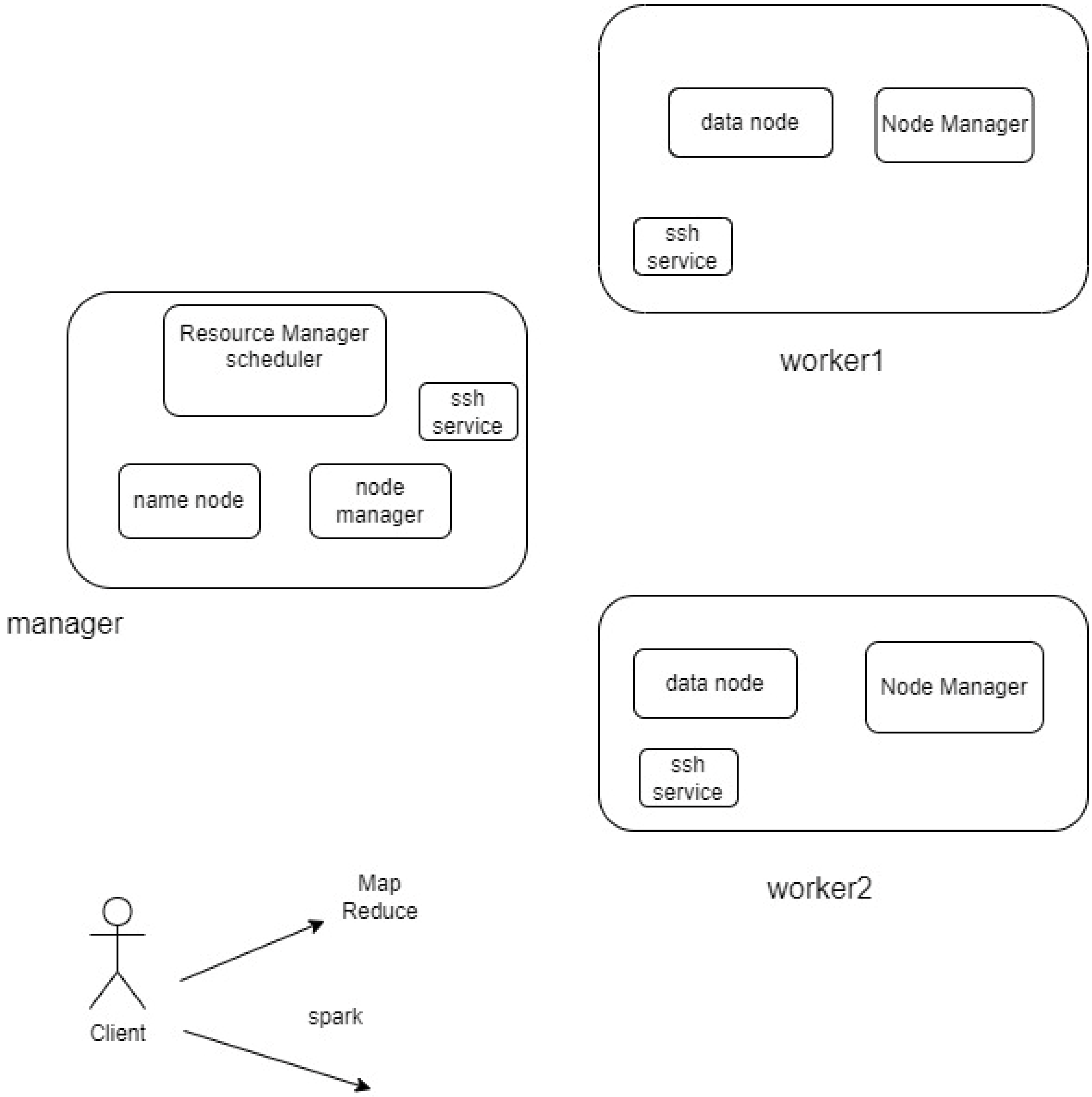
Setting Apache Hadoop framework in three ubuntu machines.

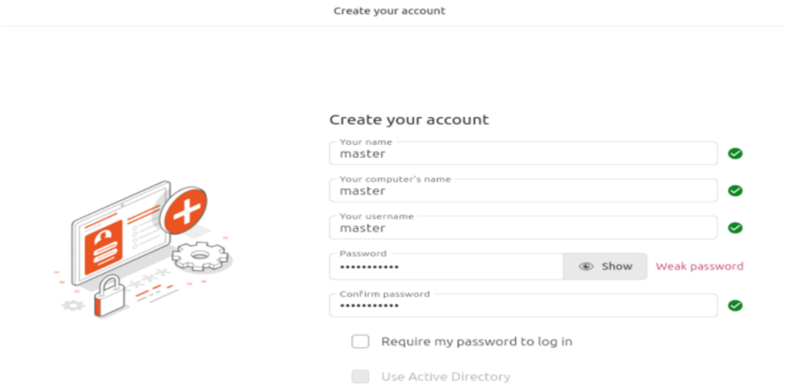
Hyper-V will be used in windows for at setup the following architecture.



1) Download ubuntu ISO files from the internet.
https://ubuntu.com/download/desktop

2) Set up 3 ubuntu machines.

- Resource manager machine: Give the Resource Manager machine name like manager and set your own password.



Create your account

Create your account

Your name: master

Your computer's name: master

Your username: master

Password: ••••••••••• Show Weak password

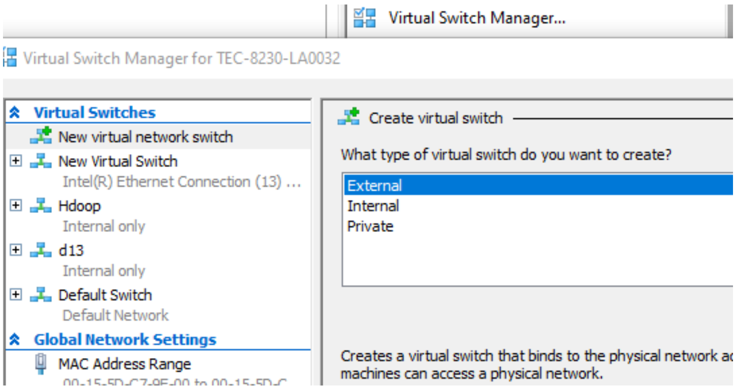
Confirm password: •••••••••••

☐ Require my password to log in

☐ Use Active Directory

- The first machine for Node manager and data node.
- The second machine for Node Manager and data node.

3) Create a virtual switch in Hyper-V which has internal type. Give it a name Hadoop.



Virtual Switch Manager for TEC-8230-LA0032

Virtual Switches
- New virtual network switch
- New Virtual Switch
  Intel(R) Ethernet Connection (13) ...
- Hdoop
  Internal only
- d13
  Internal only
- Default Switch
  Default Network

Global Network Settings
- MAC Address Range
  00-15-5D-C7-9E-00 to 00-15-5D-C

Create virtual switch

What type of virtual switch do you want to create?

External
Internal
Private

Creates a virtual switch that binds to the physical network ac machines can access a physical network.

4) Update the ubuntu machine. You can do that because you are connected to default switch in network setting. Sudo apt update sudo apt upgrade
5) Download Apache Hadoop in all machines.
   wget https://hadoop.apache.org/releases.html
6) Set up Apache Hadoop in all of the three machines. Repeat the following in all three machines.

- Extract folder with tar xzvf downloaded Hadoop file.
- Mv HadoopExtractedFolder Hadoop

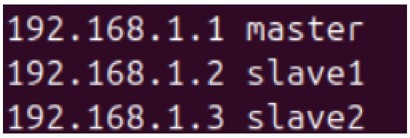
7) Install JDK in all of the three machines. sudo apt install openjdk-11-jdk
8) Set an IP address for network interface in every machine.
   - Sudo apt install net-tools.
   - Sudo ip addr add ip dev eth0. IP could be 192.168.1.1.
   - Ifconfig

9) Change the network setting of all machines to Hadoop. Ping between the machines to verify connectivity.

10) **Update /etc/hosts in all machines**

If you are not using DNS, you can manually map the NameNode's IP address to its hostname in the /etc/hosts file.

For example, on each DataNode,NameNode, add a line like this in /etc/hosts: Namenode IP address. command Nano /etc/hosts

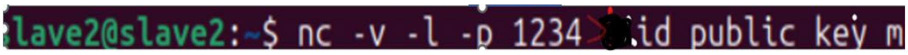


```
192.168.1.1 master
192.168.1.2 slave1
192.168.1.3 slave2
```

11) Set ssh connection with these keys between all machines. Create RSA keys in all machines.

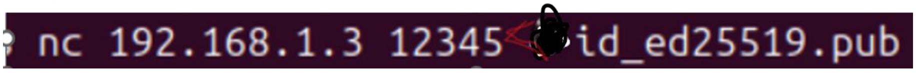
- Cd .ssh
- Ssh-keygen -t rsa b 1024
- Copy the generated public key to authorized-keys file so you could ssh to localhost.

If we want to copy public key from **manager** to **salve2**

- In worker 2 we run the command
  - 

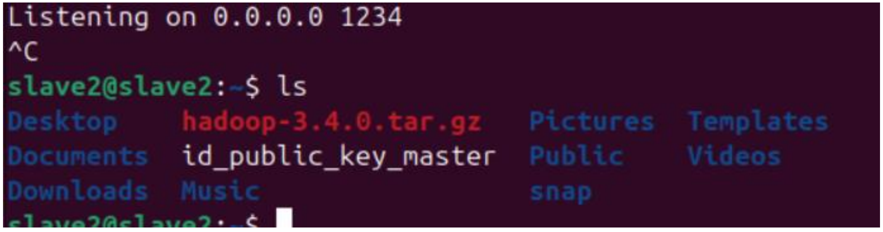
```
lave2@slave2:~$ nc -v -l -p 1234 >[redacted]id public key m
```

- In manager we run the command



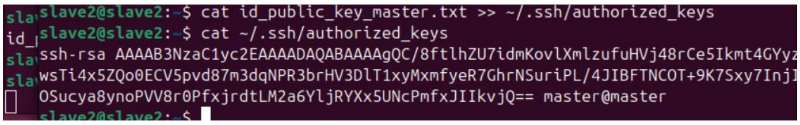
```
nc 192.168.1.3 12345 <[redacted]id_ed25519.pub
```

- In worker2 To verify the public key is copied.



```
Listening on 0.0.0.0 1234
^C
slave2@slave2:~$ ls
Desktop     hadoop-3.4.0.tar.gz    Pictures   Templates
Documents   id_public_key_master   Public     Videos
Downloads   Music                  snap
slave2@slave2:~$ 
```

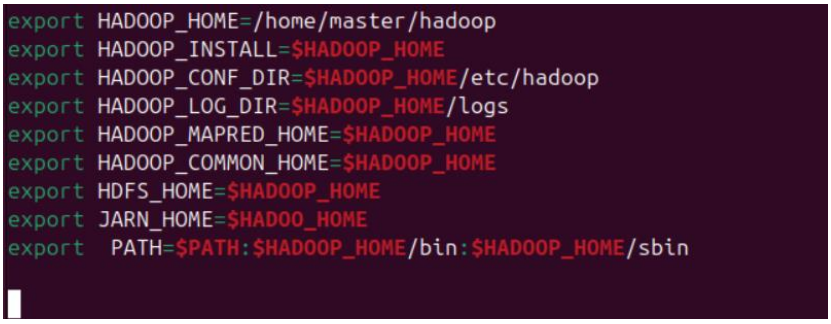
- Copy the key to the list of authorized keys that are trusted by Worker2



```
slave2@slave2:~$ cat id_public_key_master.txt >> ~/.ssh/authorized_keys
slave2@slave2:~$ cat ~/.ssh/authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAAAgQC/8ftlhZU7idmKovlXmlzufuHVj48rCe5Ikmt4GYyz
wsTi4x5ZQo0ECV5pvd87m3dqNPR3brHV3DlT1xyMxmfyeR7GhrNSuriPL/4JIBFTNCOT+9K7Sxy7InjI
OSucya8ynoPVV8r0PfxjrdtLM2a6YljRYXx5UNcPmfxJIIkvjQ== master@master
slave2@slave2:~$
```

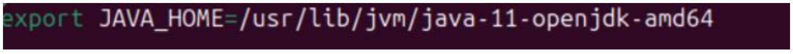
- Set up ssh service in all machines with the command Sudo apt Install openssh-server, and test ssh connection with running for example ssh manager@192.168.1.1

12) Set Environment Variables nano ~/.bashrc at the end of file



```
export HADOOP_HOME=/home/master/hadoop
export HADOOP_INSTALL=$HADOOP_HOME
export HADOOP_CONF_DIR=$HADOOP_HOME/etc/hadoop
export HADOOP_LOG_DIR=$HADOOP_HOME/logs
export HADOOP_MAPRED_HOME=$HADOOP_HOME
export HADOOP_COMMON_HOME=$HADOOP_HOME
export HDFS_HOME=$HADOOP_HOME
export JARN_HOME=$HADOO_HOME
export  PATH=$PATH:$HADOOP_HOME/bin:$HADOOP_HOME/sbin
```

13) Open Hadoop-env.sh and add the the following.



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

14) source ~/.bashrc

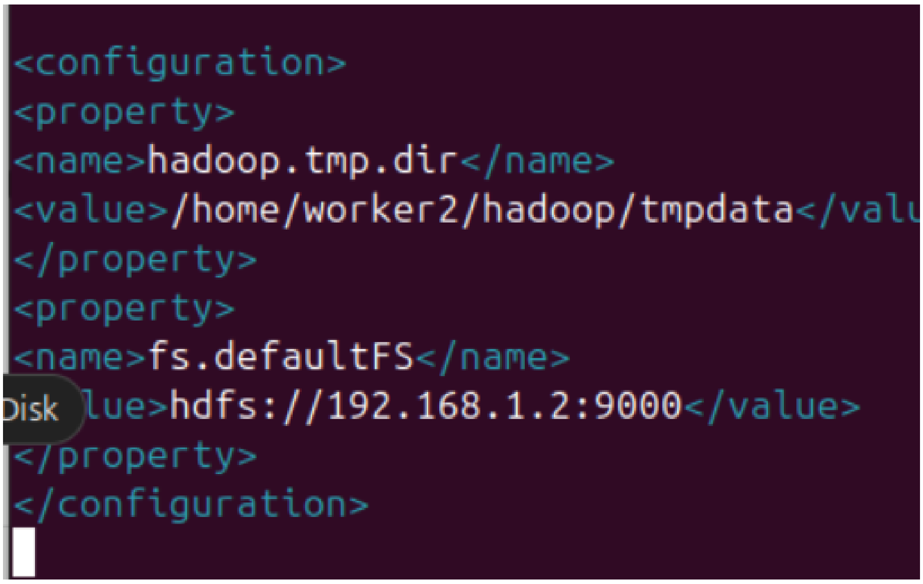
Hadoop Configuration steps.

Resource Manager and Name Node Configuration:

You need to edit a few XML files in the etc/hadoop directory.

1. **Core-site.xml**:

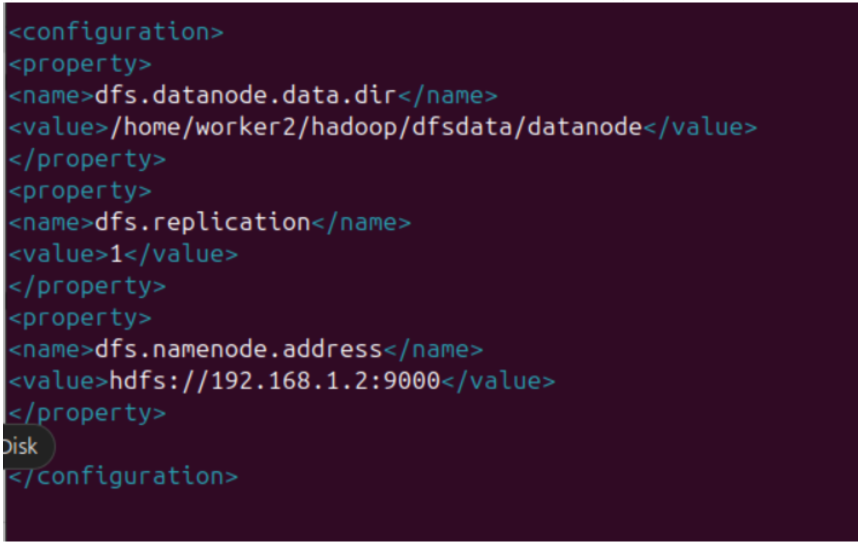
- mkdir tmpdata in hadoop directory.
- Set fs.defaultFS to point to the default file system URI. This property tells Hadooop where the Namenode is located, which essential for HDFS to function correctly.



```
<configuration>
<property>
<name>hadoop.tmp.dir</name>
<value>/home/worker2/hadoop/tmpdata</valu
</property>
<property>
<name>fs.defaultFS</name>
Disk lue>hdfs://192.168.1.2:9000</value>
</property>
</configuration>
```

2. **hdfs-site.xml**:

- Dfs.datanode.data.dir property specifies the directories on local filesystem where a datanode stores its HDFS data blocks. This property is configured normally in the data node.
- Dfs.namenode.name.dir property specifies he directories on the local filesystem where a Namenode store the namespace and transaction logs. This property is configured only on name node.
- Dfs.replication property specifies the defaults replication factor for HDFS files. It specifies how many copies of each fil are stored across the Hadoop cluster.

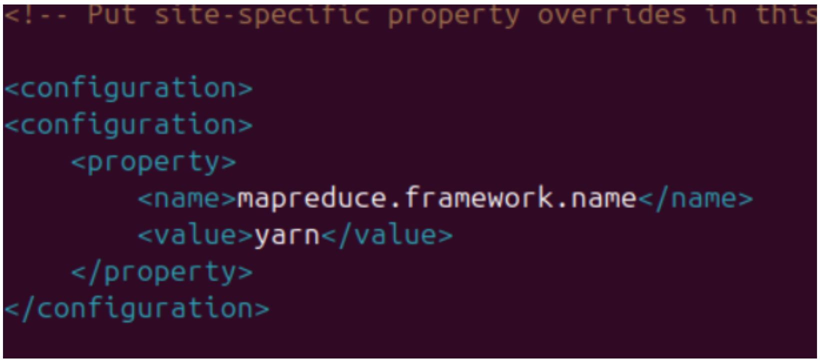


```
<configuration>
<property>
<name>dfs.datanode.data.dir</name>
<value>/home/worker2/hadoop/dfsdata/datanode</value>
</property>
<property>
<name>dfs.replication</name>
<value>1</value>
</property>
<property>
<name>dfs.namenode.address</name>
<value>hdfs://192.168.1.2:9000</value>
</property>
Disk
</configuration>
```

Create folder dfsnode and datanode inside hdfs inside Hadoop folder.

3. Edit the mapred-site.xml

- Mapresuce.framwork.name property specifies the framework that map reduce **jobs** will use to execute.

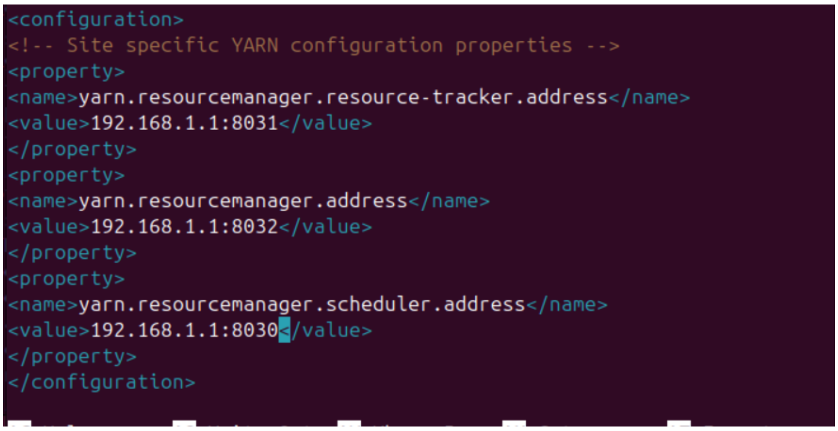


```
<!-- Put site-specific property overrides in this

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

4. Edit yarn-site.xml

- Yarn.resourcemanager.address property to specifies the hostname and port on which resource manager listen to client requests.
- Yarn.resourcemanager.address property specifies the hostname and port on which the ResourceManager listen for heartbeats from the nodemanagers.
- Yarn.resourcemanager.scheduler.address property specifies the hostname and port on which the resource manager listen for requests from application manager to allocate resource.



```
<configuration>
<!-- Site specific YARN configuration properties -->
<property>
<name>yarn.resourcemanager.resource-tracker.address</name>
<value>192.168.1.1:8031</value>
</property>
<property>
<name>yarn.resourcemanager.address</name>
<value>192.168.1.1:8032</value>
</property>
<property>
<name>yarn.resourcemanager.scheduler.address</name>
<value>192.168.1.1:8030</value>
</property>
</configuration>
```

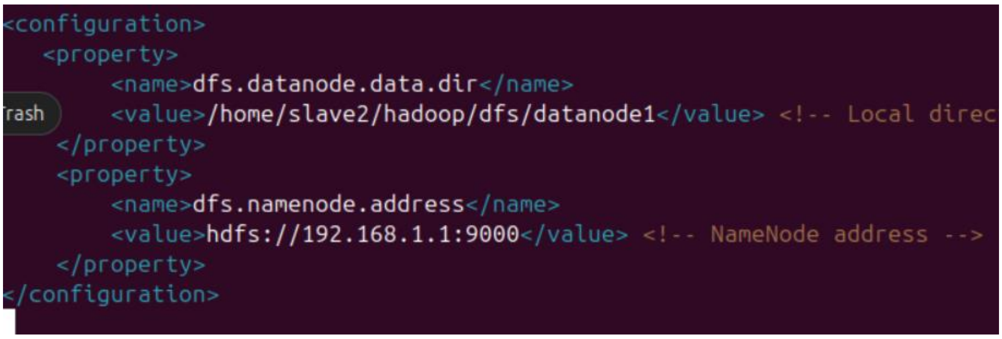
5. **Update Hadoop Configuration**:
 Ensure that the `JAVA_HOME` variable is correctly set in the Hadoop configuration file. Open the `hadoop-env.sh` file, which is usually located in the `etc/hadoop` directory of your Hadoop installation, and set the `JAVA_HOME` variable:

```
export JAVA_HOME=/path/to/java
```

Data Node configuration

In data node you should do the following configuration in

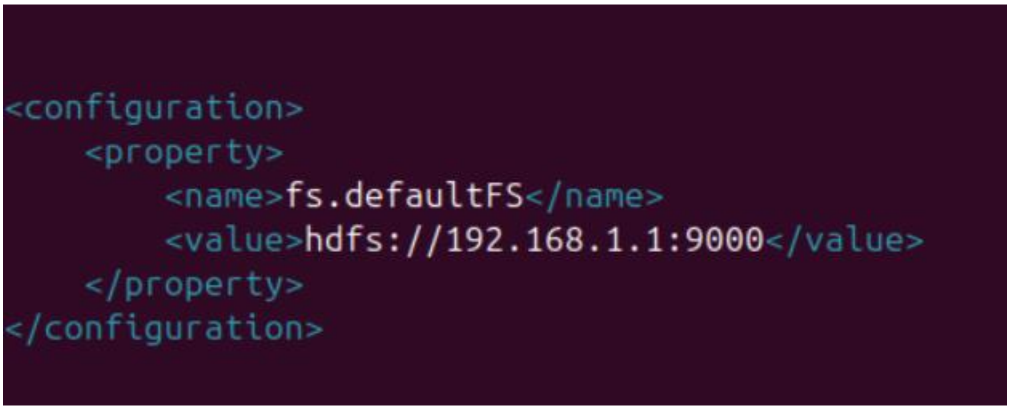
1) hdfs-site.xml



```
<configuration>
   <property>
        <name>dfs.datanode.data.dir</name>
        <value>/home/slave2/hadoop/dfs/datanode1</value> <!-- Local direc
    </property>
    <property>
        <name>dfs.namenode.address</name>
        <value>hdfs://192.168.1.1:9000</value> <!-- NameNode address -->
    </property>
</configuration>
```

You should create the folder dfs and datanode1 and give permission for dfs to access it.

Core-site.xml

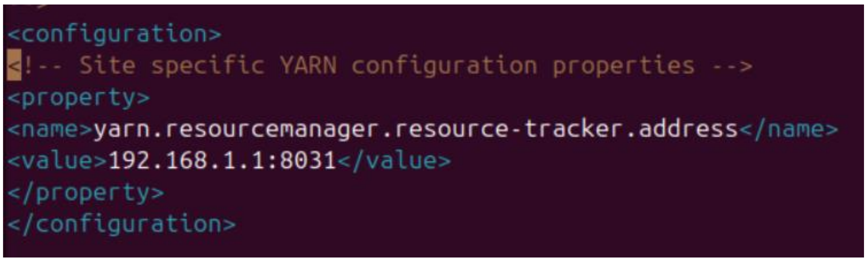


```
<configuration>
    <property>
        <name>fs.defaultFS</name>
        <value>hdfs://192.168.1.1:9000</value>
    </property>
</configuration>
```

Worker Node Configuration

Configuration of tracking address and ports must consistent between node manager and resource manager

Example yarn-site.xml for NodeManager:



```
<configuration>
<!-- Site specific YARN configuration properties -->
<property>
<name>yarn.resourcemanager.resource-tracker.address</name>
<value>192.168.1.1:8031</value>
</property>
</configuration>
```

Note

When you run spark in data node and you are trying to read and write to an existing folder in hdfs you need to have permission form the user in datanode for that.

```
hdfs dfs -ls /path/to/directory
hdfs dfs -chmod 777 /path/to/directory
```

How to check the system after configuration er finished

In Resource manager node

Namenode Web UI (HDFS Monitoring)

Open the browser and write an address IP:9870.
It will show you the status of HDFS.

Resource Manager Web UI (YARN Monitoring)

Open the browser and write an address IP:8088

This will help you to monitor and manage the YARN resource manager scheduler, including active and completed jobs.

In worker node
Node Manager web UI
Open the browser and write the address IP:8042