

Exp No: 9

Date: 09/10/2024

HADOOP

SET UP A SINGLE HADOOP CLUSTER AND SHOW THE PROCESS

USING WEB UI

AIM:

To set-up one node Hadoop cluster.

PROCEDURE:

1. System Update
2. Install Java
3. Add a dedicated Hadoop user
4. Install SSH and setup SSH certificates
5. Check if SSH works
6. Install Hadoop
7. Modify Hadoop config files
8. Format Hadoop filesystem
9. Start Hadoop
10. Check Hadoop through web UI
11. Stop Hadoop

THEORY:

Hadoop is an Apache open-source framework written in java that allows distributed processing of large datasets across clusters of computers using simple programming models.

A Hadoop frame-worked application works in an environment that provides distributed storage and computation across clusters of computers. Hadoop is designed to scale up from a single server to thousands of machines, each offering local computation and storage.

HADOOP ARCHITECTURE

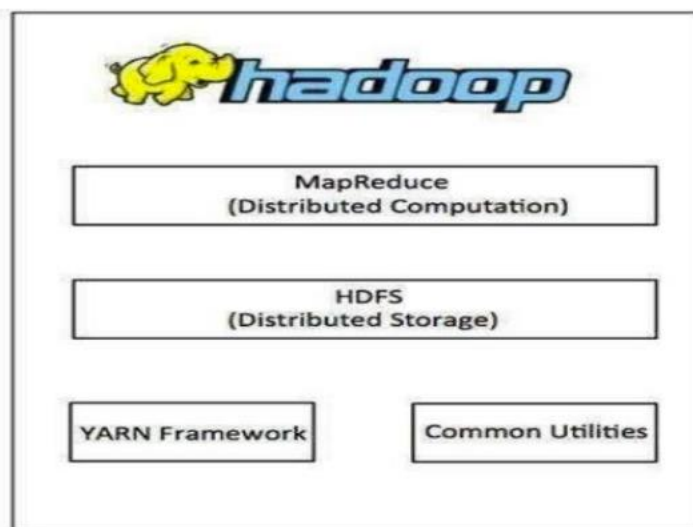
Hadoop framework includes following four modules:

Hadoop Common: These are Java libraries and utilities required by other Hadoop modules. These libraries provide filesystem and OS level abstractions and contain the necessary Java files and scripts required to start Hadoop.

Hadoop YARN: This is a framework for job scheduling and cluster resource management.

Hadoop Distributed File System (HDFS): A distributed file system that provides high-throughput access to application data.

Hadoop MapReduce: This is a YARN-based system for parallel processing of large data sets. We can use following diagram to depict these four components available in Hadoop framework.



PROCEDURE:

Step 1 – System Update

```
$ sudo apt-get update
```

Step 2 – Install Java and Set JAVA_HOME

```
$ sudo apt-get install default-jdk
```

```
$ java -version
```

```
haresh@fedora:~$ java -version
openjdk version "21.0.4" 2024-07-16
OpenJDK Runtime Environment (Red_Hat-21.0.4.0.7-2) (build 21.0.4+7)
OpenJDK 64-Bit Server VM (Red_Hat-21.0.4.0.7-2) (build 21.0.4+7, mixed mode, sharing)
```

Step 3 – Add a dedicated Hadoop user

```
$ sudo addgroup Hadoop
```

```
$ sudo adduser --ingroup hadoop hduser
```

```
// Add hduser to sudo user group
```

```
$ sudo adduser hduser sudo
```

Step 4 – Install SSH and Create Certificates

```
$ sudo apt-get install ssh
```

```
$ su hduser
```

```
$ ssh-keygen -t rsa -P ""
```

```
$ cat $HOME/.ssh/id_rsa.pub >> $HOME/.ssh/authorized_keys
```

Step 6 – Install Hadoop

```
$ wget https://archive.apache.org/dist/hadoop/core/hadoop-2.8.4/hadoop-2.8.4.tar.gz
```

```
// Extract Hadoop-2.8.4
```

```
$ sudo tar xvfz hadoop-2.8.4.tar.gz
```

```
haresh@fedora:~/Downloads$ ls
apache-hive-3.1.2-bin.tar.gz      hadoop-3.4.0.tar.gz
apache-hive-3.PN_Sf4-n.1.2-bin.tar.gz.part  pig-0.16.0.tar.gz
hadoop-3.3.6.tar.gz
```

```
// Create a folder 'hadoop' in /usr/local
$ sudo mkdir -p /usr/local/Hadoop
// Move the Hadoop folder to /usr/local/hadoop
$ sudo mv hadoop-2.8.4 /usr/local/Hadoop
// Assigning read and write access to Hadoop folder
$ sudo chown -R hduser:hadoop /usr/local/Hadoop
```

Step 7 - Modify Hadoop config files

//Hadoop Environmental variable setting – The following files will be modified

1. ~/.bashrc
2. /usr/local/hadoop/hadoop-2.8.4/etc/hadoop/hadoop-env.sh
3. /usr/local/hadoop/hadoop-2.8.4/etc/hadoop/core-site.xml
4. /usr/local/hadoop/hadoop-2.8.4/etc/hadoop/hdfs-site.xml
5. /usr/local/hadoop/hadoop-2.8.4/etc/hadoop/yarn-site.xml
6. /usr/local/hadoop/hadoop-2.8.4/etc/hadoop/mapred-site.xml.template

```
$ sudo nano ~/.bashrc
```

// Add the following lines at the end of the file

```
export JAVA_HOME=/usr/lib/jvm/java-8-oracle
export HADOOP_HOME=/usr/local/hadoop/hadoop-2.8.4 export
PATH=$PATH:$HADOOP_HOME/bin
export PATH=$PATH:$HADOOP_HOME/sbin
export HADOOP_MAPRED_HOME=$HADOOP_HOME export
HADOOP_COMMON_HOME=$HADOOP_HOME export
HADOOP_HDFS_HOME=$HADOOP_HOME
export YARN_HOME=$HADOOP_HOME
```

```
HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/lib/native export
```

```
HADOOP_OPTS="-Djava.library.path=$HADOOP_HOME/lib" export
```

```
PATH=$PATH:/usr/local/hadoop/hadoop-2.8.4/bin
```

```
export JAVA_HOME=/usr/lib/jvm/java-1.8.0-openjdk
export HADOOP_HOME=/home/haresh/hadoop
export HADOOP_INSTALL=$HADOOP_HOME
export HADOOP_MAPRED_HOME=$HADOOP_HOME
export HADOOP_COMMON_HOME=$HADOOP_HOME
export HADOOP_HDFS_HOME=$HADOOP_HOME
export YARN_HOME=$HADOOP_HOME
export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/lib/native
export PATH=$PATH:$HADOOP_HOME/sbin:$HADOOP_HOME/bin
export HADOOP_OPTS="-Djava.library.path=$HADOOP_HOME/lib/native"
export HADOOP_STREAMING=$HADOOP_HOME/share/hadoop/tools/lib/hadoop-streaming-3.
export HADOOP_LOG_DIR=$HADOOP_HOME/logs
export PDSH_RCMD_TYPE=ssh
```

// Configure Hadoop Files

```
$ cd /usr/local/hadoop/hadoop-2.8.4/etc/hadoop/
```

```
$ sudo nano hadoop-env.sh
```

// Add following line in hadoop-env.sh – Set JAVA variable in Hadoop

The java implementation to use.

```
export JAVA_HOME=/usr/lib/jvm/java-8-oracle
```

```
GNU nano 7.2 hadoop-env.sh

##
Generic settings for HADOOP
##

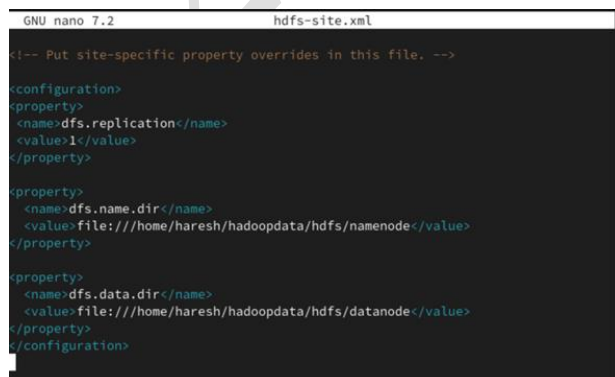
Technically, the only required environment variable is JAVA_HOME.
All others are optional.  However, the defaults are probably not
preferred.  Many sites configure these options outside of Hadoop,
such as in /etc/profile.d

The java implementation to use.  By default, this environment
variable is REQUIRED on ALL platforms except OS X!
export JAVA_HOME=/usr/lib/jvm/java-1.8.0-openjdk

Location of Hadoop.  By default, Hadoop will attempt to determine
this location based upon its execution path.
export HADOOP_HOME=

Location of Hadoop's configuration information.  i.e., where this
file is living.  If this is not defined, Hadoop will attempt to
```

```
// Create datanode and namenode
$ sudo mkdir -p /usr/local/hadoop_tmp/hdfs/namenode
$ sudo mkdir -p /usr/local/hadoop_tmp/hdfs/datanode
// Changing ownership to hadoop_tmp
$ sudo chown -R hduser:hadoop /usr/local/hadoop_tmp
// Edit hdfs-site.xml
$ sudo nano hdfs-site.xml
// Add the following lines between <configuration> ..... </configuration>
<configuration>
<property>
<name>dfs.replication</name>
<value>1</value>
</property>
<property>
<name>dfs.namenode.name.dir</name>
<value>file:/usr/local/hadoop_tmp/hdfs/namenode</value>
</property>
<property>
<name>dfs.datanode.data.dir</name>
<value>file:/usr/local/hadoop_tmp/hdfs/datanode</value>
</property>
</configuration>
```



The screenshot shows a terminal window with the title bar 'GNU nano 7.2 hdfs-site.xml'. The content of the file is displayed in a dark background with syntax highlighting. It starts with a comment: '<!-- Put site-specific property overrides in this file. -->'. This is followed by an XML configuration block. The first property sets 'dfs.replication' to '1'. The second property sets 'dfs.name.dir' to 'file:///home/haresh/hadoopdata/hdfs/namenode'. The third property sets 'dfs.data.dir' to 'file:///home/haresh/hadoopdata/hdfs/datanode'. The block ends with '</configuration>'. The cursor is at the end of the last line.

```
GNU nano 7.2 hdfs-site.xml
<!-- Put site-specific property overrides in this file. -->
<configuration>
<property>
  <name>dfs.replication</name>
  <value>1</value>
</property>
<property>
  <name>dfs.name.dir</name>
  <value>file:///home/haresh/hadoopdata/hdfs/namenode</value>
</property>
<property>
  <name>dfs.data.dir</name>
  <value>file:///home/haresh/hadoopdata/hdfs/datanode</value>
</property>
</configuration>
```

```
// Edit core-site.xml
```

```
$ sudo nano core-site.xml
```

21

```
// Add the following lines between <configuration> ..... </configuration>
```

```
<configuration>
```

```
<property>
```

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

```
<value>hdfs://localhost:9000</value>
```

```
</property>
```

```
</configuration>
```

```
// Edit yarn-site.xml
```

```
$ sudo nano yarn-site.xml
```

```
// Add the following lines between <configuration> ..... </configuration>
```

```
<configuration>
```

```
<property>
```

```
<name>yarn.nodemanager.aux-services</name>
```

```
<value>mapreduce_shuffle</value>
```

```
</property>
```

```
<property>
```

```
<name>yarn.nodemanager.aux-services.mapreduce.shuffle.class</name>
```

```
<value>org.apache.hadoop.mapred.Shuffle-Handler</value>
```

```
</property>
```

```
</configuration>
```

```
GNU nano 7.2                                core-site.xml
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://localhost:9000</value>
</property>
</configuration>
```

// Edit mapred-site.xml

\$ cp /usr/local/hadoop/hadoop-2.8.4/etc/hadoop/mapred-site.xml.template

/usr/local/hadoop/hadoop-2.8.4/etc/hadoop/mapred-site.xml

\$ sudo nano mapred-site.xml

// Add the following lines between <configuration> </configuration>

<configuration>

<property>

<name>mapreduce.framework.name</name>

<value>yarn</value>

</property>

</configuration>


```
GNU nano 7.2 mapred-site.xml
<configuration>
<property>
<name>mapred.job.tracker</name>
<value>localhost:9001</value>
</property>
<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=$HADOOP_HOME</value>
</property>
<property>
<name>mapreduce.application.classpath</name>
<value>$HADOOP_MAPRED_HOME/share/hadoop/mapreduce/*:$HADOOP_MAPRED_HOME/share/h
```

Step 8 – Format Hadoop File System

```
$ cd /usr/local/hadoop/hadoop-2.8.4/bin
```

```
$ hadoop namenode -format
```

Step 9 - Start Hadoop

```
$ cd /usr/local/hadoop/hadoop-2.8.4/sbin
```

```
// Starting dfs services
```

```
$ start-dfs.sh
```

```
// Starting mapreduce services
```

```
$ start-yarn.sh
```

```

haresh@fedora:~$ sshd service start
sshd re-exec requires execution with an absolute path
haresh@fedora:~$ sshd service start^C
haresh@fedora:~$ service sshd restart
Redirecting to /bin/systemctl restart sshd.service
haresh@fedora:~$ start-all.sh
WARNING: Attempting to start all Apache Hadoop daemons as haresh in 10 seconds.
WARNING: This is not a recommended production deployment configuration.
WARNING: Use CTRL-C to abort.
Starting namenodes on [localhost]
Starting datanodes
Starting secondary namenodes [fedora]
Starting resourcemanager
Starting nodemanagers
haresh@fedora:~$

```

\$ jps

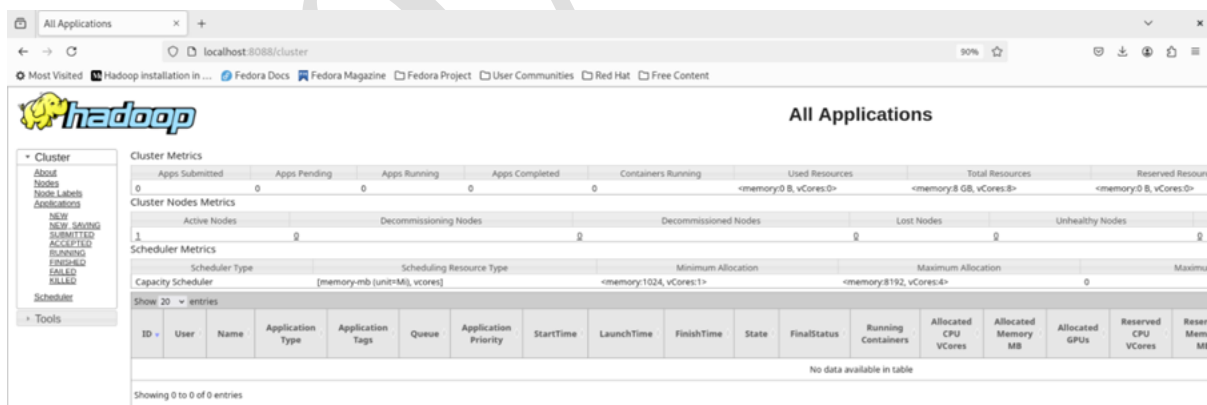
```

haresh@fedora:~$ jps
3987 NameNode
4467 SecondaryNameNode
4699 ResourceManager
4843 NodeManager
4205 DataNode
5247 Jps
haresh@fedora:~$

```

Step 10 - Check Hadoop through web UI

Go to browser type <http://localhost:8088> – All Applications Hadoop Cluster



The screenshot shows the Hadoop web UI at localhost:8088. The page title is 'All Applications'. On the left, there is a sidebar with a 'Cluster' menu and a 'Tools' menu. The main content area displays various metrics and a table of applications.

Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Used Resources	Total Resources	Reserved Resources
0	0	0	0	0	<memory:0 B, vCores:0>	<memory:8 GB, vCores:8>	<memory:0 B, vCores:0>

Cluster Nodes Metrics

Active Nodes	Decommissioning Nodes	Decommissioned Nodes	Lost Nodes	Unhealthy Nodes
1	0	0	0	0

Scheduler Metrics

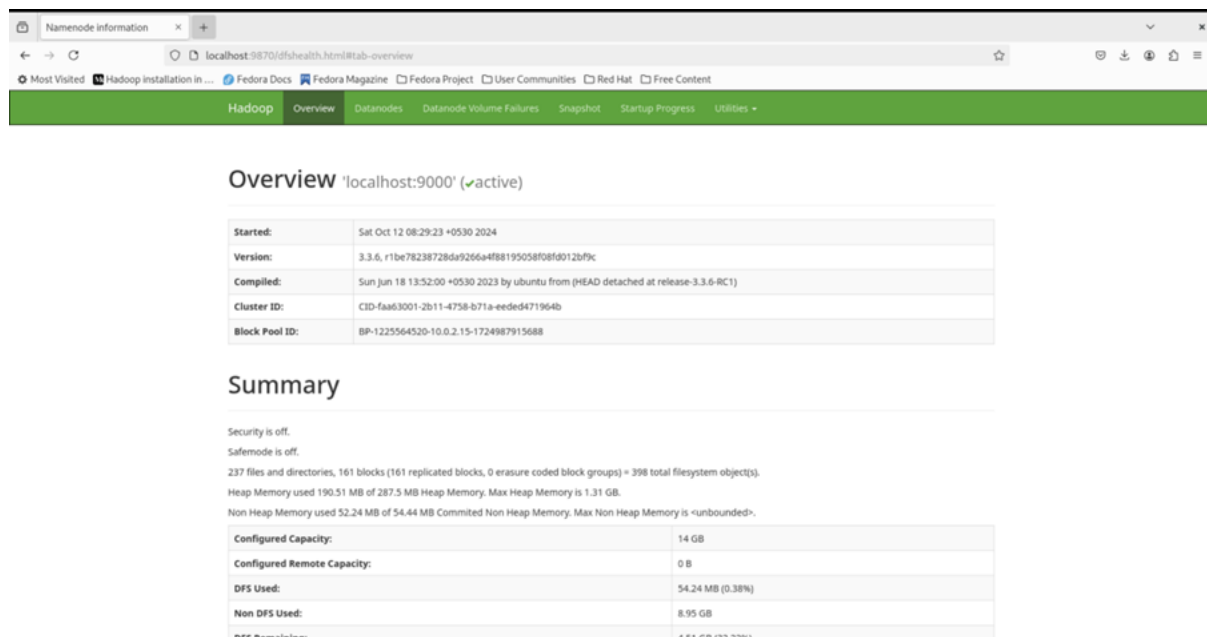
Scheduler Type	Scheduling Resource Type	Minimum Allocation	Maximum Allocation	Maximum
Capacity Scheduler	(memory-mb (unit=Mi), vcores)	<memory:1024, vCores:1>	<memory:8192, vCores:4>	0

Applications Table

ID	User	Name	Application Type	Application Tags	Queue	Application Priority	StartTime	LaunchTime	FinishTime	State	FinalStatus	Running Containers	Allocated CPU V-Cores	Allocated Memory MB	Allocated GPUs	Reserved CPU V-Cores	Reserved Memory MB
No data available in table																	

Showing 0 to 0 of 0 entries

Go to browser type `http://localhost:50070` – Hadoop Namenode



The screenshot shows a web browser window with the address bar displaying `localhost:9870/dfshealth.html#tab=overview`. The page title is "Namenode Information". The browser's address bar shows the URL `localhost:9870/dfshealth.html#tab=overview`. The page has a green header bar with the "Hadoop" logo and navigation links: Overview, Datanodes, Datanode Volume Failures, Snapshot, Startup Progress, and Utilities. The main content area is titled "Overview 'localhost:9000' (✓active)". Below this title is a table with the following information:

Started:	Sat Oct 12 08:29:23 +0530 2024
Version:	3.3.6, r1be78238728da9266a4f88195058f08f012b9c
Compiled:	Sun Jun 18 13:52:00 +0530 2023 by ubuntu from (HEAD detached at release-3.3.6-RC1)
Cluster ID:	CID-faa63001-2b11-4758-b71a-eeed471964b
Block Pool ID:	BP-1225564520-10.0.2.15-1724987915688

Below the table is a "Summary" section. It contains the following text:

Security is off.
Safemode is off.
237 files and directories, 161 blocks (161 replicated blocks, 0 erasure coded block groups) = 398 total filesystem object(s).
Heap Memory used 190.51 MB of 287.5 MB Heap Memory. Max Heap Memory is 1.31 GB.
Non Heap Memory used 52.24 MB of 54.44 MB Committed Non Heap Memory. Max Non Heap Memory is "unbounded".

Below the text is a table with the following information:

Configured Capacity:	14 GB
Configured Remote Capacity:	0 B
DFS Used:	54.24 MB (0.38%)
Non DFS Used:	8.95 GB
DFS Remaining:	8.51 GB (60.62%)

Step 11 - Stop Hadoop

```
$ stop-dfs.sh
```

```
$ stop-yarn.sh
```

RESULT:

Hadoop has been successfully installed and nodes have been successfully created.