs namenode -format
```

You will see the output similar to the following which says hadoop cluster is ready to run.

```
WARNING: /home/hadoop/hadoop/logs does not exist. Creating.
2020-03-09 11:38:04,791 INFO namenode.NameNode: STARTUP_MSG:
/*****
STARTUP_MSG: Starting NameNode
STARTUP_MSG:   host = master-node/192.168.10.1
STARTUP_MSG:   args = [-format]
STARTUP_MSG:   version = 3.1.3
STARTUP_MSG:   build = https://gitbox.apache.org/repos/asf/hadoop.g
STARTUP_MSG:   java = 11.0.5
*****/
2020-03-09 11:38:04,818 INFO namenode.NameNode: registered UNIX sig
2020-03-09 11:38:05,062 INFO namenode.NameNode: createNameNode [-fo
2020-03-09 11:38:06,162 INFO common.Util: Assuming 'file' scheme fo
2020-03-09 11:38:06,163 INFO common.Util: Assuming 'file' scheme fo
Formatting using clusterid: CID-e791ed9f-f86f-4a19-bbf4-aaa06c9c323
2020-03-09 11:38:06,233 INFO namenode.FSEditLog: Edit logging is as
2020-03-09 11:38:06,275 INFO namenode.FSNamesystem: KeyProvider: nu
2020-03-09 11:38:06,276 INFO namenode.FSNamesystem: fsLock is fair:
```

```
2020-03-09 11:38:06,277 INFO namenode.FSNamesystem: Detailed lock h
2020-03-09 11:38:06,387 INFO namenode.FSNamesystem: fsOwner
2020-03-09 11:38:06,387 INFO namenode.FSNamesystem: supergroup
2020-03-09 11:38:06,387 INFO namenode.FSNamesystem: isPermissionEna
2020-03-09 11:38:06,387 INFO namenode.FSNamesystem: HA Enabled: fal
2020-03-09 11:38:06,476 INFO common.Util: dfs.datanode.fileio.profi
2020-03-09 11:38:06,510 INFO blockmanagement.DatanodeManager: dfs.b
2020-03-09 11:38:06,510 INFO blockmanagement.DatanodeManager: dfs.r
2020-03-09 11:38:06,516 INFO blockmanagement.BlockManager: dfs.name
2020-03-09 11:38:06,516 INFO blockmanagement.BlockManager: The bloc
2020-03-09 11:38:06,518 INFO util.GSet: Computing capacity for map
2020-03-09 11:38:06,518 INFO util.GSet: VM type          = 64-bit
2020-03-09 11:38:06,530 INFO util.GSet: 2.0% max memory 908.7 MB =
2020-03-09 11:38:06,530 INFO util.GSet: capacity        = 2^21 = 2097
2020-03-09 11:38:06,537 INFO blockmanagement.BlockManager: dfs.bloc
2020-03-09 11:38:06,557 INFO Configuration.deprecation: No unit for
2020-03-09 11:38:06,557 INFO blockmanagement.BlockManagerSafeMode:
2020-03-09 11:38:06,557 INFO blockmanagement.BlockManagerSafeMode:
2020-03-09 11:38:06,557 INFO blockmanagement.BlockManagerSafeMode:
2020-03-09 11:38:06,558 INFO blockmanagement.BlockManager: defaultR
2020-03-09 11:38:06,558 INFO blockmanagement.BlockManager: maxRepli
2020-03-09 11:38:06,558 INFO blockmanagement.BlockManager: minRepli
2020-03-09 11:38:06,558 INFO blockmanagement.BlockManager: maxRepli
2020-03-09 11:38:06,558 INFO blockmanagement.BlockManager: redundan
2020-03-09 11:38:06,558 INFO blockmanagement.BlockManager: encryptD
2020-03-09 11:38:06,559 INFO blockmanagement.BlockManager: maxNumBL
```

```
2020-03-09 11:38:06,602 INFO namenode.FSDirectory: GLOBAL serial ma
2020-03-09 11:38:06,669 INFO util.GSet: Computing capacity for map
2020-03-09 11:38:06,669 INFO util.GSet: VM type          = 64-bit
2020-03-09 11:38:06,669 INFO util.GSet: 1.0% max memory 908.7 MB =
2020-03-09 11:38:06,669 INFO util.GSet: capacity        = 2^20 = 1048
2020-03-09 11:38:06,670 INFO namenode.FSDirectory: ACLs enabled? fa
2020-03-09 11:38:06,670 INFO namenode.FSDirectory: POSIX ACL inheri
2020-03-09 11:38:06,670 INFO namenode.FSDirectory: XAttrs enabled?
2020-03-09 11:38:06,670 INFO namenode.NameNode: Caching file names
2020-03-09 11:38:06,679 INFO snapshot.SnapshotManager: Loaded confi
2020-03-09 11:38:06,681 INFO snapshot.SnapshotManager: SkipList is
2020-03-09 11:38:06,685 INFO util.GSet: Computing capacity for map
2020-03-09 11:38:06,685 INFO util.GSet: VM type          = 64-bit
2020-03-09 11:38:06,686 INFO util.GSet: 0.25% max memory 908.7 MB =
2020-03-09 11:38:06,686 INFO util.GSet: capacity        = 2^18 = 2621
2020-03-09 11:38:06,697 INFO metrics.TopMetrics: NNTop conf: dfs.na
2020-03-09 11:38:06,697 INFO metrics.TopMetrics: NNTop conf: dfs.na
2020-03-09 11:38:06,697 INFO metrics.TopMetrics: NNTop conf: dfs.na
2020-03-09 11:38:06,700 INFO namenode.FSNamesystem: Retry cache on
2020-03-09 11:38:06,701 INFO namenode.FSNamesystem: Retry cache wil
2020-03-09 11:38:06,707 INFO util.GSet: Computing capacity for map
2020-03-09 11:38:06,707 INFO util.GSet: VM type          = 64-bit
2020-03-09 11:38:06,708 INFO util.GSet: 0.029999999329447746% max m
2020-03-09 11:38:06,708 INFO util.GSet: capacity        = 2^15 = 3276
2020-03-09 11:38:06,760 INFO namenode.FSImage: Allocated new BlockF
2020-03-09 11:38:06,787 INFO common.Storage: Storage directory /hom
```



```
2020-03-09 11:38:06,862 INFO namenode.FSImageFormatProtobuf: Saving
2020-03-09 11:38:07,029 INFO namenode.FSImageFormatProtobuf: Image
2020-03-09 11:38:07,045 INFO namenode.NNStorageRetentionManager: Go
2020-03-09 11:38:07,072 INFO namenode.FSImage: FSImageSaver clean c
2020-03-09 11:38:07,074 INFO namenode.NameNode: SHUTDOWN_MSG:
/*****
SHUTDOWN_MSG: Shutting down NameNode at master-node/192.168.10.1
*****/
```

With this hdfs format, your hadoop installation is now configured and ready to run.

Running Hadoop

Login to master-node as the hadoop user and start the hadoop cluster by running the below command:

```
start-dfs.sh
```

You will see similar to the following output:

```
start-dfs.sh
```

You will see similar to the following output:

```
Starting namenodes on [master-node]
Starting datanodes
worker-node2: WARNING: /home/hadoop/hadoop/logs does not exist. Cre
worker-node1: WARNING: /home/hadoop/hadoop/logs does not exist. Cre
Starting secondary namenodes [master-node]
```

This will start NameNode and SecondaryNameNode component on master-node, and DataNode on worker-node1 and worker-node2, according to the configuration in the workers config file.

Check that every process is running with the **jps** command on each node.

On master-node, type **jps** and you should see the following:

```
8066 NameNode
8292 SecondaryNameNode
8412 Jps
```

On worker-node1 and worker-node2, type **jps** and you should see the following:

```
17525 DataNode  
17613 Jps
```

You can get useful information about your hadoop cluster with the below command.

```
hdfs dfsadmin -report
```

This will print information (e.g., capacity and usage) for all running nodes in the cluster.

Next, open up your preferred web browser and navigate to **http://your_master_node_IP:9870**, and you'll get a user-friendly hadoop monitoring web console like below:

HadoopOverviewDatanodesDatanode Volume FailuresSnapshotStartup ProgressUtilities

Overview 'master-node:9000' (active)

Started:	Thu Mar 05 21:22:01 +0500 2020
Version:	3.1.3, rba631c436b806728f8ec2f54ab1e289526c90579
Compiled:	Thu Sep 12 07:47:00 +0500 2019 by ztang from branch-3.1.3
Cluster ID:	CID-29c1bd40-7f6d-4aff-8496-817be453fbd2
Block Pool ID:	BP-412023535-192.168.10.1-1583425299012

Summary

Security is off.

Safemode is off.

7 files and directories, 3 blocks (3 replicated blocks, 0 erasure coded block groups) = 10 total filesystem object(s).

Heap Memory used 35.73 MB of 61.16 MB Heap Memory. Max Heap Memory is 477.56 MB.

Non Heap Memory used 63.11 MB of 66.19 MB Committed Non Heap Memory. Max Non Heap Memory is <unbounded>.

Configured Capacity:	78.24 GB
Configured Remote Capacity:	0 B
DFS Used:	2.47 MB (0%)
Non DFS Used:	11.74 GB
DFS Remaining:	62.46 GB (79.84%)
Block Pool Used:	2.47 MB (0%)
DataNodes usages% (Min/Median/Max/stdDev):	0.00% / 0.00% / 0.00% / 0.00%
Live Nodes	2 (Decommissioned: 0, In Maintenance: 0)
Dead Nodes	0 (Decommissioned: 0, In Maintenance: 0)
Decommissioning Nodes	0
Entering Maintenance Nodes	0
Total Datanode Volume Failures	0 (0 B)
Number of Under-Replicated Blocks	0
Number of Blocks Pending Deletion (including replicas)	0
Block Deletion Start Time	Thu Mar 05 21:22:01 +0500 2020
Last Checkpoint Time	Fri Mar 06 11:42:49 +0500 2020

NameNode Journal Status

Current transaction ID: 30

Journal Manager	State
FileJournalManager(root=/home/hadoop/data/nameNode)	EditLogFileOutputStream(/home/hadoop/data/nameNode/current/edits_inprogress_000000000000000030)

Testing Hadoop Cluster

You can test your hadoop cluster by writing and reading some contents using `hdfs dfs` command.

First, manually create your home directory. All other commands will use a path relative to this default home directory:

On master-node, type below command:

```
hdfs dfs -mkdir -p /user/hadoop
```

We'll use few textbooks from the Gutenberg project as an example for this guide.

Create a books directory in hadoop file-system. The following command will create it in the home directory, `/user/hadoop/books`:

```
hdfs dfs -mkdir books
```

Now download a few books from the Gutenberg project:

```
cd /home/hadoop

wget -O franklin.txt http://www.gutenberg.org/files/13482/13482.txt
wget -O herbert.txt http://www.gutenberg.org/files/20220/20220.txt
wget -O maria.txt http://www.gutenberg.org/files/29635/29635.txt
```

Next, put these three books using hdfs, in the books directory:

```
hdfs dfs -put franklin.txt herbert.txt maria.txt books
```

List the contents of the books directory:

```
hdfs dfs -ls books
```

Next, move one of the books to the local filesystem:

```
hdfs dfs -get books/franklin.txt
```

You can also directly print the books on terminal from hdfs:

```
hdfs dfs -cat books/maria.txt
```

These are just few example of hadoop commands. However, there are many commands to manage your hdfs. For a complete list, you can look at the Apache hdfs shell documentation, or print help with:

```
hdfs dfs -help
```

Start YARN

HDFS is a distributed storage system, and doesn't provide any services for running and scheduling tasks in the cluster. This is the role of the YARN framework. The following section is about starting, monitoring, and submitting jobs to YARN.

On master-node, you can start YARN with the below script:

```
start-yarn.sh
```

You will see the output like below:

```
Starting resourcemanager  
Starting nodemanagers
```

Check that everything is running with the `jps` command. In addition to the previous HDFS daemon, you should see a ResourceManager on master-node, and a NodeManager on worker-node1 and worker-node2.

To stop YARN, run the following command on master-node:

```
stop-yarn.sh
```

Similarly, you can get a list of running applications with below command:

```
yarn application -list
```

To get all available parameters of the yarn command, see Apache YARN documentation.

As with HDFS, YARN provides a friendlier web UI, started by default on port 8088 of the Resource Manager. You can navigate to **`http://master-node-IP:8088`** to browse the YARN web console:

Submit MapReduce Jobs to YARN

YARN jobs are packaged into jar files and submitted to YARN for execution with the command `yarn jar`. The Hadoop installation package provides sample applications that can be run to test your cluster. You'll use them to run a word count on the three books previously uploaded to HDFS.

On master-node, submit a job with the sample jar to YARN:

```
yarn jar ~/hadoop/share/hadoop/mapreduce/hadoop-mapreduce-examples-
```

The last argument is where the output of the job will be saved - in HDFS.

After the job is finished, you can get the result by querying with below command:

```
hdfs dfs -ls output
```

Print the result with:

```
hdfs dfs -cat output/part-r-00000 | less
```

Wrapping up

Now that you have a YARN cluster up and running, you can learn how to code your own YARN jobs with [Apache documentation](#) and install Spark on top of your YARN cluster. You may wish to take the following resources into consideration for additional information on this topic.

Share: [f](#) [t](#) [G+](#) [p](#)

Related Posts:



[← Newer Post](#)[Home](#)[Older Post →](#)


[How To Set Up Hadoop in Stand-...](#)[Set Up a Highly Available ...](#)[Set Up a Highly Available ...](#)

0 comments:

Post a Comment

Comments with links will not be published.

Enter your comment...

 Comment as:

Sign out

Publish

Preview

☐ Notify me



Ad home.kpmg

Ad home.kpmg

Copyright © 2020 Tech Support Says