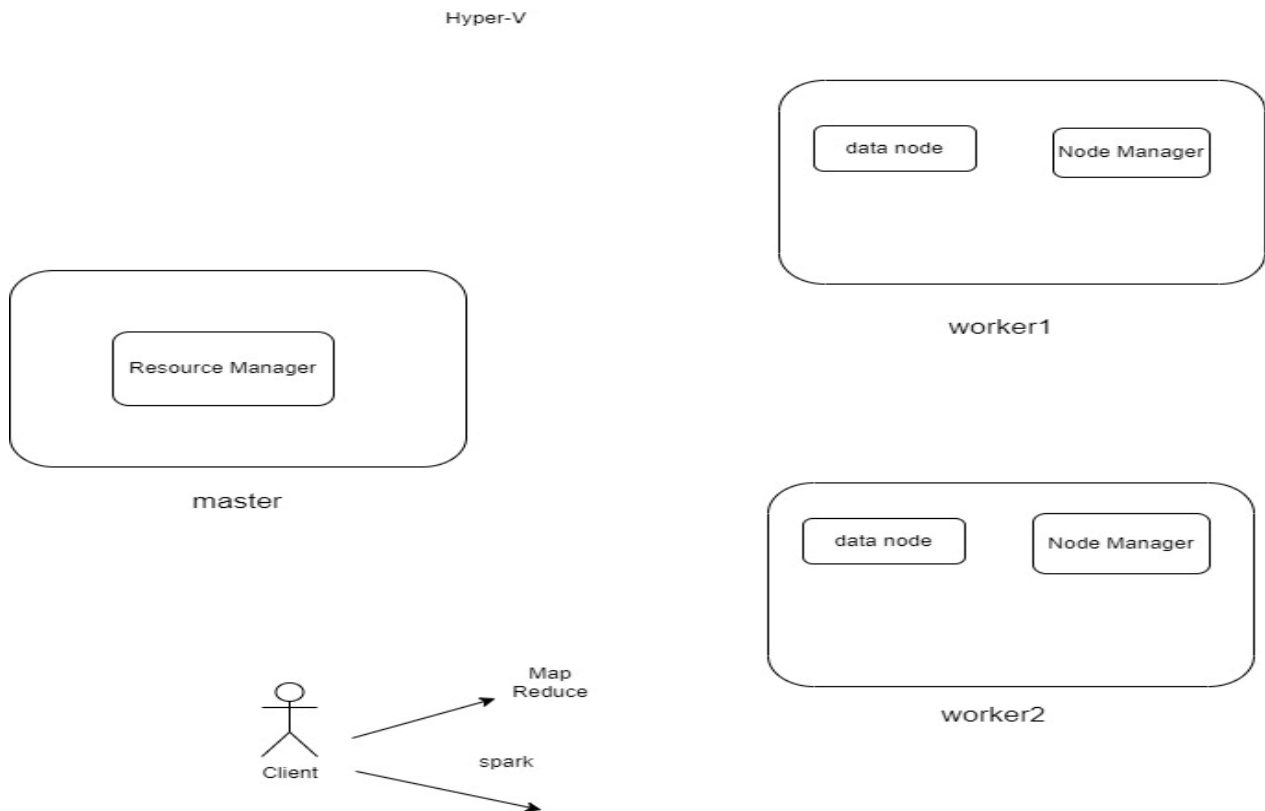


Setting Apache Hadoop in three ubuntu machines.

We will use Hyper-V in windows for at setup the following architecture.




1) Download 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 Master and set your own password.

Create your account



Create your account

Your name
master ✓

Your computer's name
master ✓

Your username
master ✓

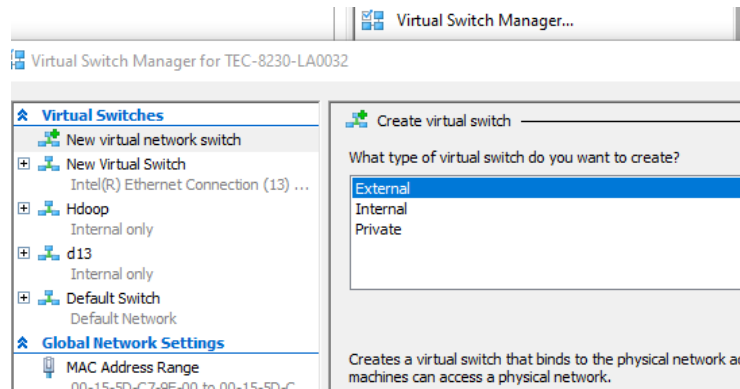
Password
***** Show Weak password

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.



- 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 **master** to **slave2**

- In slave 2 we run the command

```
slave2@slave2:~$ nc -v -l -p 1234 >> id_public_key_m
```

- In master we run the command

```
nc 192.168.1.3 12345 << id_ed25519.pub
```

- In slave2 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
```

- Copy the key to the list of authorized keys that are trusted by Slave2

```
slave2@slave2:~$ cat id_public_key_master.txt >> ~/.ssh/authorized_keys
id_
slave2@slave2:~$ cat ~/.ssh/authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQgQC/8ftlhZU7idmKovlXmIzufuHVj48rCe5Ikmt4GYyz
wsTi4xSZQo0ECVSpvd87m3dqNPR3brHV3DLT1xyMxmfyR7GhrNSurIPL/4JIBFTNCOT+9K7Sxy7Inj1
OSucya8YnoPVV8r0PfxjrdtLM2a6YljRYXx5UNCpmfxJIikvjQ== master@master
```

- Set up ssh service in all machines with the command Sudo apt Install openssh-server, and test ssh connection with running for example ssh master@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=$HADOOP_HOME
export PATH=$PATH:$HADOOP_HOME/bin:$HADOOP_HOME/sbin
```

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

13) 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 Hadoop where the Namenode is located, which essential for HDFS to function correctly.

```

<configuration>
<property>
<name>hadoop.tmp.dir</name>
<value>/home/worker2/hadoop/tmpdata</value>
</property>
<property>
<name>fs.defaultFS</name>
<value>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 the 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 file 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>
</configuration>

```

Create folder dfsnode and datanode inside hdfs inside Hadoop folder.

3. Edit the mapred-site.xml

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

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

<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 master 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>
```

Slave 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>
```

How to start the Hadoop cluster framework.

In terminal in Resource manager machine.

Hadoop-demon.sh start namenode to start.

The command initiates the NameNode process, which runs on the master node of the Hadoop cluster. This process is responsible for managing the file system namespace and regulating access to files by clients.

yarn-daemon start resourcemanager.

start the ResourceManager daemon in a Hadoop YARN (Yet Another Resource Negotiator) cluster

In terminal in slave machine use

hadoop-daemon.sh start datanode.

It used to start the DataNode daemon in a Hadoop cluster.

Start node manager in slave machine.

yarn-daemon.sh start nodeManager.

Note

To stop all services in the node you use stop-all.sh.

If it does not work, use kill -9 process number from jps command.

How to use Hadoop Distributed file system to save file in specific folder

Create a folder where you want to save a folder

```
Hadoop fs -mkdir -p /data
```

How to save a file to this folder?

```
Hdfs dfs -put rockyou.txt /data
```

How to use hadoop to remove the folder?

```
Hdfs dfs -rm -r /data
```

How to list all folders?

```
Hadoop fs -ls /
```

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 from 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

Data processing med python på spark

Start an ubuntu machine in hyper V

Update ubuntu machine

Sudo apt update

Sudo apt upgrade

Add the current user to the list of sudo user to avoid providing the password every time you want to update it

```
sudo usermod -aG sudo youruserAccount name
```

To set pyspark you have two options

option1

Using pip in a virtual environment: This is the recommended way to install PySpark if it's not available via apt.

Install package installer for python pip

Sudo install python3-pip

Install pyspark package using pip install pyspark

Option 2

Manual Installation: Download and set up Apache Spark directly if needed.

1. **Install Java** (Spark requires Java):

```
sudo apt install openjdk-11-jdk
```

2. **Download Apache Spark:**

- Go to the [Apache Spark downloads page](#) and choose a version (e.g., Spark 3.5.0).
- Copy the download link for the pre-built package for Hadoop.

3. **Use wget to download it:**

4. **Extract the downloaded file:**

```
tar -xzf spark-3.5.0-bin-hadoop3.tgz
```

5. **Move it to a more appropriate directory** (optional):

```
sudo mv spark-3.5.0-bin-hadoop3 /opt/spark
```

6. **Set Environment Variables:** Add the following lines to your ~/.bashrc or ~/.profile file:

```
export SPARK_HOME=/opt/spark
export PATH=$PATH:$SPARK_HOME/bin
```

7. **Load the environment variables:**

```
source ~/.bashrc
```

8. **Verify the Installation:** You can verify the installation by running:

```
spark-shell
```

9. Run the code with spark-submit yourcode.py

Here's a typical command structure for using spark-submit:

```
spark-submit [options] your_script.py [script arguments]
```

Common Options

1. **--master**: Specifies the cluster manager to connect to (e.g., local, yarn, mesos).
 - Example: `--master local[4]` runs the job locally with 4 threads.
2. **--deploy-mode**: Indicates how to deploy the driver program (e.g., client or cluster).
 - Example: `--deploy-mode cluster` runs the driver on the cluster.
3. **--executor-memory**: Specifies the amount of memory to use per executor process.
 - Example: `--executor-memory 2G` allocates 2 GB of memory for each executor.
4. **--num-executors**: Defines the number of executors to launch.
 - Example: `--num-executors 10` starts 10 executor instances.
5. **--py-files**: Distributes .zip or .py files to the worker nodes.
 - Example: `--py-files my_package.zip` includes additional Python packages.
- 6.
7. The problem appears when try to run `spark_submit` from data node against files that exist in hdfs
- 8.