Tribhuvan University

Institute of Engineering

Thapathali Campus, Thapathali

Big Data Technologies Assignment 2

Submitted By:

Name: Raman Bhattarai

Roll No.: THA077BEI033

Submitted To:

Department of Electronics and Computer Engineering

Date: Jan 17, 2025

Install hadoop-3.4.0 in your OS environment you prepared in your Assignment 1 section.

Objectives:

- To install Hadoop in Ubuntu OS

Introduction:

Hadoop is an open-source framework designed to store and process massive amounts of data efficiently. It provides a distributed computing environment, allowing for scalable and fault-tolerant data processing across clusters of computers. Developed originally by Apache Software Foundation, Hadoop has become a cornerstone technology in the big data ecosystem.

The key components of Hadoop file system are:

1. HDFS (Hadoop Distributed File System): A scalable file system for storing large datasets.
2. MapReduce: A programming model for parallel data processing.
3. YARN (Yet Another Resource Negotiator): A resource management system for managing cluster resources.
4. Hadoop Common: Libraries and utilities that support the other components.

The key advantages of the Hadoop system are as follows:

1. Scalability: Manages massive datasets by adding more nodes to the cluster as needed.
2. Fault Tolerance: Ensures data availability by replicating it across multiple nodes, even in case of node failures.
3. Cost Efficiency: Lowers infrastructure expenses by utilizing commodity hardware.
4. Data Security: Offers tools for securely storing and processing data.
5. Parallel Processing: Enhances data analysis speed through distributed computing.

The common use cases of Hadoop system are:

- Data Storage: Distributed storage of large datasets, such as logs or sensor data.
- Big Data Analytics: Processing and analyzing huge datasets for insights.
- Machine Learning Pipelines: Training and deploying models on large datasets.
- Data Warehousing: Supporting business intelligence workflows.

System Specifications:

The hardware and software specification of Virtual Machine system in VirtualBox is given below.

Hardware:

- CPU: 3 cores
- RAM: 4 GB
- Storage: 25 GB

Software:

- Operating System: Ubuntu 24.04.1
- Virtualization Platform: Oracle Virtual Machine (VM)

These specifications are sufficient to create an environment suitable for running Hadoop.

Hadoop Setup: Installation, Configuration, and Launching Services

1. Download and install JDK (Prerequisites)

```
Raman@Ubuntu:~$ sudo apt install openjdk-11-jdk
```

```
Raman@Ubuntu:~$ java -version
openjdk version "11.0.25" 2024-10-15
OpenJDK Runtime Environment (build 11.0.25+9-p
OpenJDK 64-Bit Server VM (build 11.0.25+9-post
 sharing)
```

2. Install SSH (Secure Shell)

```
Raman@Ubuntu:~$ sudo apt-get install ssh
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
ssh is already the newest version (1:9.6p1-3ubuntu13.5).
0 upgraded, 0 newly installed, 0 to remove and 153 not upgraded.
```

   - SSH is a protocol that allows secure remote login and communication between computers over a network.

3. Setting up environment variables (Prerequisites)
   a. Open .bashrc file

```
Raman@Ubuntu:~$ sudo nano .bashrc
```

   b. Set the JAVA_HOME and HADOOP_HOME Environment Variable
      i. JAVA_HOME: Points to the installation directory of Java.
      ii. HADOOP_HOME: Points to the installation directory of Hadoop.

iii. PATH: Adds the bin and sbin directory of Java and Hadoop to the system's path, making Java and Hadoop commands accessible.
iv. HADOOP_MAPRED_HOME: Points to the MapReduce module's directory.
v. YARN_HOME: Points to the YARN (Yet Another Resource Negotiator) directory.
vi. HADOOP_CONF_DIR: Points to the configuration files directory (etc/hadoop).
vii. HADOOP_COMMON_LIB_NATIVE_DIR: Specifies the directory for native Hadoop libraries.
viii. HADOOP_OPTS: Configures Java library paths for Hadoop.
ix. HADOOP_STREAMING: Path to the Hadoop streaming JAR for MapReduce jobs.
x. HADOOP_LOG_DIR: Directory where Hadoop logs are stored.
xi. PDSH_RCMD_TYPE: Configures Hadoop to use ssh for remote command execution.

```
# Set JAVA_HOME
#Hadoop Related Options
export JAVA_HOME=/usr/lib/jvm/java-11-openjdk-amd64
export PATH=$PATH:/usr/lib/jvm/java-11-openjdk-amd64/bin
export HADOOP_HOME=~/hadoop-3.4.0/
export PATH=$PATH:$HADOOP_HOME/bin
export PATH=$PATH:$HADOOP_HOME/sbin
export HADOOP_MAPRED_HOME=$HADOOP_HOME
export YARN_HOME=$HADOOP_HOME
export HADOOP_CONF_DIR=$HADOOP_HOME/etc/hadoop
export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/lib/native
export HADOOP_OPTS="-Djava.library.path=$HADOOP_HOME/lib/native"
export HADOOP_STREAMING=$HADOOP_HOME/share/hadoop/tools/lib/hadoop-stre
export HADOOP_LOG_DIR=$HADOOP_HOME/logs
export PDSH_RCMD_TYPE=ssh
```

c. Apply changes in .bashrc file

```
Raman@Ubuntu:~$ source .bashrc
```

4. Download Hadoop
   a. Download the Hadoop 3.4.0 tar file

```
Raman@Ubuntu:~$ wget https://downloads.apache.org/hadoop/common/hadoop-3.4.0/hadoop-3.4.0.tar.gz
```

   b. Extract the tar file

```
Raman@Ubuntu:~$ tar -xzf hadoop-3.4.0.tar.gz
```

   c. Open hadoop-env.sh and set path for JAVA_HOME

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

5. Configure core-site.xml
    a. Open core-site.xml

```
Raman@Ubuntu:~/hadoop-3.4.0/etc/hadoop$ sudo nano core-site.xml
```

    b. Add following configuration

```
  GNU nano 7.2                                        core-site.xml *

<configuration>
 <property>
 <name>fs.defaultFS</name>
 <value>hdfs://localhost:9000</value>  </property>
 <property>
<name>hadoop.proxyuser.dataflair.groups</name> <value>*</value>
 </property>
 <property>
<name>hadoop.proxyuser.dataflair.hosts</name> <value>*</value>
 </property>
 <property>
<name>hadoop.proxyuser.server.hosts</name> <value>*</value>
 </property>
 <property>
<name>hadoop.proxyuser.server.groups</name> <value>*</value>
 </property>
</configuration>
```

        i. fs.defaultFS: Sets the default filesystem to HDFS, accessible at hdfs://localhost:9000.
        ii. hadoop.proxyuser.*: Configures proxy user permissions, allowing unrestricted access (*) for specified hosts and groups.
6. Configure hdfs-site.xml
    a. Open hdfs-site.xml

```
Raman@Ubuntu:~/hadoop-3.4.0/etc/hadoop$ sudo nano hdfs-site.xml
```

    b. Add following configuration

```
  GNU nano 7.2                                        hdfs-site.xml *

<configuration>
 <property>
 <name>dfs.replication</name>
 <value>1</value>
 </property>
</configuration>
```

        i. dfs.replication: Specifies the number of replicas for each HDFS block. Setting this to 1 is appropriate for a single-node setup to save storage space.
7. Configure mapred-site.xml
    a. Open mapred-site.xml

```
Raman@Ubuntu:~/hadoop-3.4.0/etc/hadoop$ sudo nano mapred-site.xml
```

b. Add following configuration

```
  GNU nano 7.2                                  mapred-site.xml *
<!-- Put site-specific property overrides in this file. -->


<configuration>
 <property>
 <name>mapreduce.framework.name</name>  <value>yarn</value>
 </property>
 <property>
 <name>mapreduce.application.classpath</name>

<value>$HADOOP_MAPRED_HOME/share/hadoop/mapreduce/*:$HADOOP_MAPRED_HOME/share/hadoop/map
 </property>
</configuration>
```

i. mapreduce.framework.name: Specifies the framework for MapReduce jobs. Setting it to yarn tells Hadoop to use the YARN (Yet Another Resource Negotiator) framework for resource management.

ii. mapreduce.application.classpath: Defines the classpath for MapReduce applications, ensuring they can locate required libraries.

8. Configure yarn-site.xml

a. Open yarn-site.xml

```
Raman@Ubuntu:~/hadoop-3.4.0/etc/hadoop$ sudo nano yarn-site.xml
```

b. Add following configuration

```
  GNU nano 7.2                                  yarn-site.xml *

<configuration>
 <property>
 <name>yarn.nodemanager.aux-services</name>
 <value>mapreduce_shuffle</value>
 </property>
 <property>
 <name>yarn.nodemanager.env-whitelist</name>

<value>JAVA_HOME,HADOOP_COMMON_HOME,HADOOP_HDFS_HOME,HADOOP_CONF_DIR,CLASSPATH_PREP END_DISTCA>
 </property>
</configuration>
```

i. yarn.nodemanager.aux-services: Enables the mapreduce_shuffle service, which is required for MapReduce to shuffle data during job execution.

ii. yarn.nodemanager.env-whitelist: Specifies the environment variables that should be available to YARN containers, ensuring that necessary paths and settings (like JAVA_HOME, HADOOP_COMMON_HOME, etc.) are accessible for running YARN applications.

9. Configuring ssh for Hadoop

a. Test SSH connection

```
Raman@Ubuntu:~/hadoop-3.4.0/etc/hadoop$ ssh localhost
```

b. Generate SSH key pair to enable passwordless ssh access

```
Raman@Ubuntu:~$ ssh-keygen -t rsa -P '' -f ~/.ssh/id_rsa
```

c. Add Public Key to authorized_keys

```
Raman@Ubuntu:~$ cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys
```

d. Set Correct Permissions for authorized_keys

```
Raman@Ubuntu:~$ chmod 0600 ~/.ssh/authorized_keys
```

10. Formatting the Hadoop NameNode

```
Raman@Ubuntu:~$ hadoop-3.4.0/bin/hdfs namenode -format
```

11. Configure PDSH(Parallel Distributed Shell) to use SSH

```
Raman@Ubuntu:~$ export PDSH_RCMD_TYPE=ssh
```

12. Start Hadoop Services

```
Raman@Ubuntu:~$ start-all.sh
WARNING: Attempting to start all Apache Hadoop daemons as Raman 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 [Ubuntu]
Ubuntu: Warning: Permanently added 'ubuntu' (ED25519) to the list of known hos
Starting resourcemanager
Starting nodemanagers
```

Challenges and Solutions

During the Hadoop installation and configuration process, several challenges were encountered. Some critical steps were missed during the configuration phase, leading to an incomplete installation and rendering Hadoop non-functional. This issue required multiple reinstallation attempts and extensive consultation of online resources. To address the problem, a step-by-step guide was followed carefully to ensure all configuration steps were completed. Eventually, the problems were resolved, and Hadoop was successfully installed and configured.

Validation

1. Hadoop Version

```
Raman@Ubuntu:~$ hadoop version
Hadoop 3.4.0
Source code repository git@github.com:apache/hadoop.git -r bd8b77f398f626bb77917
83192ee7a5dfaeec760
Compiled by root on 2024-03-04T06:35Z
Compiled on platform linux-x86_64
Compiled with protoc 3.21.12
From source with checksum f7fe694a3613358b38812ae9c31114e
This command was run using /home/Raman/hadoop-3.4.0/share/hadoop/common/hadoop-c
ommon-3.4.0.jar
```

2. Hadoop NameNode Web UI

Overview 'localhost:9000' (✔active)

Started:	Fri Jan 17 07:12:28 +0000 2025
Version:	3.4.0, rbd8b77f398f626bb7791783192ee7a5dfaeec760
Compiled:	Mon Mar 04 06:35:00 +0000 2024 by root from (HEAD detached at release-3.4.0-RC3)
Cluster ID:	CID-270fe6de-fef8-4e8f-baa6-251915a96a42
Block Pool ID:	BP-1225248034-127.0.1.1-1737097741842

Summary

Security is off.

Safemode is off.

1 files and directories, 0 blocks (0 replicated blocks, 0 erasure coded block groups) = 1 total filesystem object(s).

Heap Memory used 109.26 MB of 256 MB Heap Memory. Max Heap Memory is 980 MB.

Non Heap Memory used 53.62 MB of 56.88 MB Commited Non Heap Memory. Max Non Heap Memory is <unbounded>.

Configured Capacity:	24.44 GB
Configured Remote Capacity:	0 B
DFS Used:	24 KB (0%)
Non DFS Used:	8.65 GB
DFS Remaining:	14.52 GB (59.42%)
Block Pool Used:	24 KB (0%)
DataNodes usages% (Min/Median/Max/stdDev):	0.00% / 0.00% / 0.00% / 0.00%
Live Nodes	1 (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	Fri Jan 17 07:12:28 +0000 2025
Last Checkpoint Time	Fri Jan 17 07:09:02 +0000 2025
Last HA Transition Time	Never
Enabled Erasure Coding Policies	RS-6-3-1024k

NameNode Journal Status

Current transaction ID: 3

Journal Manager	State
FileJournalManager(root=/tmp/hadoop-Raman/dfs/name)	EditLogFileOutputStream(/tmp/hadoop-Raman/dfs/name/current/edits_inprogress_0000000000000000003)

NameNode Storage

Storage Directory	Type	State
/tmp/hadoop-Raman/dfs/name	IMAGE_AND_EDITS	Active

DFS Storage Types

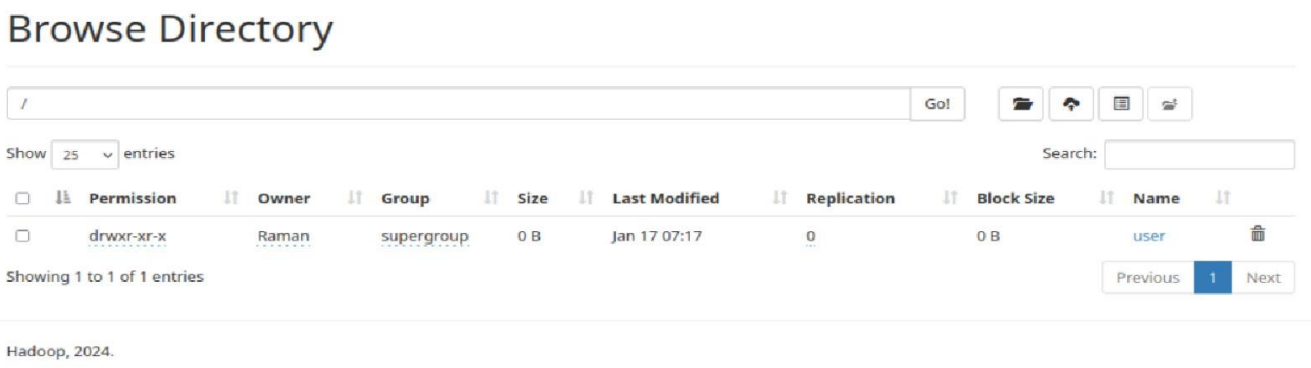
Storage Type	Configured Capacity	Capacity Used	Capacity Remaining	Block Pool Used	Nodes In Service
DISK	24.44 GB	24 KB (0%)	14.52 GB (59.42%)	24 KB	1

Hadoop, 2024.

3. Making a directory

```
Raman@Ubuntu:~$ hadoop fs -mkdir /user
```

```
Raman@Ubuntu:~$ hadoop fs -ls /
Found 1 items
drwxr-xr-x   - Raman supergroup          0 2025-01-17 15:00 /user
```



Conclusion:

In this assignment, I installed and configured Apache Hadoop 3.4.0 on an Ubuntu virtual machine in VirtualBox. This hands-on experience involved setting up the distributed data processing framework, configuring core Hadoop components, and starting services like NameNode, DataNode, ResourceManager, and NodeManager.

Hadoop is crucial for big data applications, providing a scalable system for storing and processing large datasets through its HDFS and YARN components. It enables efficient data storage, resource management, and job scheduling, making it essential for big data analytics and machine learning.

This exercise enhanced my understanding of the complexities involved in setting up and maintaining a Hadoop ecosystem for big data applications.