

In this blog, I am going to talk about HDFS 2.x High Availability Cluster Architecture and the procedure to set up an HDFS High Availability cluster. The order in which the topics have been covered in this blog are as follows:

- ### Introduction:

**NameNode Availability:**

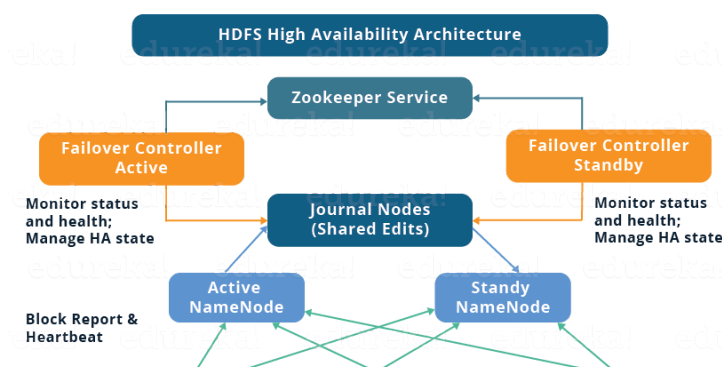
The reasons for unavailability of NameNode can be:

- In either of the above cases, we have a downtime where we are not able to use the HDFS cluster which becomes a challenge.

Let us understand that how HDFS HA Architecture solved this critical problem of NameNode availability:

The HA architecture solved this problem of NameNode availability by allowing us to have two NameNodes in an active/passive configuration. So, we have two running NameNodes at the same time in a High Availability cluster:

- Active NameNode
- Standby/Passive NameNode.



DataNode

DataNode

DataNode

If one NameNode goes down, the other NameNode can take over the responsibility and therefore, reduce the cluster down time. The standby NameNode serves the purpose of a backup NameNode (unlike the Secondary NameNode) which incorporate failover capabilities to the Hadoop cluster. Therefore, with the StandbyNode, we can have automatic failover whenever a NameNode crashes (unplanned event) or we can have a graceful (manually initiated) failover during the maintenance period.

There are two issues in maintaining consistency in the HDFS High Availability cluster:

- Active and Standby NameNode should always be in sync with each other, i.e. They should have the same metadata. This will allow us to restore the Hadoop cluster to the same namespace state where it got crashed and therefore, will provide us to have fast failover.
- There should be only one active NameNode at a time because two active NameNode will lead to corruption of the data. *This kind of scenario is termed as a split-brain scenario where a cluster gets divided into smaller cluster, each one believing that it is the only active cluster.* To avoid such scenarios fencing is done. Fencing is a process of ensuring that only one NameNode remains active at a particular time.

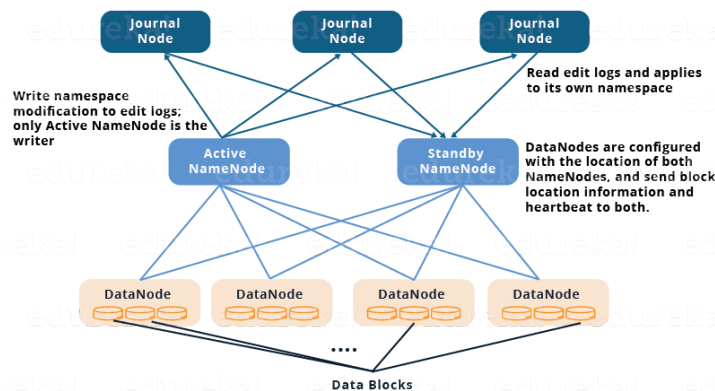
### Implementation of HA Architecture:

Now, you know that in HDFS HA Architecture, we have two NameNodes running at the same time. So, we can implement the Active and Standby NameNode configuration in following two ways:

1. Using Quorum Journal Nodes
2. Shared Storage using NFS

Let us understand these two ways of implementation taking one at a time:

#### 1. Using Quorum Journal Nodes:



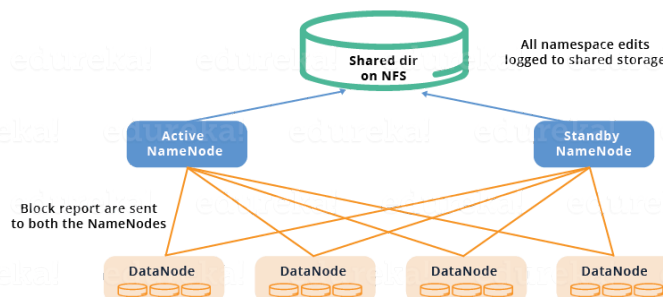
- The standby NameNode and the active NameNode keep in sync with each other through a separate group of nodes or daemons -called **JournalNodes**. The JournalNodes follows the ring topology where the nodes are connected to each other to form a ring. The JournalNode serves the request coming to it and copies the information into other nodes in the ring. This provides fault tolerance in case of JournalNode failure.
- The active NameNode is responsible for updating the EditLogs (metadata information) present in the JournalNodes.
- The StandbyNode reads the changes made to the EditLogs in the JournalNode and applies it to its own namespace in a constant manner.
- During failover, the StandbyNode makes sure that it has updated its meta data information from the JournalNodes before becoming the new Active NameNode. This makes the current namespace state synchronized with the state before failover.
- The IP Addresses of both the NameNodes are available to all the DataNodes and they send their heartbeats and block location information to both the NameNode. This provides a fast failover (less down time) as the StandbyNode has an updated information about the block location in the cluster.

### Fencing of NameNode:

Now, as discussed earlier, it is very important to ensure that there is only one Active NameNode at a time. So, fencing is a process to ensure this very property in a cluster.

- The JournalNodes performs this fencing by allowing only one NameNode to be the writer at a time.
- The Standby NameNode takes over the responsibility of writing to the JournalNodes and forbid any other NameNode to remain active.
- Finally, the new Active NameNode can perform its activities safely.

#### 2. Using Shared Storage:



- The StandbyNode and the active NameNode keep in sync with each other by using a **shared storage device**. The active NameNode logs the record of any modification done in its namespace to an EditLog present in this shared storage. The StandbyNode reads the changes made to the EditLogs in this shared storage

and applies it to its own namespace.

- Now, in case of failover, the StandbyNode updates its metadata information using the EditLogs in the shared storage at first. Then, it takes the responsibility of the Active NameNode. This makes the current namespace state synchronized with the state before failover.
- The administrator must configure at least one fencing method to avoid a split-brain scenario.
- The system may employ a range of fencing mechanisms. It may include killing of the NameNode's process and revoking its access to the shared storage directory.
- As a last resort, we can fence the previously active NameNode with a technique known as STONITH, or "shoot the other node in the head". STONITH uses a specialized power distribution unit to forcibly power down the NameNode machine.

### Automatic Failover:

Failover is a procedure by which a system automatically transfers control to secondary system when it detects a fault or failure. There are two types of failover:

**Graceful Failover:** In this case, we manually initiate the failover for routine maintenance.

**Automatic Failover:** In this case, the failover is initiated automatically in case of NameNode failure (unplanned event).

Apache Zookeeper is a service that provides the automatic failover capability in HDFS High Availability cluster. It maintains small amounts of coordination data, informs clients of changes in that data, and monitors clients for failures. Zookeeper maintains a session with the NameNodes. In case of failure, the session will expire and the Zookeeper will inform other NameNodes to initiate the failover process. In case of NameNode failure, other passive NameNode can take a lock in Zookeeper stating that it wants to become the next Active NameNode.

The ZookeeperFailoverController (ZKFC) is a Zookeeper client that also monitors and manages the NameNode status. Each of the NameNode runs a ZKFC also. ZKFC is responsible for monitoring the health of the NameNodes periodically.

Now that you have understood what is High Availability in a Hadoop cluster, it's time to set it up. To set up High Availability in Hadoop cluster you have to use Zookeeper in all the nodes.

The daemons in Active NameNode are:

- Zookeeper
- Zookeeper Fail Over controller
- JournalNode
- NameNode

The daemons in Standby NameNode are:

- Zookeeper
- Zookeeper Fail Over controller
- JournalNode
- NameNode

The daemons in DataNode are:

- Zookeeper
- JournalNode
- DataNode

If you wish to master HDFS and Hadoop, check out the specially curated Big Data and Hadoop course by Edureka. Click on the button below to get started.

[Become a HDFS Expert!](https://www.edureka.co/big-data-and-hadoop)

(<https://www.edureka.co/big-data-and-hadoop>)

## Setting Up and Configuring High Availability Cluster in Hadoop:

You have to first set up the Java and host names of each node.

Virtual machine	IP address	Host name
Active NameNode	192.168.1.81	nn1.cluster.com or nn1
Standby NameNode	192.168.1.58	nn2.cluster.com or nn2
DataNode	192.168.1.82	dn1.cluster.com or dn1

Download the Hadoop and Zookeeper binary tar file, extract the files to edit configuration files.

**Command :** `wget https://archive.apache.org/dist/zookeeper/zookeeper-3.4.6/zookeeper-3.4.6.tar.gz`

```

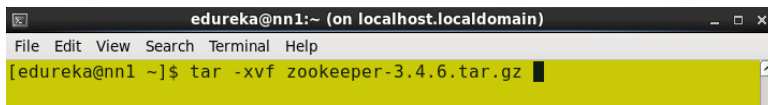
edureka@nn1:~ (on localhost.localdomain)
File Edit View Search Terminal Help
[edureka@nn1 ~]$ wget https://archive.apache.org/dist/zookeeper/zookeeper-3.4.6/zookeeper-3.4.6.tar.gz
--2016-11-25 16:23:49-- https://archive.apache.org/dist/zookeeper/zookeeper-3.4.6/zookeeper-3.4.6.tar.gz
Resolving archive.apache.org... 163.172.17.199
Connecting to archive.apache.org|163.172.17.199|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 17699306 (17M) [application/x-gzip]
Saving to: "zookeeper-3.4.6.tar.gz.1"

6% [>] 1,081,344 282K/s eta 67s

```

Untar the zookeeper-3.4.6.tar.gz

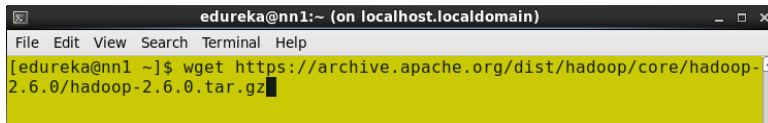
**Command:** `tar -xvf zookeeper-3.4.6.tar.gz`



```
edureka@nn1:~ (on localhost.localdomain)
File Edit View Search Terminal Help
[edureka@nn1 ~]$ tar -xvf zookeeper-3.4.6.tar.gz
```

Download the stable Hadoop binary tar to from Apache Hadoop site.


**Command:** `wget https://archive.apache.org/dist/hadoop/core/hadoop-2.6.0/hadoop-2.6.0.tar.gz`



```
edureka@nn1:~ (on localhost.localdomain)
File Edit View Search Terminal Help
[edureka@nn1 ~]$ wget https://archive.apache.org/dist/hadoop/core/hadoop-2.6.0/hadoop-2.6.0.tar.gz
```

Extract the Hadoop tar ball.

**Command:** `tar -xvf hadoop-2.6.0.tar.gz`



```
edureka@nn1:~
File Edit View Search Terminal Help
[edureka@nn1 ~]$ tar -xvf hadoop-2.6.0.tar.gz
```

([https://d1jnx9ba8s6j9r.cloudfront.net/blog/wp-content/uploads/2015/06/untar\\_Hadoop\\_binary.png](https://d1jnx9ba8s6j9r.cloudfront.net/blog/wp-content/uploads/2015/06/untar_Hadoop_binary.png))  
Untar hadoop binary.

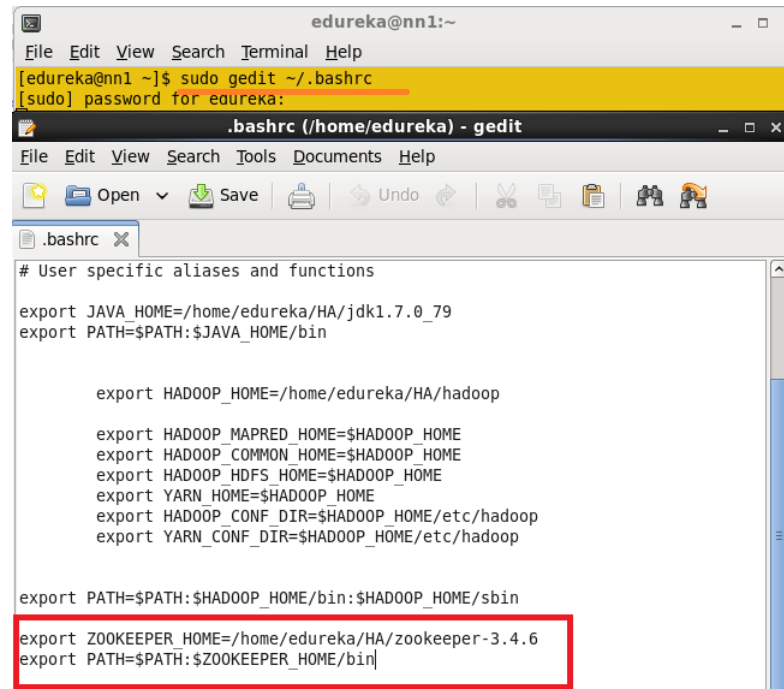
Add the Hadoop, Zookeeper and paths to .bashrc file.

Open the .bashrc file.

**Command:** `sudo gedit ~/.bashrc`

Add the below paths:

```
export HADOOP_HOME=< Path to your Hadoop-2.6.0 directory>
export HADOOP_MAPRED_HOME=$HADOOP_HOME
export HADOOP_COMMON_HOME=$HADOOP_HOME
export HADOOP_HDFS_HOME=$HADOOP_HOME
export YARN_HOME=$HADOOP_HOME
export HADOOP_CONF_DIR=$HADOOP_HOME/etc/hadoop
export YARN_CONF_DIR=$HADOOP_HOME/etc/hadoop
export JAVA_HOME=<Path to your Java Directory>
export ZOOKEEPER_HOME = <Path to your Zookeeper Directory>
export PATH=$PATH: $JAVA_HOME/bin: $HADOOP_HOME/bin: $HADOOP_HOME/sbin:$ZOOKEEPER_HOME/bin
```



```
edureka@nn1:~
File Edit View Search Terminal Help
[edureka@nn1 ~]$ sudo gedit ~/.bashrc
[sudo] password for edureka:

.bashrc (/home/edureka) - gedit
File Edit View Search Tools Documents Help
[Icons] Open Save Undo Redo Cut Copy Paste Print
.bashrc
# User specific aliases and functions

export JAVA_HOME=/home/edureka/HA/jdk1.7.0_79
export PATH=$PATH:$JAVA_HOME/bin

export HADOOP_HOME=/home/edureka/HA/hadoop

export HADOOP_MAPRED_HOME=$HADOOP_HOME
export HADOOP_COMMON_HOME=$HADOOP_HOME
export HADOOP_HDFS_HOME=$HADOOP_HOME
export YARN_HOME=$HADOOP_HOME
export HADOOP_CONF_DIR=$HADOOP_HOME/etc/hadoop
export YARN_CONF_DIR=$HADOOP_HOME/etc/hadoop

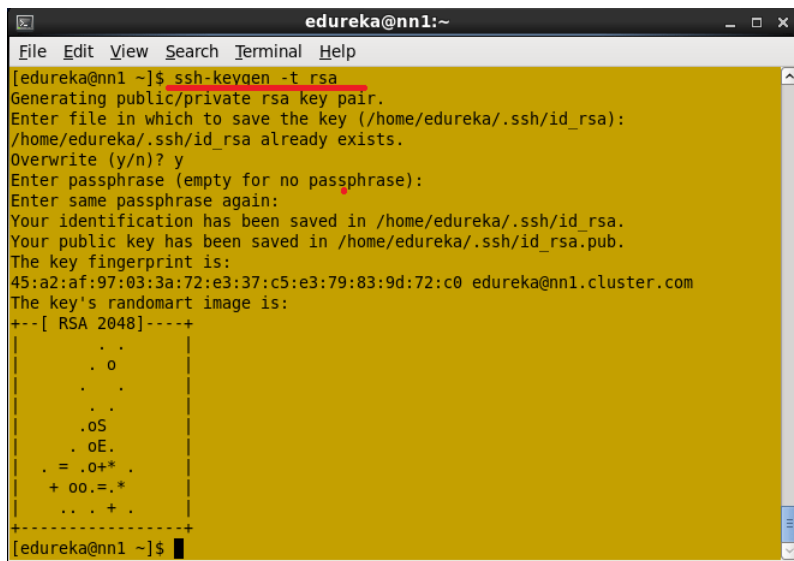
export PATH=$PATH:$HADOOP_HOME/bin:$HADOOP_HOME/sbin
export ZOOKEEPER_HOME=/home/edureka/HA/zookeeper-3.4.6
export PATH=$PATH:$ZOOKEEPER_HOME/bin
```

(<https://d1jnx9ba8s6j9r.cloudfront.net/blog/wp-content/uploads/2015/06/bashrc.png>)  
Edit .bashrc file.

Enable the SSH in all the node.

Generate the SSH key in all the nodes.

**Command:** `ssh-keygen -t rsa` (This Step in all the Nodes)



```

edureka@nn1:~
File Edit View Search Terminal Help
[edureka@nn1 ~]$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/edureka/.ssh/id_rsa):
/home/edureka/.ssh/id_rsa already exists.
Overwrite (y/n)? y
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/edureka/.ssh/id_rsa.
Your public key has been saved in /home/edureka/.ssh/id_rsa.pub.
The key fingerprint is:
45:a2:af:97:03:3a:72:e3:37:c5:e3:79:83:9d:72:c0 edureka@nn1.cluster.com
The key's randomart image is:
+--[ RSA 2048 ]-----+
|      . .              |
|      . 0              |
|      . .              |
|      . .              |
|      .OS              |
|      .OE              |
|      . = .O+*          |
|      + .O.=.*          |
|      . . +            |
+-----+
[edureka@nn1 ~]$

```

([https://d1jnx9ba8s6j9r.cloudfront.net/blog/wp-content/uploads/2015/06/ssh\\_key\\_generation.png](https://d1jnx9ba8s6j9r.cloudfront.net/blog/wp-content/uploads/2015/06/ssh_key_generation.png))

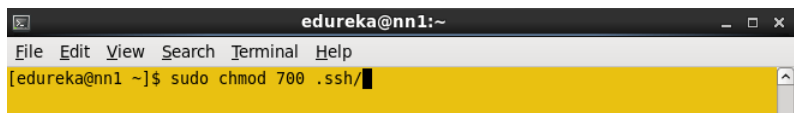
Set up SSH key in all the nodes.

Don't give any path to the Enter file to save the key and don't give any passphrase. Press enter button.

Generate the ssh key process in all the nodes.

Once ssh key is generated, you will get the public key and private key.

The .ssh key Directory should contain the Permission 700 and all the keys inside the .ssh directory should contain the permissions 600.



```

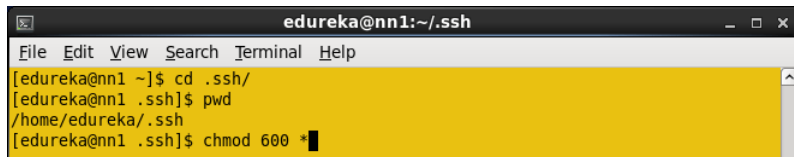
edureka@nn1:~
File Edit View Search Terminal Help
[edureka@nn1 ~]$ sudo chmod 700 .ssh/

```

(<https://d1jnx9ba8s6j9r.cloudfront.net/blog/wp-content/uploads/2015/06/ssh-permission.png>)

Change the SSH directory permission.

Change the directory to .ssh and change the permission of files to 600



```

edureka@nn1:~/.ssh
File Edit View Search Terminal Help
[edureka@nn1 ~]$ cd .ssh/
[edureka@nn1 .ssh]$ pwd
/home/edureka/.ssh
[edureka@nn1 .ssh]$ chmod 600 *

```

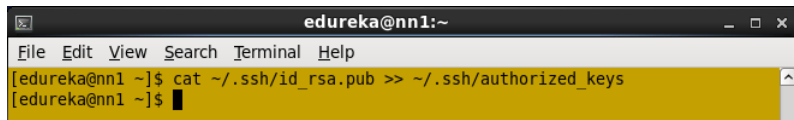
([https://d1jnx9ba8s6j9r.cloudfront.net/blog/wp-content/uploads/2015/06/ssh\\_key\\_permission.png](https://d1jnx9ba8s6j9r.cloudfront.net/blog/wp-content/uploads/2015/06/ssh_key_permission.png))

Change public and private key permission.

You have to copy the Name nodes ssh public key to all the nodes.

In Active Namenode, copy the id\_rsa.pub using cat command.

**Command:** cat ~/.ssh/id\_rsa.pub >> ~/.ssh/authorized\_keys



```

edureka@nn1:~
File Edit View Search Terminal Help
[edureka@nn1 ~]$ cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys
[edureka@nn1 ~]$

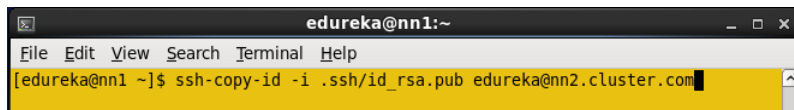
```

([https://d1jnx9ba8s6j9r.cloudfront.net/blog/wp-content/uploads/2015/06/copy\\_ssh\\_to\\_namenode.png](https://d1jnx9ba8s6j9r.cloudfront.net/blog/wp-content/uploads/2015/06/copy_ssh_to_namenode.png))

Copy Namenode ssh key to it's authorized keys.

Copy the NameNode public key to all the nodes using **ssh-copy-id** command.

**Command:** ssh-copy-id -i .ssh/id\_rsa.pub edureka@nn2.cluster.com (mailto:edureka@nn2.cluster.com)



```

edureka@nn1:~
File Edit View Search Terminal Help
[edureka@nn1 ~]$ ssh-copy-id -i .ssh/id_rsa.pub edureka@nn2.cluster.com

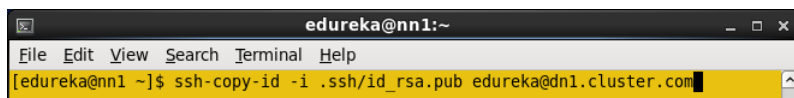
```

([https://d1jnx9ba8s6j9r.cloudfront.net/blog/wp-content/uploads/2015/06/ssh\\_copy\\_secondday.png](https://d1jnx9ba8s6j9r.cloudfront.net/blog/wp-content/uploads/2015/06/ssh_copy_secondday.png))

Copy namenode key to Standby NameNode.

Copy NameNode public key to data node.

**Command:** ssh-copy-id -i .ssh/id\_rsa.pub edureka@dn1.cluster.com (mailto:edureka@dn1.cluster.com)



```

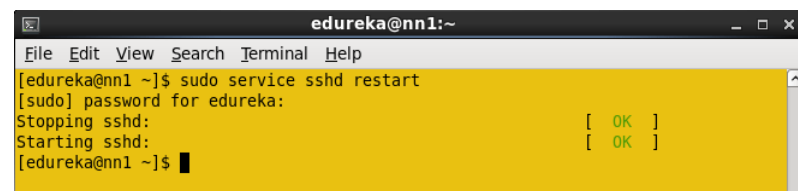
edureka@nn1:~
File Edit View Search Terminal Help
[edureka@nn1 ~]$ ssh-copy-id -i .ssh/id_rsa.pub edureka@dn1.cluster.com

```

([https://d1jnx9ba8s6j9r.cloudfront.net/blog/wp-content/uploads/2015/06/ssh\\_copy\\_datanodes.png](https://d1jnx9ba8s6j9r.cloudfront.net/blog/wp-content/uploads/2015/06/ssh_copy_datanodes.png))  
Copy Namenode public key to data node.

Restart the sshd service in all the nodes.

**Command:** sudo service sshd restart (Do in all the nodes)



```

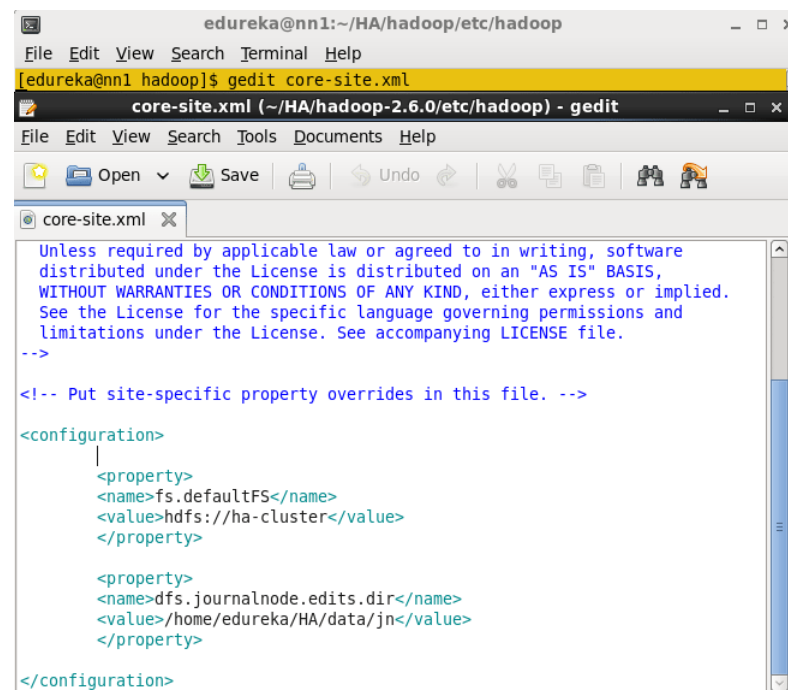
edureka@nn1:~
File Edit View Search Terminal Help
[edureka@nn1 ~]$ sudo service sshd restart
[sudo] password for edureka:
Stopping sshd: [ OK ]
Starting sshd: [ OK ]
[edureka@nn1 ~]$

```

(<https://d1jnx9ba8s6j9r.cloudfront.net/blog/wp-content/uploads/2015/06/restartsshdservice.png>)  
Restart SSH service.

Now you can login to the any node from Namenode without any authentication.

Open the core-site.xml file from the Active Name node and add the below properties.



```

edureka@nn1:~/HA/hadoop/etc/hadoop
File Edit View Search Terminal Help
[edureka@nn1 hadoop]$ gedit core-site.xml
core-site.xml (~/HA/hadoop-2.6.0/etc/hadoop) - gedit
File Edit View Search Tools Documents Help
core-site.xml
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.defaultFS</name>
    <value>hdfs://ha-cluster</value>
  </property>
  <property>
    <name>dfs.journalnode.edits.dir</name>
    <value>/home/edureka/HA/data/jn</value>
  </property>
</configuration>

```

(<https://d1jnx9ba8s6j9r.cloudfront.net/blog/wp-content/uploads/2015/06/core-site.xml>)  
Edit core-site.xml from Active namenode

Open hdfs-site.xml file in Active Namenode. Add the below Properties.

```

<property>
  <name>dfs.namenode.name.dir</name>
  <value>/home/edureka/HA/data/namenode</value>
</property>
<property>
  <name>dfs.replication</name>
  <value>1</value>
</property>
<property>
  <name>dfs.permissions</name>
  <value>>false</value>
</property>
<property>
  <name>dfs.nameservices</name>
  <value>ha-cluster</value>
</property>
<property>
  <name>dfs.ha.namenodes.ha-cluster</name>
  <value>nn1,nn2</value>
</property>
<property>
  <name>dfs.namenode.rpc-address.ha-cluster.nn1</name>
  <value>nn1.cluster.com:9000</value>
</property>
<property>
  <name>dfs.namenode.rpc-address.ha-cluster.nn2</name>
  <value>nn2.cluster.com:9000</value>
</property>

```

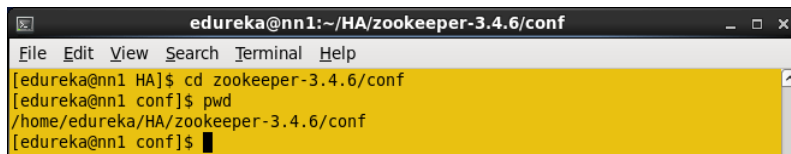
```

<property>
<name>dfs.namenode.http-address.ha-cluster.nn1</name>
<value>nn1.cluster.com:50070</value>
</property>
<property>
<name>dfs.namenode.http-address.ha-cluster.nn2</name>
<value>nn2.cluster.com:50070</value>
</property>
<property>
<name>dfs.namenode.shared.edits.dir</name>
<value>qjournal://nn1.cluster.com:8485;nn2.cluster.com:8485;dn1.cluster.com:8485/ha-cluster</value>
</property>
<property>
<name>dfs.client.failover.proxy.provider.ha-cluster</name>
<value>org.apache.hadoop.hdfs.server.namenode.ha.ConfiguredFailoverProxyProvider</value>
</property>
<property>
<name>dfs.ha.automatic-failover.enabled</name>
<value>>true</value>
</property>
<property>
<name>ha.zookeeper.quorum</name>
<value> nn1.cluster.com:2181,nn2.cluster.com:2181,dn1.cluster.com:2181 </value>
</property>
<property>
<name>dfs.ha.fencing.methods</name>
<value>sshfence</value>
</property>
<property>
<name>dfs.ha.fencing.ssh.private-key-files</name>
<value>/home/edureka/.ssh/id_rsa</value>
</property>

```

Change the directory to zookeeper's conf directory.

**Command:** cd zookeeper-3.4.6/conf



```

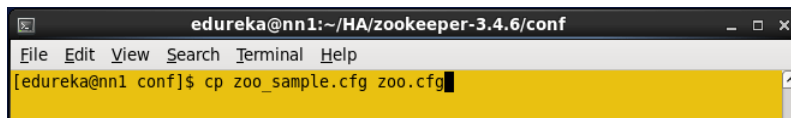
edureka@nn1:~/HA/zookeeper-3.4.6/conf
File Edit View Search Terminal Help
[edureka@nn1 HA]$ cd zookeeper-3.4.6/conf
[edureka@nn1 conf]$ pwd
/home/edureka/HA/zookeeper-3.4.6/conf
[edureka@nn1 conf]$

```

([https://d1jnx9ba8s6j9r.cloudfront.net/blog/wp-content/uploads/2015/06/change\\_zookeeper\\_directory\\_2.13.png](https://d1jnx9ba8s6j9r.cloudfront.net/blog/wp-content/uploads/2015/06/change_zookeeper_directory_2.13.png))  
Zookeeper Conf directory.

In a conf directory you have zoo\_sample.cfg file, create the zoo.cfg using zoo\_sample.cfg file.

**Command:** cp zoo\_sample.cfg zoo.cfg



```

edureka@nn1:~/HA/zookeeper-3.4.6/conf
File Edit View Search Terminal Help
[edureka@nn1 conf]$ cp zoo_sample.cfg zoo.cfg

```

([https://d1jnx9ba8s6j9r.cloudfront.net/blog/wp-content/uploads/2015/06/create\\_zoo\\_cfg\\_2.14.png](https://d1jnx9ba8s6j9r.cloudfront.net/blog/wp-content/uploads/2015/06/create_zoo_cfg_2.14.png))  
Create zoo.cfg file.

Create the directory in any location and use this directory to store the zookeeper data.

**Command:** mkdir <path, where you want to store the zookeeper files>



```

edureka@nn1:~/HA/data
File Edit View Search Terminal Help
[edureka@nn1 data]$ mkdir zookeeper

```

([https://d1jnx9ba8s6j9r.cloudfront.net/blog/wp-content/uploads/2015/06/create\\_zookeeper\\_data\\_2.15.png](https://d1jnx9ba8s6j9r.cloudfront.net/blog/wp-content/uploads/2015/06/create_zookeeper_data_2.15.png))  
Create a directory to store zookeeper data.

Open the zoo.cfg file.

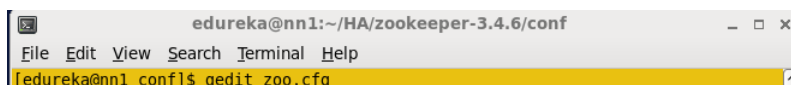
**Command:** gedit zoo.cfg

Add the directory path that is created in above step to the dataDir property and add the below details regarding remaining node, in the zoo.cfg file.

**Server.1=nn1.cluster.com:2888:3888**

**Server.2=nn2.cluster.com:2888:3888**

**Server.3=dn1.cluster.com:2888:3888**

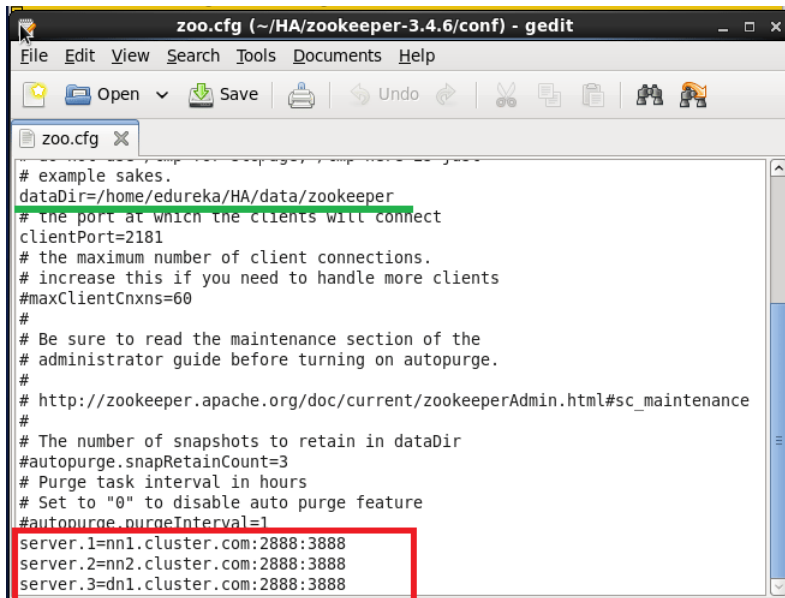


```

edureka@nn1:~/HA/zookeeper-3.4.6/conf
File Edit View Search Terminal Help
[edureka@nn1 conf]$ gedit zoo.cfg

```





```

zoo.cfg (~/.HA/zookeeper-3.4.6/conf) - gedit
File Edit View Search Tools Documents Help
zoo.cfg
# example sakes.
dataDir=/home/edureka/HA/data/zookeeper
# the port at which the clients will connect
clientPort=2181
# the maximum number of client connections.
# increase this if you need to handle more clients
#maxClientCnxns=60
#
# Be sure to read the maintenance section of the
# administrator guide before turning on autopurge.
#
# http://zookeeper.apache.org/doc/current/zookeeperAdmin.html#sc_maintenance
#
# The number of snapshots to retain in dataDir
#autopurge.snapRetainCount=3
# Purge task interval in hours
# Set to "0" to disable auto purge feature
#autopurge.purgeInterval=1
server.1=nn1.cluster.com:2888:3888
server.2=nn2.cluster.com:2888:3888
server.3=dn1.cluster.com:2888:3888

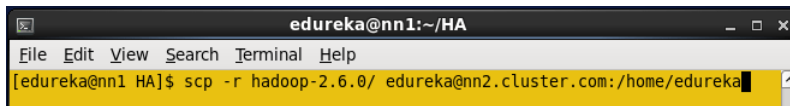
```

([https://d1jnx9ba8s6j9r.cloudfront.net/blog/wp-content/uploads/2015/06/zoo\\_cfg\\_file\\_2.16.1.png](https://d1jnx9ba8s6j9r.cloudfront.net/blog/wp-content/uploads/2015/06/zoo_cfg_file_2.16.1.png))

Edit zoo.cfg file.

Now copy the Java and Hadoop-2.6.0, zookeeper-3.4.6 directories, and .bashrc file to all the nodes (Standby name node, Data node) using scp command.

**Command:** `scp -r <path of directory> edureka@<ip address>:<path where you need to copy>`



```

edureka@nn1:~/HA
File Edit View Search Terminal Help
[edureka@nn1 HA]$ scp -r hadoop-2.6.0/ edureka@nn2.cluster.com:/home/edureka/

```

([https://d1jnx9ba8s6j9r.cloudfront.net/blog/wp-content/uploads/2015/06/Copy\\_HA\\_Z\\_.png](https://d1jnx9ba8s6j9r.cloudfront.net/blog/wp-content/uploads/2015/06/Copy_HA_Z_.png))

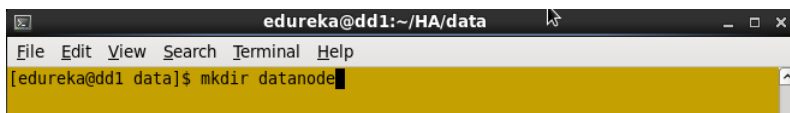
Copy Hadoop, Zookeeper and .bashrc file to all nodes.

Similarly, copy the .bashrc file and zookeeper directory to all the nodes and change the environment variables in each according to the respective node.

In a data node, create any directory where you need to store the HDFS blocks.

In a data node, you have to add the dfs.datanode.data.dir properties.

In my case, I created **datanode** directory to store the blocks.



```

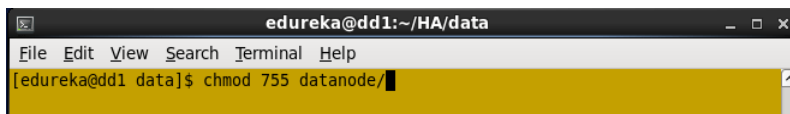
edureka@dd1:~/HA/data
File Edit View Search Terminal Help
[edureka@dd1 data]$ mkdir datanode

```

(<https://d1jnx9ba8s6j9r.cloudfront.net/blog/wp-content/uploads/2015/06/createdatanode.png>)

Create Datanode directory.

Change the permission to data node directory.



```

edureka@dd1:~/HA/data
File Edit View Search Terminal Help
[edureka@dd1 data]$ chmod 755 datanode/

```

([https://d1jnx9ba8s6j9r.cloudfront.net/blog/wp-content/uploads/2015/06/datanode\\_permission.png](https://d1jnx9ba8s6j9r.cloudfront.net/blog/wp-content/uploads/2015/06/datanode_permission.png))

Change Datanode directory permission.

Open the HDFS-site.xml file, add this Datanode directory path in dfs.datanode.data.dir property.

Note: Keep all the properties that are copied from the Active namenode; add dfs.datanode.data.dir one extract property in namenode.

```

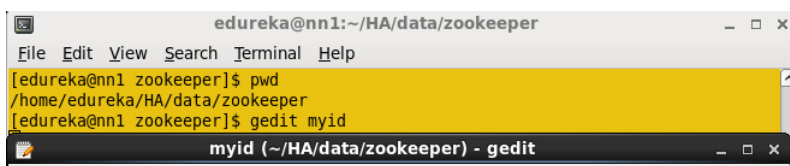
<property>
<name> dfs.datanode.data.dir</name>
<value>/home/edureka/HA/data/datanode</value>
</property>

```

In Active namenode, change the directory where you want to store the zookeeper configuration file (dataDir property path).

Create the myid file inside the directory and add numeric 1 to the file and save the file.

**Command:** `vi myid`

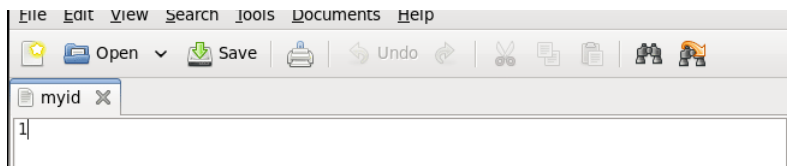


```

edureka@nn1:~/HA/data/zookeeper
File Edit View Search Terminal Help
[edureka@nn1 zookeeper]$ pwd
/home/edureka/HA/data/zookeeper
[edureka@nn1 zookeeper]$ gedit myid
myid (~/.HA/data/zookeeper) - gedit

```





(<https://d1jnx9ba8s6j9r.cloudfront.net/blog/wp-content/uploads/2015/06/myid.png>)

Create myid file.

In a standby namenode change the directory where you want to store the zookeeper configuration file (dataDir property path).

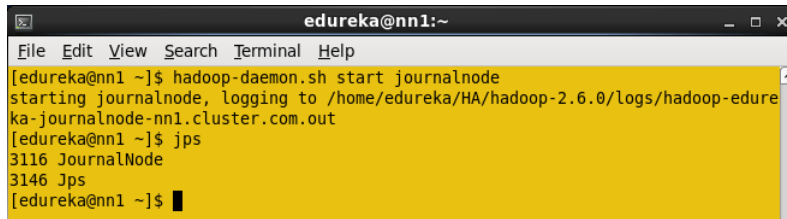
Create the myid file inside the directory and add numeric 2 to the file and save the file.

In a data node, change the directory where you want to store the zookeeper configuration file (dataDir property path).

Create the myid file inside the directory and add numeric 3 to the file and save the file.

Start the Journalnode in all the three nodes.

**Command:** `hadoop-daemon.sh start journalnode`



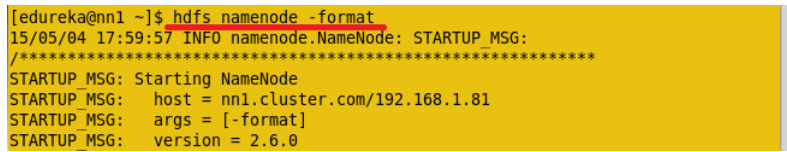
(<https://d1jnx9ba8s6j9r.cloudfront.net/blog/wp-content/uploads/2015/06/startjournal.png>)

Start the Journalnode.

When you enter jps command, you will see the JournalNode daemon in all the nodes.

Format the Active namenode.

**Command:** `HDFS namenode -format`

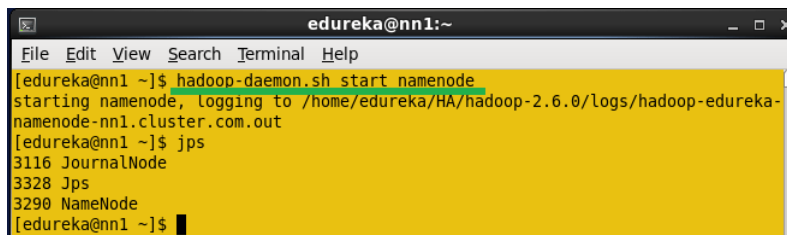


(<https://d1jnx9ba8s6j9r.cloudfront.net/blog/wp-content/uploads/2015/06/formatNN.png>)

Format Active NameNode.

Start the Namenode daemon in Active namenode.

**Command:** `hadoop-daemon.sh start namenode`

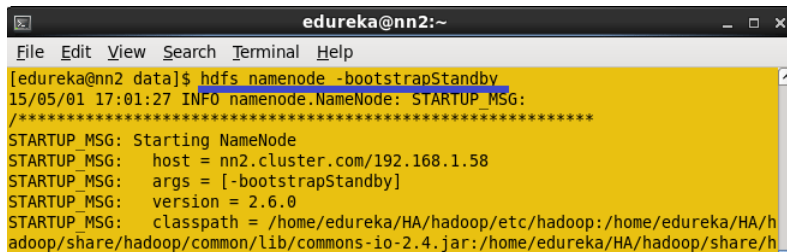


(<https://d1jnx9ba8s6j9r.cloudfront.net/blog/wp-content/uploads/2015/06/startNN.png>)

Start Namenode.

Copy the HDFS Meta data from active name node to standby namenode.

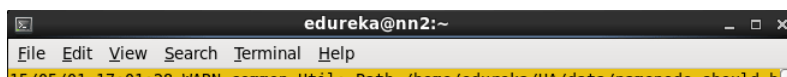
**Command:** `HDFS namenode -bootstrapStandby`



(<https://d1jnx9ba8s6j9r.cloudfront.net/blog/wp-content/uploads/2015/06/bootstrap.png>)

Copy the HDFS Meta data from Active name node to Standby Namenode.

Once you run this command, you will get the information from which node and location the meta data is copying and whether it is copying successfully or not.



```

15/05/01 17:01:28 WARN common.Util: Path /home/edureka/HA/data/namenode should b
e specified as a URI in configuration files. Please update hdfs configuration.
15/05/01 17:01:28 WARN common.Util: Path /home/edureka/HA/data/namenode should b
e specified as a URI in configuration files. Please update hdfs configuration.
15/05/01 17:01:28 WARN util.NativeCodeLoader: Unable to load native-hadoop libra
ry for your platform... using builtin-java classes where applicable

About to bootstrap Standby ID nn2 from:
  Nameservice ID: ha-cluster
  Other Namenode ID: nn1
  Other NN's HTTP address: http://nn1.cluster.com:50070
  Other NN's IPC address: nn1.cluster.com/192.168.1.81:9000
  Namespace ID: 1585101516
  Block pool ID: BP-1327768063-192.168.1.81-1430511737446
  Cluster ID: CID-fcc43bfb-e9fd-455c-ba21-3b2a8901c2c4
  Layout version: -60

Re-format filesystem in Storage Directory /home/edureka/HA/data/namenode ? (Y or
N) Y
15/05/01 17:01:31 INFO common.Storage: Storage directory /home/edureka/HA/data/n
amenode has been successfully formatted.
15/05/01 17:01:31 WARN common.Util: Path /home/edureka/HA/data/namenode should b
e specified as a URI in configuration files. Please update hdfs configuration.
15/05/01 17:01:31 WARN common.Util: Path /home/edureka/HA/data/namenode should b
e specified as a URI in configuration files. Please update hdfs configuration.

```

(<https://d1jnx9ba8s6j9r.cloudfront.net/blog/wp-content/uploads/2015/06/information.png>)  
Information of Active namenode details.

Once Meta data is copied from Active namenode to standby namenode, you will get the message shown below in the screenshot.

```

edureka@nn2:~
File Edit View Search Terminal Help
15/05/01 17:01:31 INFO common.Storage: Storage directory /home/edureka/HA/data/n
amenode has been successfully formatted.
15/05/01 17:01:31 WARN common.Util: Path /home/edureka/HA/data/namenode should b
e specified as a URI in configuration files. Please update hdfs configuration.
15/05/01 17:01:31 WARN common.Util: Path /home/edureka/HA/data/namenode should b
e specified as a URI in configuration files. Please update hdfs configuration.
15/05/01 17:01:32 INFO namenode.TransferFsImage: Opening connection to http://nn
1.cluster.com:50070/imageTransfer?getImage=1&txid=0&storageInfo=-60:1585101516:0
:CID-fcc43bfb-e9fd-455c-ba21-3b2a8901c2c4
15/05/01 17:01:32 INFO namenode.TransferFsImage: Image Transfer timeout configur
ed to 60000 milliseconds
15/05/01 17:01:33 INFO namenode.TransferFsImage: Transfer took 0.01s at 0.00 KB/
s
15/05/01 17:01:33 INFO namenode.TransferFsImage: Downloaded file fsimage.ckpt_00
0000000000000000 size 354 bytes.
15/05/01 17:01:33 INFO util.ExitUtil: Exiting with status 0
15/05/01 17:01:33 INFO namenode.NameNode: SHUTDOWN MSG:
/*****
SHUTDOWN MSG: Shutting down NameNode at nn2.cluster.com/192.168.1.58
*****/

```

(<https://d1jnx9ba8s6j9r.cloudfront.net/blog/wp-content/uploads/2015/06/completed.png>)  
Information regarding HDFS in Standby Namenode.

Start the namenode daemon in Standby namenode machine.

**Command:** `hadoop-daemon.sh start namenode`

Now start the Zookeeper service in all the three nodes.

**Command:** `zkServer.sh start` (Run this command in all the nodes)

In Active Namenode:

```

[edureka@nn1 ~]$ zkServer.sh start
JMX enabled by default
Using config: /home/edureka/HA/zookeeper-3.4.6/bin/./conf/zoo.cfg
Starting zookeeper ... STARTED

```

(<https://d1jnx9ba8s6j9r.cloudfront.net/blog/wp-content/uploads/2015/06/activezookeeper.png>)  
Start zookeeper in Active NameNode.

In Standby Namenode:

```

[edureka@nn2 data]$ zkServer.sh start
JMX enabled by default
Using config: /home/edureka/HA/zookeeper-3.4.6/bin/./conf/zoo.cfg
Starting zookeeper ... STARTED

```

([https://d1jnx9ba8s6j9r.cloudfront.net/blog/wp-content/uploads/2015/06/zookeeper\\_standby.png](https://d1jnx9ba8s6j9r.cloudfront.net/blog/wp-content/uploads/2015/06/zookeeper_standby.png))  
Start zookeeper in standby NameNode.

In Data node:

```

[edureka@dn1 data]$ zkServer.sh start
JMX enabled by default
Using config: /home/edureka/HA/zookeeper-3.4.6/bin/./conf/zoo.cfg
Starting zookeeper ... STARTED

```

([https://d1jnx9ba8s6j9r.cloudfront.net/blog/wp-content/uploads/2015/06/zookeeper\\_datanode.png](https://d1jnx9ba8s6j9r.cloudfront.net/blog/wp-content/uploads/2015/06/zookeeper_datanode.png))  
Start zookeeper in DataNode.

After running the Zookeeper server, enter JPS command. In all the nodes you will see the QuorumPeerMain service.

Start the Data node daemon in Data node machine.

**Command:** `hadoop-daemon.sh start datanode`

Start the Zookeeper fail over controller in Active name node and standby name node.

Format the zookeeper fail over controller in Active namenode.

**Command:** HDFS zkfc -formatZK

```

edureka@nn1:~/.ssh
File Edit View Search Terminal Help
[edureka@nn1 ~]$ hdfs zkfc -formatZK
15/05/01 22:34:14 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
15/05/01 22:34:14 INFO tools.DFSZKFailoverController: Failover controller configured for NameNode NameNode at nn1.cluster.com/192.168.1.81:9000
15/05/01 22:34:15 INFO zookeeper.ZooKeeper: Client environment:zookeeper.version=3.4.6-1569965, built on 02/20/2014 09:09 GMT
15/05/01 22:34:15 INFO zookeeper.ZooKeeper: Client environment:host.name=nn1.cluster.com
15/05/01 22:34:15 INFO zookeeper.ZooKeeper: Client environment:java.version=1.7.0_79
  
```

(<https://d1jnx9ba8s6j9r.cloudfront.net/blog/wp-content/uploads/2015/06/formatzk.png>)

Format ZKFC.

Start the ZKFC in Active namenode.

**Command:** hadoop-daemon.sh start zkfc

Enter jps command to check the DFSZKFailoverController daemons.

```

[edureka@nn1 ~]$ hadoop-daemon.sh start zkfc
starting zkfc, logging to /home/edureka/HA/hadoop-2.6.0/logs/hadoop-edureka-zkfc-nn1.cluster.com.out
[edureka@nn1 ~]$ jps
8009 QuorumPeerMain
7606 NameNode
8142 DFSZKFailoverController
8206 Jps
7430 JournalNode
  
```

(<https://d1jnx9ba8s6j9r.cloudfront.net/blog/wp-content/uploads/2015/06/zkfcrun.png>)

Start ZKFC.

Format the zookeeper fail over controller in Standby namenode.

**Command:** hdfs zkfc -formatZK

Start the ZKFC in Standby namenode.

**Command:** hadoop-daemon.sh start zkfc

Enter jps command to check the DFSZKFailoverController daemons.

Now check the status of each Namenode, which node is Active or which node is on Standby by using the below command.

**Command:** hdfs haadmin -getServiceState nn1

```

[edureka@nn1 ~]$ hdfs haadmin -getServiceState nn1
15/05/01 22:36:01 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
active
[edureka@nn1 ~]$ hdfs haadmin -getServiceState nn2
15/05/01 22:36:17 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
standby
[edureka@nn1 ~]$
  
```

(<https://d1jnx9ba8s6j9r.cloudfront.net/blog/wp-content/uploads/2015/06/NNstatus.png>)

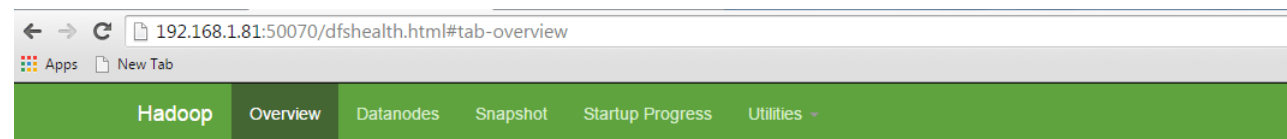
Check status of each NameNode.

Now Check the status of each Namenode using the web browser.

Open the Web browser and enter the below URL.

<IP Address of Active Namenode>:50070

It will show whether the name node is Active or on standby.



Overview 'nn1.cluster.com:9000' (active)

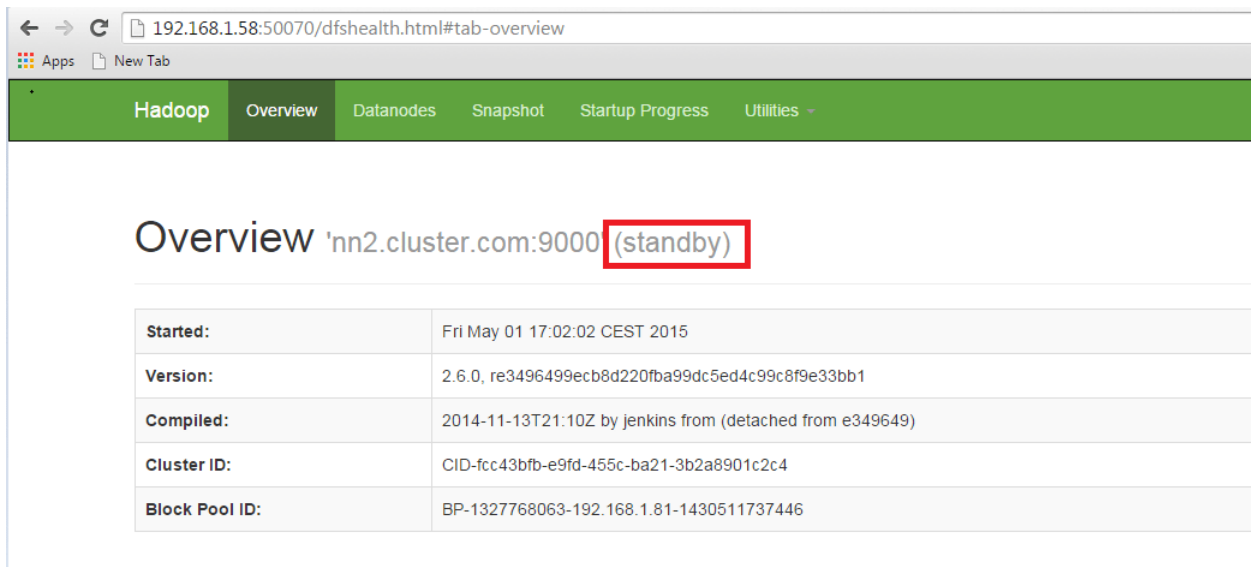
Started:	Fri May 01 22:23:12 CEST 2015
Version:	2.6.0-r3496499ecb8d220fba99dc5ed4c98c9f6a33bb1

<b>Version:</b>	2.6.0, re3496499ecb8d220fba99dc5ed4c99c8f9e33bb1
<b>Compiled:</b>	2014-11-13T21:10Z by jenkins from (detached from e349649)
<b>Cluster ID:</b>	CID-fcc43bfb-e9fd-455c-ba21-3b2a8901c2c4
<b>Block Pool ID:</b>	BP-1327768063-192.168.1.81-1430511737446

(<https://d1jnx9ba8s6j9r.cloudfront.net/blog/wp-content/uploads/2015/06/ActiveNN.png>)

Active NameNode.

Open another name node details using the web browser.



The screenshot shows a web browser window with the URL `192.168.1.58:50070/dfshealth.html#tab-overview`. The page title is "Overview 'nn2.cluster.com:9000' (standby)". Below the title is a table with the following details:

<b>Started:</b>	Fri May 01 17:02:02 CEST 2015
<b>Version:</b>	2.6.0, re3496499ecb8d220fba99dc5ed4c99c8f9e33bb1
<b>Compiled:</b>	2014-11-13T21:10Z by jenkins from (detached from e349649)
<b>Cluster ID:</b>	CID-fcc43bfb-e9fd-455c-ba21-3b2a8901c2c4
<b>Block Pool ID:</b>	BP-1327768063-192.168.1.81-1430511737446

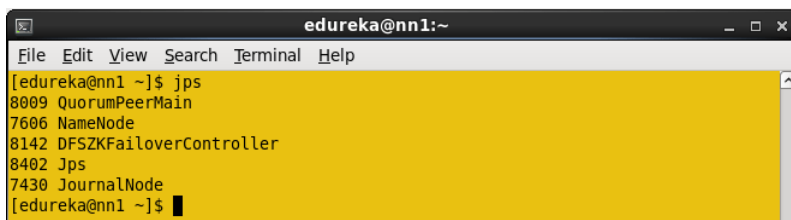
(<https://d1jnx9ba8s6j9r.cloudfront.net/blog/wp-content/uploads/2015/06/standby.png>)

Standby NameNode.

In the Active namenode, kill the namenode daemon to change the Standby name node to active namenode.

Enter jps in Active namenode and kill the daemon.

**Command:** `sudo kill -9 <namenode process ID>`



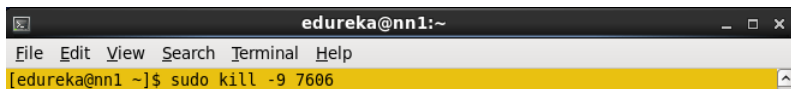
```
edureka@nn1:~$ jps
8009 QuorumPeerMain
7606 NameNode
8142 DFSZKFailoverController
8402 Jps
7430 JournalNode
edureka@nn1:~$
```

(<https://d1jnx9ba8s6j9r.cloudfront.net/blog/wp-content/uploads/2015/06/JPS.png>)

Daemons Process ID.

The Namenode process ID is 7606, kill the namenode.

**Command:** `Sudo kill -9 7606`

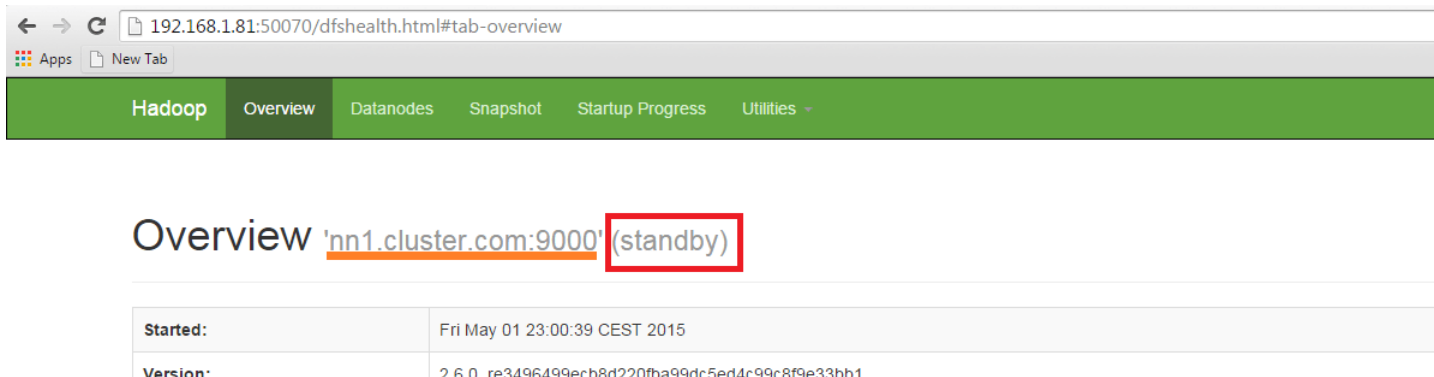


```
edureka@nn1:~$ sudo kill -9 7606
```

(<https://d1jnx9ba8s6j9r.cloudfront.net/blog/wp-content/uploads/2015/06/kill.png>)

Kill the Name Node process

Open the two nodes through web browser and check the status.



The screenshot shows a web browser window with the URL `192.168.1.81:50070/dfshealth.html#tab-overview`. The page title is "Overview 'nn1.cluster.com:9000' (standby)". Below the title is a table with the following details:

<b>Started:</b>	Fri May 01 23:00:39 CEST 2015
<b>Version:</b>	2.6.0, re3496499ecb8d220fba99dc5ed4c99c8f9e33bb1

Compiled:	2014-11-13T21:10Z by jenkins from (detached from e349649)
Cluster ID:	CID-fcc43bfb-e9fd-455c-ba21-3b2a8901c2c4
Block Pool ID:	BP-1327768063-192.168.1.81-1430511737446

(https://d1jnx9ba8s6j9r.cloudfront.net/blog/wp-content/uploads/2015/06/after5.png)  
Namenode details.

← → ↺ 192.168.1.58:50070/dfshealth.html#tab-overview

Apps New Tab Other bookmarks

Hadoop Overview Datanodes Snapshot Startup Progress Utilities

Overview 'nn2.cluster.com:9000' (active)

Started:	Fri May 01 17:02:02 CEST 2015
Version:	2.6.0, re3496499ecb8d220fba99dc5ed4c99c8f9e33bb1
Compiled:	2014-11-13T21:10Z by jenkins from (detached from e349649)
Cluster ID:	CID-fcc43bfb-e9fd-455c-ba21-3b2a8901c2c4
Block Pool ID:	BP-1327768063-192.168.1.81-1430511737446

(https://d1jnx9ba8s6j9r.cloudfront.net/blog/wp-content/uploads/2015/06/new.png)  
NameNode status.

Congratulations, you have successfully setup a HDFS High Availability Cluster in Hadoop.

Now that you have understood Hadoop High Availability Cluster Architecture, check out the **Hadoop training (https://www.edureka.co/big-data-and-hadoop/)** by Edureka, a trusted online learning company with a network of more than 250,000 satisfied learners spread across the globe. The Edureka Big Data Hadoop Certification Training course helps learners become expert in HDFS, Yarn, MapReduce, Pig, Hive, HBase, Oozie, Flume and Sqoop using real-time use cases on Retail, Social Media, Aviation, Tourism, Finance domain.

Got a question for us? Please mention it in the comments section and we will get back to you.

```
window._LQ_ = window._LQ_ || {};  
lqQuizModal(window, document, {quizId:'XAIVp8',baseUrl:'https://quiz.leadquizzes.com/',trigger:'exit'}, _LQ_);
```

About Ashish Bakshi (11 Posts (https://www.edureka.co/blog/author/ashishbedureka-co/))

f t in g+

(https://www.edureka.co/blog/how-to-set-up-hadoop-cluster-with-hdfs-high-availability/) to- to- to- to- set- set- set- set- up- up- up- up- hadoop hadoop hadoop hadoop cluster- cluster- cluster- cluster- with- with- with- with- hdfs- hdfs- hdfs- hdfs- high- high- high- high- availability availability availability availability)

Share on

Got your brain cells running?  
Stay tuned to latest technology updates

Enter your Email Address

SUBSCRIBE

## Related Posts



**Overview of Hadoop 2.0 Cluster Architecture Federation**  
👁 18K

(<https://www.edureka.co/blog/overview-of-hadoop-2-0-cluster-architecture-federation/>)



**Hadoop Developer Job Responsibilities & Skills**  
👁 54.5K

(<https://www.edureka.co/blog/hadoop-developer-job-responsibilities-skills/>)



**Setting Up A Multi Node Cluster In Hadoop 2.X**  
👁 317.3K

(<https://www.edureka.co/blog/hadoop-up-a-multi-node-cluster-in-hadoop-2-x/>)



**Hadoop Tutorial: All you need to know about Hadoop!**  
👁 87.9K

(<https://www.edureka.co/blog/hadoop-tutorial/>)

## Comments

18 Comments

17 Comments

<https://www.edureka.co/blog/>

Rajiv Chaudhuri ▾

🤍 Recommend 5

🔗 Share

Sort by Best ▾



Join the discussion...

**Ltifi Sedki** • 6 months ago

thank you a lot for this course .

when i typed ssh-keygen -t rsa to generate keys, this message is displayed " Too many arguments", please help me.

^ | ▾ • Reply • Share ›

**Vikash** → Ltifi Sedki • 6 months ago

type "-t" . rather than copy and paste

1 ^ | ▾ • Reply • Share ›

**SacTiw** • 8 months ago

Normally a client would send a get/put file request to a particular "namenode" right? So once a failover has happened how would client get to know about it?

Assuming it is client responsibility to perform the retry on failure in that case is there a way client can first query for currently active namenode and then send a request to that one?

^ | ▾ • Reply • Share ›

**Baris** • 9 months ago

It would be really good to show how to restart this system.

Thank you for sharing this valuable information.

^ | ▾ • Reply • Share ›

**EdurekaSupport** Mod → Baris • 7 months ago

Thank you @Baris for appreciating our work. We will look into your suggestions as well. Cheers :)

^ | ▾ • Reply • Share ›

**Hassan Asghar** • 9 months ago

my hadoop cluster is setup, and working fine:

i ran word count example:

can anybody provide me the following formulas to calculate some parameters:

Response Time:

Throughput:

Average I/o Rate:

Execution Time:

Thanks in advance

^ | ▾ • Reply • Share ›

**Den Kushnerik** • 2 years ago

Hello. Its a very helpful instruction for me!

Do we need to format the ZKFC on Standby NameNode too?

According to this page: <http://hadoop.apache.org/do...> we must do it one time: "...next step is to initialize required state in ZooKeeper. You can do so by running the following command from one of the NameNode hosts."

^ | v • Reply • Share ›



**Sanjay** • 2 years ago

Normally when we setup a hadoop cluster (non HA), we need to configure yarn by modifying its yarn-site.xml . For HA, don't we require any HA specific modification to yarn-site.xml ?

^ | v • Reply • Share ›



**Ashish Bakshi** → Sanjay • 2 years ago

Thanks Sanjay for going through the blog.

In this blog, we are modifying hdfs-site.xml because we are enabling HA feature only for NameNode. And yes you are absolutely correct, you can have HA for ResourceManager as well where you will have to modify the yarn-site.xml similarly. You can follow the Hadoop documentations to setup HA for ResourceManager which is given below:

<https://hadoop.apache.org/d...>

^ | v • Reply • Share ›



**Rakibul hassan Rakib** • 2 years ago

I am just correcting your HA Architecture image



^ | v • Reply • Share ›



**Rakibul hassan Rakib** • 2 years ago

After killing active or standby namenode I am not getting web view of killing namenode. Is it possible to getting web view after killing namenode ? . But you have seen two namenode web view after killing one namenode. How it is possible? I am facing some problem in my namenode.

Thank you

Rakib

^ | v • Reply • Share ›



**EdurekaSupport** Mod → Rakibul hassan Rakib • 2 years ago

Hey Rakibul, thanks for checking out the blog. Please follow the steps given below:

-> Please Check your hdfs-site.xml configuration file and make sure that you have set up the automatic failover as per given in the blog.

-> In case you are still facing the issue, change the directory for namenode, datanode, JN and zookeeper and give the permission 755 for these directories  
`chmod 755 directory_path`

-> Format the Active Namenode and start the services as per given in the blog

Hope this helps.

^ | v • Reply • Share ›



**Mani** → Rakibul hassan Rakib • 2 years ago

Hey Rakib,

If the namenode is manually transitioned from active to standby you should be able to see the WEB UI of the namenode as it is still active. But if there is a failover in the active namenode and the it got a automatic transition to the standby namenode you can't have the web ui because of the obvious reason that the namenode is down. Once you fix the dead namenode you can see the UI with STANDBY mentioned in the UI. Hope this helps

Thanks,

MK

^ | v • Reply • Share ›



**anil kumar** • 3 years ago

am inistaling high avalability like nn1 & nn2 and dn1 .... in that nn1 and nn2 both are standby mode only what i do now

^ | v • Reply • Share ›



**Mani** → anil kumar • 2 years ago

Hope you got the solution by now anil. It might be the reason that you did not enable automatic failover property in hdfs-site.xml. According to what you are saying that your cluster is in manual failover mode. In this scenario you have to individually designate which name node should be active or standby.

`hdfs haadmin -transitionToActive nn1`  
 (nn1 - Active , nn2 - Standby)

`hdfs haadmin -transitionToStandby nn1`  
 (nn1 - Standby , nn2 - Standby)

`hdfs haadmin -transitionToActive nn2`  
 (nn1 - Standby , nn2 - Active)

`hdfs haadmin -transitionToStandby nn2`  
 (nn1 - Standby , nn2 - Standby)

Check your name node service status using the command:

`hdfs haadmin -getServiceStatus <nn1,nn2>`

If you by mistake make both of them active you might encounter scenario of split-brain where on both nodes edits will be in progress resulting in corrupted metadata.

Hope this helps!



Thanks,  
MK

^ | v • Reply • Share ›



**sureseh** • 3 years ago

Getting below error when i follow the above configuration settings.

15/11/08 01:58:34 ERROR namenode.FSNamesystem: FSNamesystem initialization failed.

java.io.IOException: Invalid configuration: a shared edits dir must not be specified if HA is not enabled.

and i dont find solution for this from google.

Can someone help

regards  
suresh bk

^ | v • Reply • Share ›



**EdurekaSupport** Mod ➔ sureseh • 3 years ago

Hi Suresh bk

Thank you for reaching out to us.

You can connect with our 24/7 support team with all your queries and doubts regarding Hadoop once you enroll for the course.

You can also get in touch with us by contacting our sales team on +91-8880862004 (India) or 1800 275 9730 (US toll free). You can mail us on sales@edureka.co.

^ | v • Reply • Share ›

ALSO ON [HTTPS://WWW.EDUREKA.CO/BLOG/](https://www.edureka.co/blog/)

### Git vs Github – Demystifying The Differences

2 comments • 8 months ago

Donald Peoples — this was very helpful! Thanks.  
Avatar

### AI and IoT in FIFA: Smart Sports

3 comments • 2 months ago

Apoorva Verma — The digital transformation of football which began with Goal line  
Avatar technology (GLT), is going beyond GLT towards Internet of ...

### Spring Boot Microservices: Building Microservices Application Using Spring Boot

60 comments • 3 months ago

Gunasekaran — Thanks!!!  
Avatar

### Top Technical Skills to Secure Jobs of the Future

3 comments • a month ago

Gvsm Chaithanya — Hi Mir Juned sir, Thanks for your response my email ID is:  
Avatar gvsmc1996@gmail.com

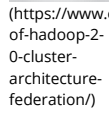
Subscribe Add Disqus to your site [Add Disqus](#) [Disqus' Privacy Policy](#) [Privacy Policy](#) [Privacy Policy](#)

Subscribe  
to our newsletter

Enter your Email Address

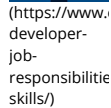
**SUBSCRIBE**

Related Blogs



Top 50 Hadoop Interview Questions for 2018

(<https://www.interviewquestions.top/50-hadoop-interview-questions-2016/>)



([https://www.cloudera.com/documentation/enterprise/5-10/topics/setting\\_up\\_a\\_multi-node\\_cluster\\_in\\_hadoop\\_2.X.html](https://www.cloudera.com/documentation/enterprise/5-10/topics/setting_up_a_multi-node_cluster_in_hadoop_2.X.html))

(<http://www.edureka.co/big-data-and-hadoop>)

**edureka!**  
(<https://www.edureka.co>)  
© 2014 Brain4ce Education Solutions Pvt. Ltd. All rights Reserved.

"PMP®", "PMI®", "PMI-ACP®" and "PMBOK®" are registered marks of the Project Management Institute, Inc.  
MongoDB®, Mongo and the leaf logo are the registered trademarks of MongoDB, Inc.