How to Setup / Install an Apache Spark 3.1.1 Cluster on Ubuntu

Published on April 6, 2022

Dr. Virendra Kumar Shrivastava
Professor (Computer Science & Engineering) || Researcher || Blogger || Mentor || Alliance...
49 articles + Follow

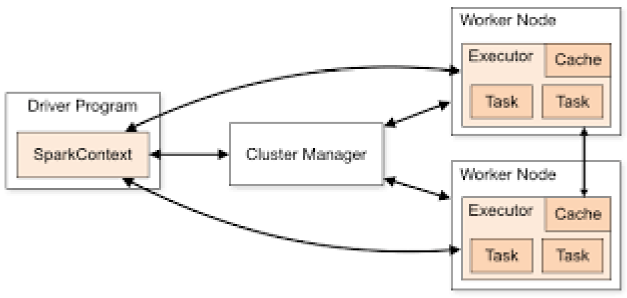
Hey guys,

In this article, I will explain how to setup **Apache Spark** 3.1.1 on a multi-node cluster which includes installing spark master and workers. I will provide step by step instructions to setup spark on Ubuntu 16.04. So, if you are you are looking to make your hand dirty on Apache Spark cluster, this article can be a steppingstone for you.

What is Apache Spark?

It is an open-source and distributed processing system used for big data workloads. Spark is a fast, general engine and powerful engine for big data processing. Apache Spark follows a master/worker architecture (two main daemons) and a cluster manager.

- Master Daemon (Master/Driver Process)
- Worker Daemon (Slave Process)
- Cluster Manager



An Apache Spark cluster used to have a Master and one or more than one Workers /Slaves.

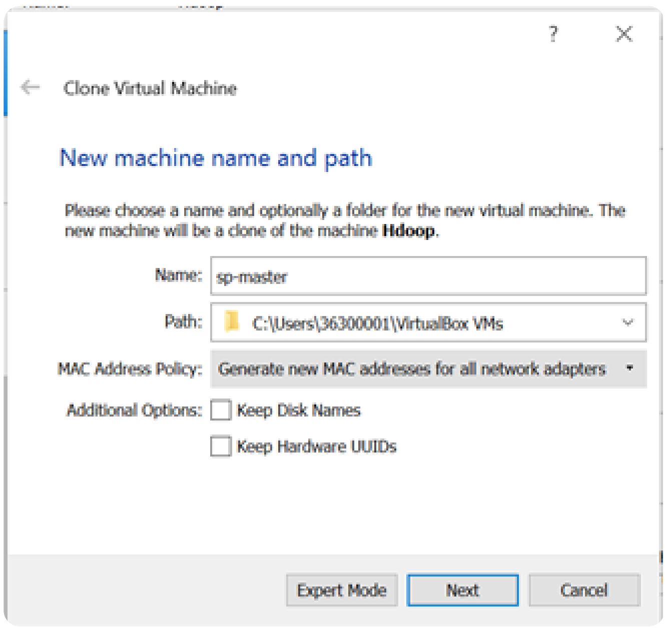
Pre-requisite:

Ubuntu 16.04 or higher installed on a virtual machine.

Steps for installation of Apache Spark 3.1.1 Cluster on Hadoop 3.2

Step 1.

has been earlier created. Select option "Generate new MAC addresses for all network adapters" in MAC Address Policy. And also choose the option "Full Clone" in clone type.



Clone Virtual Machine

New machine name and path

Please choose a name and optionally a folder for the new virtual machine. The new machine will be a clone of the machine **Hdoop**.

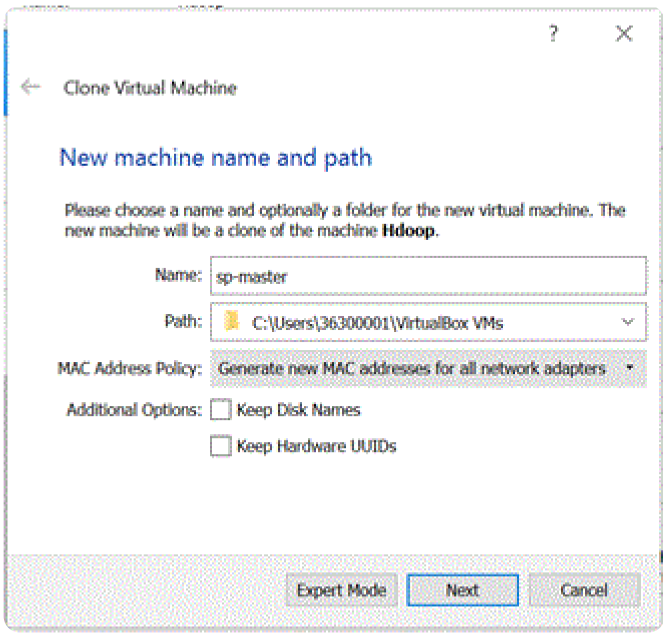
Name: sp-master

Path: C:\Users\36300001\VirtualBox VMs

MAC Address Policy: Generate new MAC addresses for all network adapters

Additional Options: [ ] Keep Disk Names

[ ] Keep Hardware UUIDs



Clone Virtual Machine

New machine name and path

Please choose a name and optionally a folder for the new virtual machine. The new machine will be a clone of the machine **Hdoop**.

Name: sp-master

Path: C:\Users\36300001\VirtualBox VMs

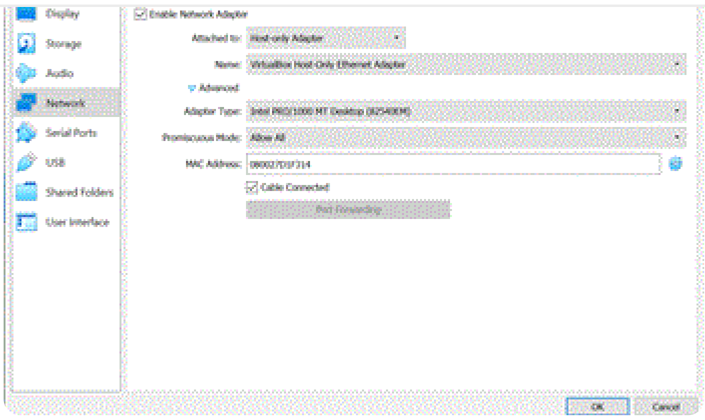
MAC Address Policy: Generate new MAC addresses for all network adapters

Additional Options: [ ] Keep Disk Names

[ ] Keep Hardware UUIDs

Step 2.

Go to settings option of virtual machines and make following network configuration on Adapter 2.



☑ Enable Network Adapter
Attached to: Host-only Adapter
Name: VirtualBox Host-Only Ethernet Adapter
Advanced
Adapter Type: Intel PRO/1000 MT Desktop (82540EM)
Promiscuous Mode: Allow All
MAC Address: 080027D5F314
☑ Cable Connected

Step 3.

You need to set the hostname of each virtual machine. Open the /etc/hostname file and type the name of the machine in it and save. Run the following command on each virtual machine:

$ **sudo nano /etc/hostname**

Step 4.

To figure out IP address of the virtual machines run the following command:

$ **ip addr**

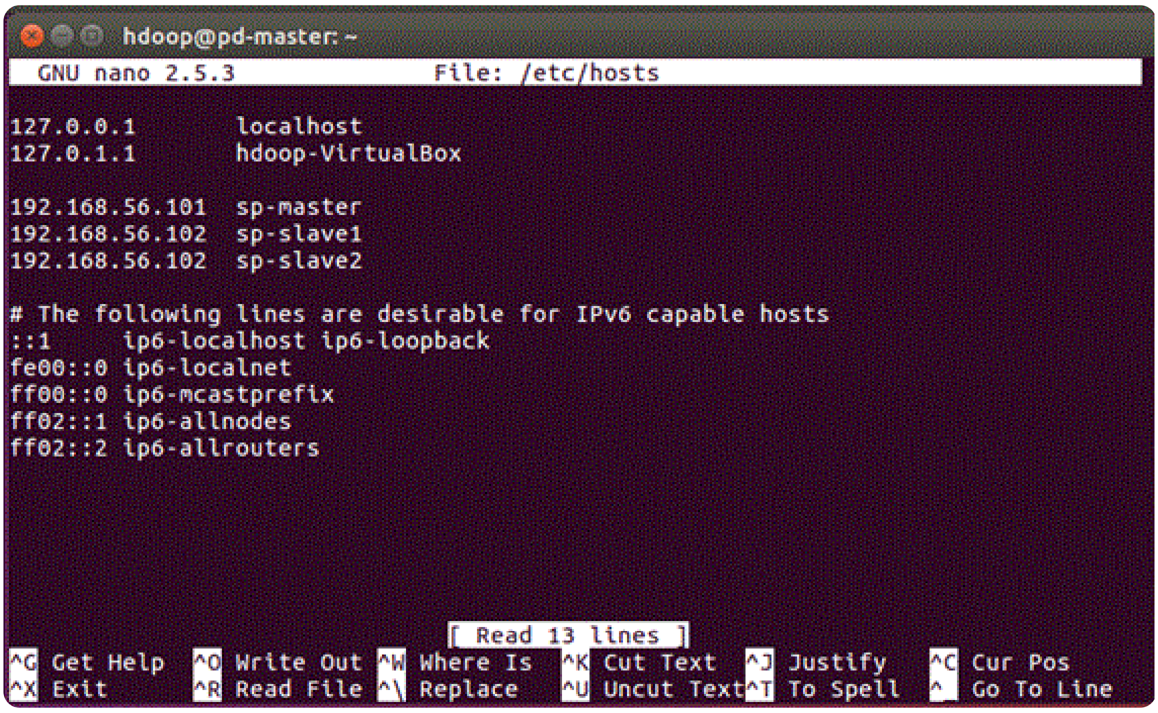
I run the above-mentioned command on master and workers (slaves). On my system, I found following IP addresses:

sp-master	192.168.56.101
sp-slave1	192.168.56.102
sp-slave2	192.168.56.103

Step 5.

In this step, I edit **hosts** file and added IP address and hostname information, saved it, and reboot the machine. Run the following command on all the machines.

$ **sudo nano /etc/hosts**



```
hdoop@pd-master: ~
  GNU nano 2.5.3                 File: /etc/hosts

127.0.0.1       localhost
127.0.1.1       hdoop-VirtualBox

192.168.56.101  sp-master
192.168.56.102  sp-slave1
192.168.56.102  sp-slave2

# The following lines are desirable for IPv6 capable hosts
::1     ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters

                               [ Read 13 lines ]
^G Get Help  ^O Write Out ^W Where Is  ^K Cut Text  ^J Justify   ^C Cur Pos
^X Exit      ^R Read File ^\ Replace   ^U Uncut Text^T To Spell  ^_ Go To Line
```

$ **sudo reboot**

Step 6.

Run the following commands on all the Machines (master and workers / slaves).

$ **sudo apt-get update**

#to check version of java, run the following command.

```
$ java -version
```

Step 7. (On Master and workers)

Install Scala on the all the machines (master and the worker / slaves). Run the following command:

```
$ sudo apt-get install scala
```

To check the version of Scala, run the following command:

```
$ scala -version
```

Step 8. (On Master only)

Now configure Open SSH server-client on master. To configure Open SSH server-client, run the following command:

```
$ sudo apt-get install openssh-server openssh-client
```

Next step is to generate key pairs. For this purpose, run the following command:

```
$ ssh-keygen -t rsa -P ""
```

Run the following command to authorize the key:

```
$ cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys
```

following commands:

```
$ ssh-copy-id user@192.168.56.101

$ ssh-copy-id user@192.168.56.102

$ ssh-copy-id user@192.168.56.103
```

Note: user name and IP will be different of your machines. So, use accordingly.

Now it's time to check if everything installed properly. Run the following command on master to connect to the slaves / workers:

```
$ ssh 192.168.56.102

$ ssh 192.168.56.103
```

You can exit from slave machine by type the command:

```
$ exit
```

Step 9. (On all the Virtual Machines – Master and workers)

Download the stable version of Apache Spark. I will install spark-3.1.1 with Hadoop-3.2. To download spark-3.1.1 with Hadoop-3.2, run the following command:

Now run the following command to untar the spark tar file:

$ **tar xvf spark-3.1.1-bin-hadoop3.2.tgz**

Run the following command to move the spark files to the spark directory (*/usr/local/bin*):

$ **sudo mv spark-3.1.1-bin-hadoop3.2 /usr/local/spark**

To set up the environment for Apache Spark, we need to edit the *.bashrc* file. Run the following command to edit .bashrc file:

$ **sudo nano ~/.bashrc**

Add the following line to the file and save.

export PATH = $PATH:/usr/local/spark/bin

The above line sets the location (Path) where the spark software file is located to the PATH variable.

Run the following command to make effective changes in the .bashrc file:

$ **source ~/.bashrc**

Step 10. (On Master only)

*env.sh.*template file as a spark-env.sh

$ **cd /usr/local/spark/conf**

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

Now, to edit the *spark-env.sh* configuration file, run the following command:

$ **sudo nano *spark-env.sh***

Add the following parameters (line of code) at the end of file, save and exit.

export SPARK_MASTER_HOST='<Master-IP>'export JAVA_HOME=<Path_of_JAVA_installation>

Note: In my system, MASTER IP is 192.168.5.101 and JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64. You need to use these parameters as per your system.

Add Workers or Slaves

Now, to edit the configuration file *usr/local/spark/conf/slaves, run the following command on master:*

$ **sudo nano /usr/local/spark/conf/slaves**

And add the master and workers/slaves name (given below) in the above-mentioned file, save and exit.

sh-slave1

sh-slave2

Step 11.

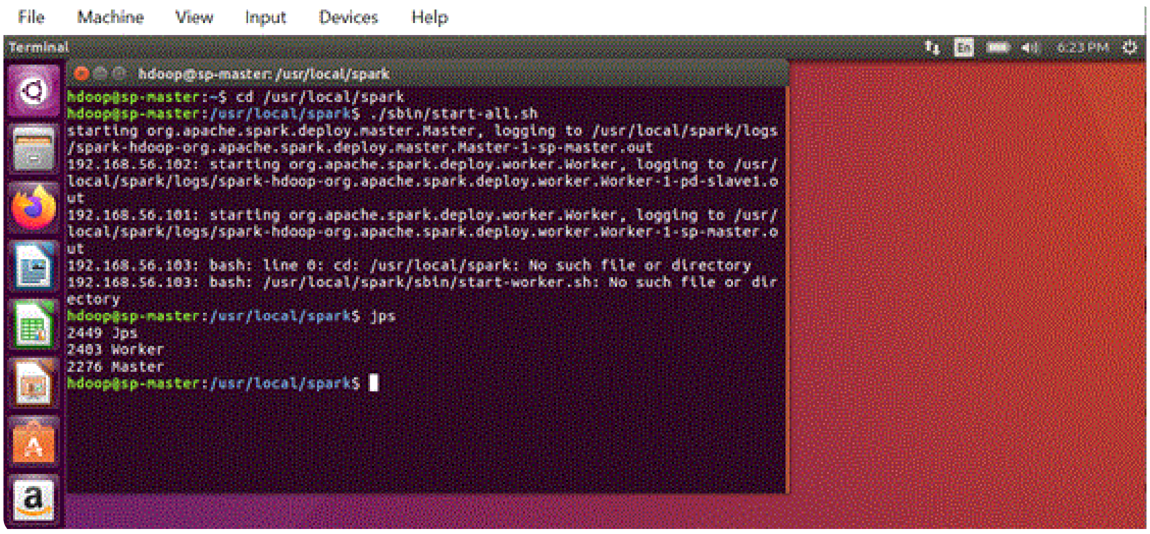
Now, our Apache Spark Cluster is ready. To start the Apache Spark cluster, run the following command on master:

$ **cd /usr/local/spark**

$ **./sbin/start-all.sh**

Now, to check the services started by spark, run the following command:

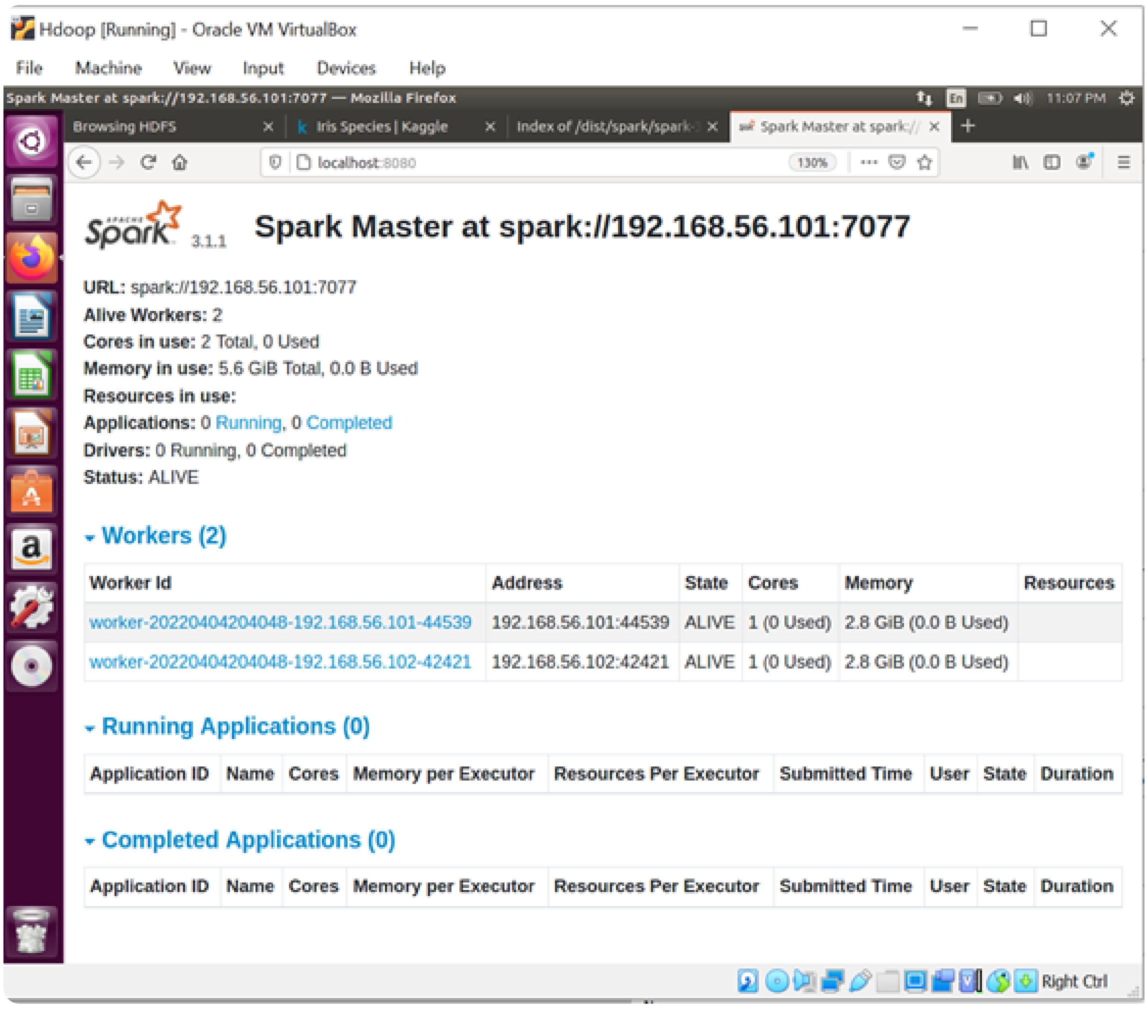
$ **jps**



```
hdoop@sp-master:~$ cd /usr/local/spark
hdoop@sp-master:/usr/local/spark$ ./sbin/start-all.sh
starting org.apache.spark.deploy.master.Master, logging to /usr/local/spark/logs
/spark-hdoop-org.apache.spark.deploy.master.Master-1-sp-master.out
192.168.56.102: starting org.apache.spark.deploy.worker.Worker, logging to /usr/
local/spark/logs/spark-hdoop-org.apache.spark.deploy.worker.Worker-1-pd-slave1.o
ut
192.168.56.101: starting org.apache.spark.deploy.worker.Worker, logging to /usr/
local/spark/logs/spark-hdoop-org.apache.spark.deploy.worker.Worker-1-sp-master.o
ut
192.168.56.103: bash: line 0: cd: /usr/local/spark: No such file or directory
192.168.56.103: bash: /usr/local/spark/sbin/start-worker.sh: No such file or dir
ectory
hdoop@sp-master:/usr/local/spark$ jps
2449 Jps
2403 Worker
2276 Master
hdoop@sp-master:/usr/local/spark$ 
```

Step 12.

In case of my system, my master IP is 192.168.56.101.



Hdoop [Running] - Oracle VM VirtualBox

Spark Master at spark://192.168.56.101:7077 — Mozilla Firefox

localhost:8080

3.1.1 Spark Master at spark://192.168.56.101:7077

URL: spark://192.168.56.101:7077
Alive Workers: 2
Cores in use: 2 Total, 0 Used
Memory in use: 5.6 GiB Total, 0.0 B Used
Resources in use:
Applications: 0 Running, 0 Completed
Drivers: 0 Running, 0 Completed
Status: ALIVE

Workers (2)

Worker Id	Address	State	Cores	Memory	Resources
worker-20220404204048-192.168.56.101-44539	192.168.56.101:44539	ALIVE	1 (0 Used)	2.8 GiB (0.0 B Used)	
worker-20220404204048-192.168.56.102-42421	192.168.56.102:42421	ALIVE	1 (0 Used)	2.8 GiB (0.0 B Used)	

Running Applications (0)

Application ID	Name	Cores	Memory per Executor	Resources Per Executor	Submitted Time	User	State	Duration

Completed Applications (0)

Application ID	Name	Cores	Memory per Executor	Resources Per Executor	Submitted Time	User	State	Duration

you can see in the above snippet; we have two alive Workers are running.

Step 13.

To stop services of Apache Spark cluster, run the following command:

$ **./sbin/stop-all.sh**