**Installation**

1. **Install Linux**
2. **sudo apt update && sudo apt upgrade**
3. **create user:**

**sudo adduser hadoop**

**--passowrd: 0000**

**--login : su - hadoop**

**-- cheange hadoop user to admin**

1. **install ssh:**

**sudo apt update**

**sudo apt upgrade**

**sudo apt install openssh-server**

**sudo apt install openssh-client**

**--sudo systemctl status ssh**

**--sudo apt-get install ssh**

**sudo systemctl enable ssh**

**sudo systemctl start ssh**

**ssh hadoop@localhost -p 22**

**ssh hadoop@127.0.0.1 -p 22**

**ssh hadoop@192.168.220.129 -p 22**

**ssh-keygen -t rsa**

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

**chmod 640 ~/.ssh/authorized\_keys**

**ssh hadoop@localhost**

**--TNlriQax/cWtTNz4H4DF+yQzQF1wk2Q3ZT783DWnN workr 1**

**ls ~/.ssh/id\_rsa\***

**ssh-copy-id hadoop@127.0.0.1**

**sudo ufw allow ssh**

**sudo ufw enable**

**sudo ufw status**

1. **Install java**

**sudo apt update && sudo apt install openjdk-11-jdk**

**java -version**

1. **Install Hadoop HDFS**

**--Step 3: Install Hadoop on Ubuntu**

**--Use the following command to download Hadoop 3.3.4:**

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

**--enter ctrl + shift + v**

**ls**

**--unzip it to a folder on your hard drive:**

**tar -xzf hadoop-3.3.6.tar.gz**

**ls**

**test java:**

**ls /usr/lib/jvm/java-1.11.0-openjdk-amd64**

**--Rename the extracted folder to remove version information:**

**rm hadoop-3.3.6.tar.gz**

**ls**

**test:**

**cd hadoop-3.3.6/sbin**

**cd hadoop-3.3.6/etc/hadoop/**

**ls**

**exit**

**cntl+L**

**nano core-site.xml**

**<configuration>**

**<property>**

**<name>fs.defaultFS</name>**

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

**</property>**

**</configuration>**

**nano hdfs-site.xml**

**<configuration>**

**<property>**

**<name>dfs.replication</name>**

**<value>1</value>**

**</property>**

**<property>**

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

**<value>/hadoop/hdfs/namenode</value>**

**</property>**

**<property>**

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

**<value>/hadoop/hdfs/datanode</value>**

**</property>**

**</configuration>**

**cd -**

**mkdir -p hdfs/namenode**

**mkdir -p hdfs/datanode**

**cd hadoop-3.3.6/etc/hadoop/**

**nano hadoop-env.sh**

**--add in: export java\_home=/usr/lib/jvm/java-1.11.0-openjdk-amd64**

**--add in: export java\_home=/usr/lib/jvm/java-11-openjdk-amd64 --**

**--add in: export java\_home=/usr/lib/jvm/java-8-openjdk-amd64 --**

**nano yarn-site.xml**

**--File**

**cd hadoop-3.3.6/**

**cd bin/**

**./hdfs namenode -format**

**nano capacity-scheduler.xml**

**--File**

**cd ..**

**cd sbin/**

**ls**

**cd hadoop-3.3.6/**

**cd sbin/**

**./start-all.sh**

**./stop-all.sh**

**./start-dfs.sh**

**./stop-dfs.sh**

**http://localhost:9870**

**./start-yarn.sh**

**http://localhost:8088**

1. **Move data in HDFS**

**--create file test.csv**

**cd hadoop-3.3.6/**

**cd bin**

**./hdfs dfs -ls /**

**nano ~/test.csv**

**--insert date test**

**--insert file test in dhsf**

**cd hadoop-3.3.6/**

**./hdfs dfs -put ~/test.csv /**

**./hdfs dfs -ls /**

**--create folder and move text,csv into**

**./hdfs dfs -mkdir /test\_date**

**./hdfs dfs -mv /test.csv /test\_date/test.csv**

**hadoop@ubuntu:~/hadoop-3.3.6/bin$ ./hdfs dfs -mv yellow\_tripdata\_2022-02.parquet /test\_date/yellow\_tripdata\_2022-02.parquet**

**./hdfs dfs -ls /**

**--show in hadoop browse**

**--download & install apache spark over hadoop and yarn**

**--link https://www.apache.org/dyn/closer.lua/spark/spark-3.4.1/spark-3.4.1-bin-hadoop3.tgz**

**wget https://dlcdn.apache.org/spark/spark-3.4.1/spark-3.4.1-bin-hadoop3.tgz**

**ls**

**tar -xzf spark-3.4.1-bin-hadoop3.tgz**

**ls**

**rm spark-3.4.1-bin-hadoop3.tgz**

**--------------------**

**cd spark-3.4.1-bin-hadoop3/conf/**

**ls**

**cp spark-env.sh.template spark-env.sh**

**nano spark-env.sh**

**--add in file**

**export HADOOP\_CONF\_DIR=/home/hadoop-3.3.6/hadoop/etc/hadoop/**

**export YARN\_CONF\_DIR=/home/hadoop-3.3.6/hadoop/etc/hadoop/**

**export PYSPARK\_PYTHON=python3**

**export HADOOP\_CONF\_DIR=/home/hadoop-3.3.6/hadoop/etc/hadoop/**

**export YARN\_CONF\_DIR=/home/hadoop-3.3.6/hadoop/etc/hadoop/**

**export PYSPARK\_PYTHON=python3**

**hadoop-3.3.6/etc/hadoop/**

**----**

**cp workers.template workers**

**nano workers**

**--add**

**localhost**

**--192.168.220.130**

**--192.168.100.22**

**---------**

**--running hadoop and yarn**

**cd spark-3.4.1-bin-hadoop3/**

**cd bin/**

**ls**

**----done----**

**------------------------------------------------------------------------**

**Monitor Your Spark Applications**

**When you submit a job, Spark Driver automatically starts a web UI on port 4040 that displays information about the application. However, when execution is finished, the Web UI is dismissed with the application driver and can no longer be accessed.**

**Spark provides a History Server that collects application logs from HDFS and displays them in a persistent web UI. The following steps will enable log persistence in HDFS:**

**Edit $SPARK\_HOME/conf/spark-defaults.conf and add the following lines to enable Spark jobs to log in HDFS:**

**File: $SPARK\_HOME/conf/spark-defaults.conf**

**spark.eventLog.enabled true**

**spark.eventLog.dir hdfs://node-master:9000/spark-logs**

**Create the log directory in HDFS:**

**hdfs dfs -mkdir /spark-logs**

**Configure History Server related properties in $SPARK\_HOME/conf/spark-defaults.conf:**

**File: $SPARK\_HOME/conf/spark-defaults.conf**

**spark.history.provider org.apache.spark.deploy.history.FsHistoryProvider**

**spark.history.fs.logDirectory hdfs://localhost:9000/spark-logs**

**spark.history.fs.update.interval 10s**

**spark.history.ui.port 18080**

**You may want to use a different update interval than the default 10s. If you specify a bigger interval, you will have some delay between what you see in the History Server and the real time status of your application. If you use a shorter interval, you will increase I/O on the HDFS.**

**Run the History Server:**

**$SPARK\_HOME/sbin/start-history-server.sh**

**Repeat steps from previous section to start a job with spark-submit that will generate some logs in the HDFS:**

**Access the History Server by navigating to http://localhost:18080 in a web browser:**

**spark-3.4.1-bin-hadoop3/sbin/start-history-server.sh**

**spark-sql> set spark.SQL.warehouse.dir**

**spark-sql> !hdfs dfs -ls /user/**

1. **--Run hadoop & yarn & spark**

**hadoop-3.3.6/sbin/start-dfs.sh**

**hadoop-3.3.6/sbin/start-yarn.sh**

**-- or hadoop-3.3.6/sbin/start-all.sh**

**hadoop@ubuntu:~$ schematool -initSchema -dbType derby;**

**spark-3.4.1-bin-hadoop3/bin/pyspark --master yarn --queue dev --name py\_app**

**spark-3.4.1-bin-hadoop3/bin/spark-shell --master yarn --queue dev --name query\_app**

**spark-3.4.1-bin-hadoop3/bin/spark-sql --master yarn --queue prod --name query\_app**

**http://localhost:9870**

**http://localhost:8088**

**ssh hadoop@192.168.220.129 -p 22**

1. **--Run delta & spark sql**

**spark-3.4.1-bin-hadoop3/bin/spark-sql --master yarn --queue prod --name query\_app** --packages io.delta:delta-core\_2.12:2.4.0 --conf "spark.sql.extensions=io.delta.sql.DeltaSparkSessionExtension" --conf "spark.sql.catalog.spark\_catalog=org.apache.spark.sql.delta.catalog.DeltaCatalog"

**--get data**

**df=spark.read.csv("hdfs://localhost:9000/test\_date/test.csv", header=True)**

**df.show(5)**

**df.write.mode("Overwrite").parquet("hdfs://localhost:9000/test\_date/test.parquet")**

**df.read.parquet("hdfs://localhost:9000/test\_date/test.parquet")**

**--spark-sql**

**create table Mytable (id int,name string);**

**insert into mytable(1,'muhammad');**

**select \* from mytable;**

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**Downloading Hive**

**$ cd Downloads**

**$ ls**

**apache-hive-0.14.0-bin.tar.gz**

**$ tar zxvf apache-hive-0.14.0-bin.tar.gz**

**$ ls**

**--------------------------**

**hadoop-3.3.6/sbin/stop-dfs.sh**

**hadoop-3.3.6/sbin/stop-yarn.sh**

**spark-submit --conf spark.dynamicAllocation.enabled=false**

**spark-submit --deploy-mode cluster --master yarn \**

**--driver-memory 3g --executor-memory 3g \**

**--num-executors 2 --executor-cores 2 \**

**--conf spark.dynamicAllocation.enabled=false \**

**readcsv.py**

**---------------------**

**export PYSPARK\_SUBMIT\_ARGS='--master yarn --deploy -mode cluster pyspark-shell'**

**--cluster mode - the spark driver is run in the spark master node**

**--client mode - the spark driver is run from the client side where the interactive shell is run.**

**------------------------**

yarn application -kill application\_1691006486173\_0001

**---------------------------------**

**--open port**

**sudo ufw allow 22/tc**

**sudo ufw status verbose**

**sudo ufw allow http**

**sudo ufw allow https**

**---------------------------------------------------**

**sudo mv /home/muhammad/Downloads/hadoop-3.3.6.tar.gz /home/hadoop/bin/**

**yarn application -list**

**yarn application -kill <jobid>**

**sudo apt -o Dpkg::Options::="--force-overwrite" --fix-broken install**

**sudo apt-get update && sudo apt-get upgrade**

**sudo dpkg --configure -a**

**sudo apt-get clean && sudo apt-get autoremove**

**sudo reboot**

**cd Downloades**

**--sudo usermod -aG hadoop**

**display:**

**sudo apt install neofetch**

**nano /etc/default/grub**

**GRUB\_CMDLINE\_LINUX\_DEFAULT="quiet splash video=hyperv\_fb:1720x880"**

**"quiet splash nomodset"**

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**Install Spark History Server**

**# Enable to store the event log**

**spark.eventLog.enabled true**

**#Location where to store event log**

**spark.eventLog.dir file:///user/spark/spark-events**

**#Location from where history server to read event log**

**spark.history.fs.logDirectory** [**file:///user/spark/spark-events**](file:///user/spark/spark-events)

**$SPARK\_HOME/sbin/start-history-server.sh**

**/spark-3.4.1-bin-hadoop3/sbin$ spark-class org.apache.spark.deploy.history.HistoryServer**

**error failed to launch: nice -n 0 /home/hadoop//bin/spark-class org.apache.spark.deploy.history.HistoryServer -v**

**solution**

**hadoop@ubuntu:/tmp$ mkdir spark-events**

The file /etc/spark/conf/spark-defaults.conf  is not used by Oozie Spark Actions by default.  In order to tell Oozie Spark Action to use this file, I had to add this to /etc/oozie/conf/oozie-site.xml

<property>

<name>oozie.service.SparkConfigurationService.spark.configurations</name>

<value>\*=/etc/spark/conf/</value>

</property>

Now I can see the logs in the Spark History Server.  I wonder why this should be the default.

**spark.eventLog.enabled true**

**spark.eventLog.dir hdfs://localhost:9000/spark/events**

**spark.history.provider org.apache.spark.deploy.history.FsHistoryProvider**

**spark.history.fs.logDirectory hdfs://localhost:9000/spark/events**

**spark.history.fs.update.interval 10s**

**spark.history.ui.port 18080**

**spark.yarn.services org.apache.spark.deploy.yarn.history.YarnHistoryService**

**the same instllation all node and master---**

**--- copy machin and cheange name and ip and ssh**

**master 192.168.220.129**

**worker 192.168.220.130**

**add ip node in file workers**

**cd hadoop-3.3.6/etc/hadoop/**

**cd /bin**

**nano workers**

**--add ip all node**

**--------------------**

**add worker**

**ls -a**

**cd .ssh**

**ls**

**nano authorized\_keys --add workers Kye in master and workers**

**------**

**cd hadoop-3.3.6/etc/hadoop/**

**add ip maser in:**

**nano core-site.xml**

**<configuration>**

**<property>**

**<name>fs.defaultFS</name>**

**<value>hdfs://192.168.220.129:9000</value>**

**</property>**

**</configuration>**

**--add property in woekrs and master**

**nano yarn-site.xml**

**<property>**

**<name>yarn.resourcemanager.hostname</name>**

**<value>192.168.220.129</value>**

**<description>The hostname of the RM.</description>**

**</property>**

**<property>**

**<name>yarn.resourcemanager.address</name>**

**<value>192.168.220.129:8032</value>**

**<description>The hostname of the RM.</description>**

**</property>**

**add java path in:**

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

**hadoop@ubuntworker:~/hadoop-3.3.6/etc/hadoop$ nano mapred-env.sh**

**hadoop@ubuntworker:~/hadoop-3.3.6/etc/hadoop$ nano yarn-env.sh**

**hadoop@ubuntworker:~/hadoop-3.3.6/etc/hadoop$ nano hadoop-env.sh**

**<configuration>**

**<property>**

**<name>fs.default.name</name>**

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

**</property>**

**</configuration>**

**-----------------------------------**

**--cheange in master and node mapred-site.xml**

**cd hadoop-3.3.6/etc/hadoop/**

**cp etc/hadoop/mapred-site.xml.template etc/hadoop/mapred-site.xml**

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

**<property>**

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

**<value>yarn</value>**

**</property>**

**--start master only**

**--Run**

**hadoop-3.3.6/sbin/start-dfs.sh**

**hadoop-3.3.6/sbin/start-yarn.sh**

**-- or hadoop-3.3.6/sbin/start-all.sh**

**spark-3.4.1-bin-hadoop3/bin/pyspark --master yarn --queue dev --name py\_app**

**error**

**ls: RPC response has invalid length**

**sudo lsof -i -P -n | grep LISTEN**

sudo netstat -lpten | grep java

**I faced the same issue today and would like to note it here if anyone faced a similar issue. A quick command jps show me that the NameNode process is not there - although there is no warning or error show up.**

**As I discovered in the .log file of the NameNode in Hadoop, there was a java.net.BindException: Problem binding to [localhost:9000], which made me think that the port 9000 is used by another process. I use the command from this source to check open ports, indeed it is used by a python process (I ran only PySpark at that time). (sudo lsof -i -P -n | grep LISTEN for anyone needs by the way)**

**The solution is pretty straightforward: change the port number in fs.defaultFS field in etc/core-site.xml to another port that is not in used (mine is 9900).**