**To install Hadoop 3.3.6 on Ubuntu, along with OpenJDK and SSH (for Hadoop's distributed capabilities), follow these detailed steps.**

Requirements:

1. Ubuntu 22.04
2. 8GB RAM & 4 Core Processor (Minimum)

### **Step 1: Update and Install Java (OpenJDK)**

Hadoop requires Java. We'll install OpenJDK 11.

sudo apt update

sudo apt install openjdk-11-jdk -y

Verify the Java installation:

java -version

This should show the installed Java version, e.g., OpenJDK version 11.

echo $JAVA\_HOME

This command should display like:

/usr/lib/jvm/java-11-openjdk-amd64

### **Step 2: Install SSH**

Hadoop requires SSH for communication between its nodes.

sudo apt install ssh -y

Now, set up password-less SSH for localhost (to allow Hadoop to work in pseudo-distributed mode):

ssh-keygen -t rsa -P ""

Press: Y

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

chmod 0600 ~/.ssh/authorized\_keys

Test SSH configuration:

ssh localhost

If you're not prompted for a password, the setup is correct.

### **Step 3: Download and Extract Hadoop 3.3.6**

Download Hadoop 3.3.6 from the official Apache website:

sudo chmod -R 777 /opt

cd /opt

wget https://dlcdn.apache.org/hadoop/common/hadoop-3.3.6/hadoop-3.3.6.tar.gz

Extract the downloaded tar file:

tar -xzf hadoop-3.3.6.tar.gz

sudo mv hadoop-3.3.6 /usr/local/hadoop

### **Step 4: Configure Hadoop Environment Variables**

Set up environment variables by editing the .bashrc file:

nano ~/.bashrc

Add the following lines to the end of the file:

# Set Hadoop environment variables

export HADOOP\_HOME=/usr/local/hadoop

export PATH=$PATH:$HADOOP\_HOME/bin:$HADOOP\_HOME/sbin

export JAVA\_HOME=/usr/lib/jvm/java-11-openjdk-amd64

Reload the .bashrc file:

source ~/.bashrc

### **Step 5: Configure Hadoop**

Now, configure the Hadoop environment by editing configuration files.

1. **Configure hadoop-env.sh to set the Java home:**

nano $HADOOP\_HOME/etc/hadoop/hadoop-env.sh

Find the line for JAVA\_HOME and modify it as follows:

export JAVA\_HOME=/usr/lib/jvm/java-11-openjdk-amd64

1. **Configure core-site.xml:**

nano $HADOOP\_HOME/etc/hadoop/core-site.xml

Add the following configuration between <configuration> and </configuration> tags:

<property>

<name>fs.defaultFS</name>

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

</property>

1. **Configure hdfs-site.xml:**

nano $HADOOP\_HOME/etc/hadoop/hdfs-site.xml

Add the following configurations between <configuration> and </configuration> tags:

<property>

<name>dfs.replication</name>

<value>1</value>

</property>

<property>

<name>dfs.name.dir</name>

<value>file:///usr/local/hadoop/hadoop\_data/hdfs/namenode</value>

</property>

<property>

<name>dfs.data.dir</name>

<value>file:///usr/local/hadoop/hadoop\_data/hdfs/datanode</value>

</property>

1. **Configure mapred-site.xml:**

cp $HADOOP\_HOME/etc/hadoop/mapred-site.xml.template $HADOOP\_HOME/etc/hadoop/mapred-site.xml

nano $HADOOP\_HOME/etc/hadoop/mapred-site.xml

Add the following configuration:

<property> <name>yarn.app.mapreduce.am.env</name> <value>HADOOP\_MAPRED\_HOME=/usr/local/hadoop</value> <!-- Adjust this path --> </property> <property> <name>mapreduce.map.env</name> <value>HADOOP\_MAPRED\_HOME=/usr/local/hadoop</value> <!-- Adjust this path --> </property> <property> <name>mapreduce.reduce.env</name> <value>HADOOP\_MAPRED\_HOME=/usr/local/hadoop</value> <!-- Adjust this path --> </property>

1. **Configure yarn-site.xml:**

nano $HADOOP\_HOME/etc/hadoop/yarn-site.xml

Add the following configuration:

<property>

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

<value>mapreduce\_shuffle</value>

</property>

### **Step 6: Format the Hadoop Filesystem**

Now, format the Hadoop filesystem (HDFS):

hdfs namenode -format

### **Step 7: Start Hadoop Services**

You can now start the Hadoop Distributed File System (HDFS) and YARN resource manager:

start-dfs.sh

start-yarn.sh

Verify that all Hadoop daemons are running:

jps

You should see something like:

110480 NodeManager

112194 NameNode

112327 DataNode

112534 SecondaryNameNode

110362 ResourceManager

112908 Jps

### **Step 8: Access the Hadoop Web UI**

* HDFS Web UI: http://localhost:9870
* YARN Web UI: http://localhost:8088

### **Step 9: Test Hadoop Setup**

You can run a simple MapReduce job to test your setup:

hadoop jar $HADOOP\_HOME/share/hadoop/mapreduce/hadoop-mapreduce-examples-3.3.6.jar pi 2 5

This should run a simple Pi estimation job.

**EXAMPLE**

### **Step 1: Create Input Data**

1. **Create a Sample Text File**

First, create a simple text file to use as input for the WordCount job. For example, create a file named input.txt:

echo "Hadoop is great. Hadoop is fast." > input.txt

1. **Create a Directory in HDFS**

Create a directory in HDFS where you will place the input file:

hdfs dfs -mkdir -p /user/$(whoami)/input

1. **Upload the Input File to HDFS**

Now, upload the input.txt file to HDFS:

hdfs dfs -put input.txt /user/$(whoami)/input

### **Step 2: Run the WordCount Job**

Hadoop includes a built-in example for WordCount. You can run it using the following command:

hadoop jar $HADOOP\_HOME/share/hadoop/mapreduce/hadoop-mapreduce-examples-3.3.6.jar wordcount /user/$(whoami)/input /user/$(whoami)/output

In this command:

* /user/$(whoami)/input is the HDFS directory where the input file (input.txt) is stored.
* /user/$(whoami)/output is the directory where the output will be saved (make sure this directory does not already exist).

### **Step 3: View the WordCount Output**

Once the WordCount job has completed successfully, you can view the output by running the following commands:

1. **List the Output Directory in HDFS**

Check the output directory to see the result files:

hdfs dfs -ls /user/$(whoami)/output

You should see files like part-r-00000.

1. **View the Contents of the Output File**

Display the contents of the output file (which contains the word counts):

hdfs dfs -cat /user/$(whoami)/output/part-r-00000

You should see something like this:

Hadoop 2

fast. 1

great. 1

is 2

### **Step 4: Clean Up (Optional)**

If you want to remove the output directory after you've checked the results, you can do so with:

hdfs dfs -rm -r /user/$(whoami)/output

### **Conclusion**

You've now installed Hadoop 3.3.6 on Ubuntu along with OpenJDK and SSH, and set it up to work in pseudo-distributed mode.