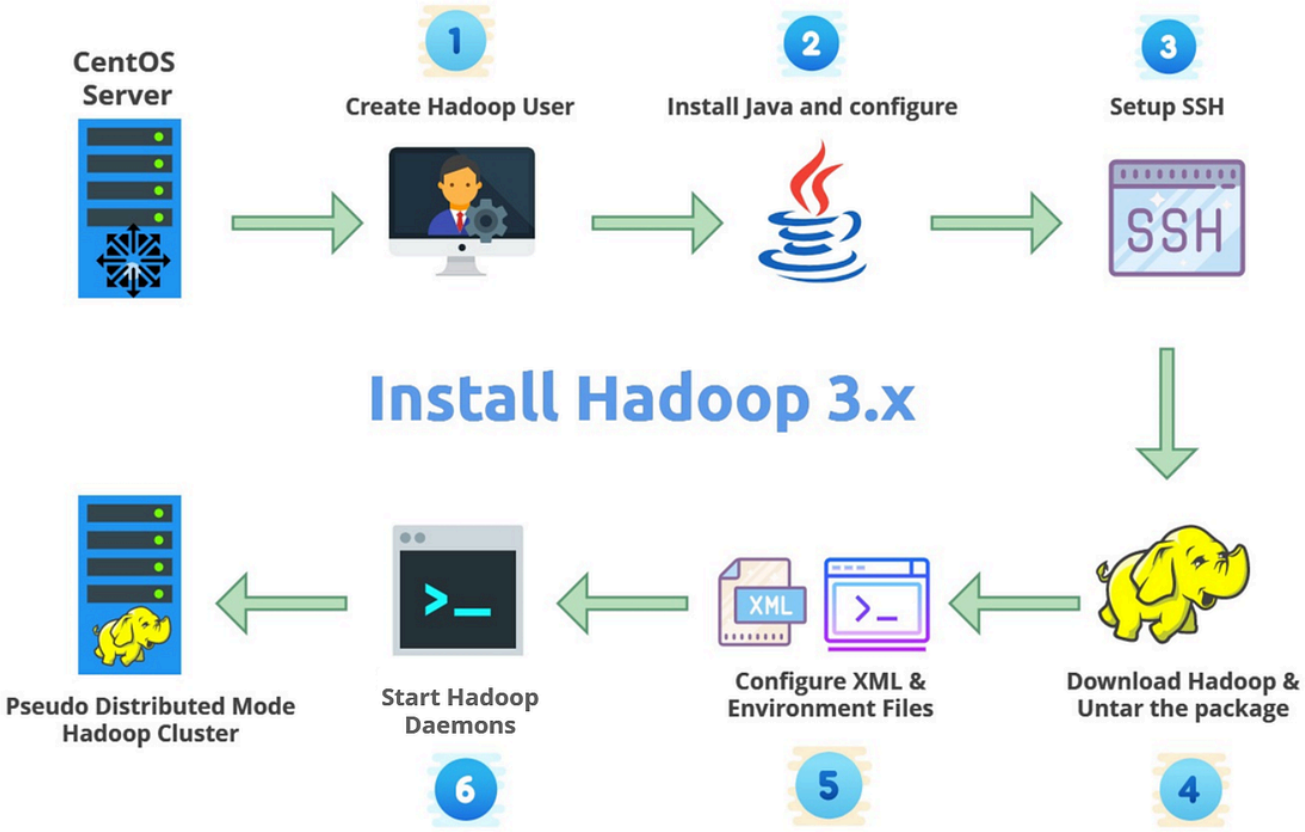
Setup Hadoop on Single Node Cluster



Links:
https://www.digitalocean.com/community/tutorials/how-to-install-hadoop-in-stand-alone-mode-on-ubuntu-20-04

https://www.youtube.com/watch?v=HAlv7uGi6OE

Prerequisites

- Linux OS (e.g., Ubuntu) or Windows with WSL or VM running Linux
- Java JDK 8 or later installed
- SSH server installed and running (for Hadoop daemon communication)
- Hadoop binary downloaded (choose stable version from Apache Hadoop Mirror)

Step 1: Install Java

Verify Java installation:

```
java -version
```

If not installed, install OpenJDK 8 on Ubuntu:

```
sudo apt update
sudo apt install openjdk-8-jdk -y
```

Set JAVA_HOME in `.bashrc` or `.profile`:

```
export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
export PATH=$PATH:$JAVA_HOME/bin
```

Refresh environment variables:

```
source ~/.bashrc
```

Step 2: Install and Extract Hadoop

Download Hadoop (example version 3.3.6) from:
https://hadoop.apache.org/releases.html
Extract tarball to `/usr/local/hadoop` or your preferred directory:

```
tar -xzf hadoop-3.3.6.tar.gz
sudo mv hadoop-3.3.6 /usr/local/hadoop
```

Update environment variables (`~/.bashrc`):

```
export HADOOP_HOME=/usr/local/hadoop
export PATH=$PATH:$HADOOP_HOME/bin:$HADOOP_HOME/sbin
```

Reload:

```
source ~/.bashrc
```

Step 3: Configure SSH for Hadoop

Hadoop requires SSH access to manage daemons, so configure passwordless SSH for localhost:

```
ssh-keygen -t rsa -P "" -f ~/.ssh/id_rsa
cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys
chmod 0600 ~/.ssh/authorized_keys
```

Test:

```
ssh localhost
```

If it logs in without password, SSH is configured.

Step 4: Configure Hadoop XML Files

Edit core Hadoop config files in `$HADOOP_HOME/etc/hadoop/`:

1. `hadoop-env.sh`

Set your java home:

```
export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
```

2. core-site.xml

Add:

```
<configuration>
  <property>
    <name>fs.defaultFS</name>
    <value>hdfs://localhost:9000</value>
  </property>
</configuration>
```

3. hdfs-site.xml

Set replication factor to 1 (single node):

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

<value>file:///usr/local/hadoop/hadoopdata/hdfs/namenode<
/value>
  </property>
  <property>
    <name>dfs.datanode.data.dir</name>

<value>file:///usr/local/hadoop/hadoopdata/hdfs/datanode<
/value>
  </property>
</configuration>
```

Create those directories:

```
mkdir -p /usr/local/hadoop/hadoopdata/hdfs/namenode
mkdir -p /usr/local/hadoop/hadoopdata/hdfs/datanode
```

4. mapred-site.xml

Copy template first:

```
cp mapred-site.xml.template mapred-site.xml
```

Add:

```
<configuration>
  <property>
    <name>mapreduce.framework.name</name>
    <value>yarn</value>
  </property>
</configuration>
```

5. yarn-site.xml

Add:

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

Step 5: Format the Hadoop NameNode

Initialize HDFS:

```
hdfs namenode -format
```

Step 6: Start Hadoop Daemons

Start HDFS:

```
start-dfs.sh
```

Start YARN:

```
start-yarn.sh
```

Check processes with:

```
jps
```

You should see processes like:

- NameNode
- DataNode
- ResourceManager
- NodeManager
- SecondaryNameNode

Step 7: Access Hadoop UI

- NameNode UI: http://localhost:9870/
- ResourceManager UI: http://localhost:8088/

Step 8: Test with Sample Job

Copy sample input and run a MapReduce example:

```
mkdir input
echo "Hello Hadoop" > input/file1.txt
hdfs dfs -mkdir /user
hdfs dfs -mkdir /user/$(whoami)
hdfs dfs -put input/file1.txt /user/$(whoami)/

hadoop jar
$HADOOP_HOME/share/hadoop/mapreduce/hadoop-mapreduce-exam
ples-3.3.6.jar grep /user/$(whoami)/
/user/$(whoami)/output 'Hadoop'
hdfs dfs -cat /user/$(whoami)/output/part-r-00000
```

This completes the setup and testing of a single-node Hadoop cluster on your machine.