

# Introduction

Every major industry is implementing [Apache Hadoop](#) as the standard framework for processing and storing big data. Hadoop is designed to be deployed across a network of hundreds or even thousands of [dedicated servers](#). All these machines work together to deal with the massive volume and variety of incoming datasets.

Deploying Hadoop services on a single node is a great way to get yourself acquainted with basic Hadoop commands and concepts.

**This easy-to-follow guide helps you install Hadoop on Ubuntu 18.04 or Ubuntu 20.04.**

## Prerequisites

- Access to a terminal window/command line
- **Sudo** or **root** privileges on local /remote machines

# Install OpenJDK on Ubuntu

The [Hadoop framework](#) is written in Java, and its services require a compatible Java Runtime Environment (JRE) and Java Development Kit (JDK). Use the following command to update your system before initiating a new installation:

```
sudo apt update
```

At the moment, **Apache Hadoop 3.x fully supports Java 8**. The OpenJDK 8 package in Ubuntu contains both the runtime environment and development kit.

Type the following command in your terminal to install OpenJDK 8:

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

The OpenJDK or Oracle Java version can affect how elements of a Hadoop ecosystem interact. To install a specific Java version, check out our detailed guide on [how to install Java on Ubuntu](#).

Once the installation process is complete, [verify the current Java version](#):

```
java -version; javac -version
```

The output informs you which Java edition is in use.

```
pnap@pnap-VirtualBox:~$ java -version;javac -version
openjdk version "1.8.0_252"
OpenJDK Runtime Environment (build 1.8.0_252-8u252-b09-1~18.04-b09)
OpenJDK 64-Bit Server VM (build 25.252-b09, mixed mode)
javac 1.8.0_252
```

## Set Up a Non-Root User for Hadoop Environment

It is advisable to create a non-root user, specifically for the Hadoop environment. A distinct user improves security and helps you manage your cluster more efficiently. To ensure the smooth functioning of Hadoop services, the user should have the ability to establish a [passwordless SSH connection](#) with the localhost.

### Install OpenSSH on Ubuntu

Install the OpenSSH server and client using the following command:

```
sudo apt install openssh-server openssh-client -y
sudo apt-get install vsftpd
```

In the example below, the output confirms that the latest version is already installed.

```
pnap@pnap-VirtualBox:~$ sudo apt-get install openssh-server openssh-client
Reading package lists... Done
Building dependency tree
Reading state information... Done
openssh-client is already the newest version (1:7.6p1-4ubuntu0.3).
openssh-server is already the newest version (1:7.6p1-4ubuntu0.3).
0 upgraded, 0 newly installed, 0 to remove and 54 not upgraded.
```

If you have installed OpenSSH for the first time, use this opportunity to implement these vital [SSH security recommendations](#).

## Enable Passwordless SSH for Hadoop User

[Generate an SSH key pair](#) and define the location it is to be stored in:

```
ssh-keygen -t rsa -P "" -f ~/.ssh/id_rsa
```

The system proceeds to generate and save the SSH key pair.

```
hadoop@pnap-VirtualBox:~$ ssh-keygen -t rsa -P "" -f ~/.ssh/id_rsa
Generating public/private rsa key pair.
Your identification has been saved in /home/hadoop/.ssh/id_rsa.
Your public key has been saved in /home/hadoop/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:9GEXT7rDLjtcp5dT4KE0LevsYzloAKxir/E0zPjP58Y hadoop@pnap-VirtualBox
The key's randomart image is:
+---[RSA 2048]---+
|                 . . |
|                 =   |
|      . . . O O . . |
|      O. O ++. +   |
|      . S ..+* O   |
| O+.   . . +.O .   |
| .oo= . O.=.+ O   |
|      =.O E =OO +   |
|      ..O.O+...+.+ . |
+---[SHA256]-----+
hadoop@pnap-VirtualBox:~$ cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys
hadoop@pnap-VirtualBox:~$ chmod 0600 ~/.ssh/authorized_keys
```

Use the `cat` command to store the public key as **authorized\_keys** in the `ssh` directory:

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

Set the permissions for your user with the `chmod` command:

```
chmod 0600 ~/.ssh/authorized_keys
```

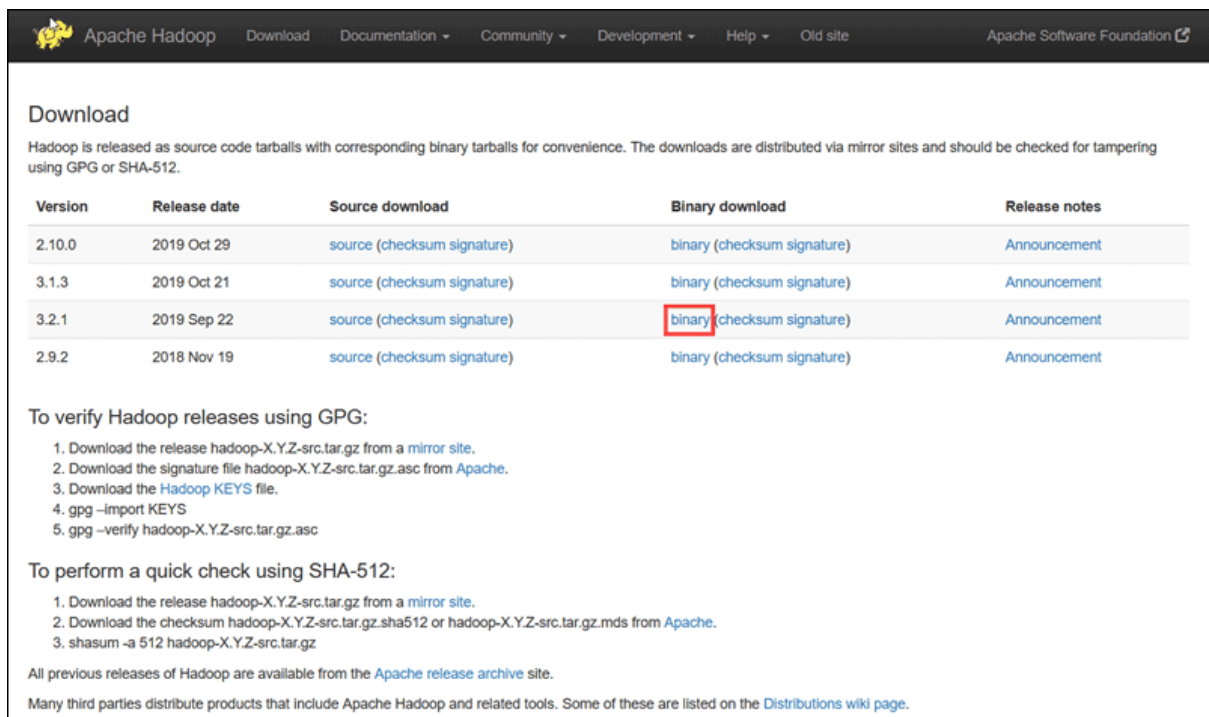
The new user is now able to SSH without needing to enter a password every time. Verify everything is set up correctly by using the **hadoop** user to SSH to localhost:

ssh localhost

After an initial prompt, the Hadoop user is now able to establish an SSH connection to the localhost seamlessly.

# Download and Install Hadoop on Ubuntu

Visit the [official Apache Hadoop project page](#), and select the version of Hadoop you want to implement.

The screenshot shows the Apache Hadoop website's download page. At the top is a navigation bar with links for Download, Documentation, Community, Development, Help, and Old site, along with the Apache Software Foundation logo. The main heading is "Download". Below it, a paragraph states that Hadoop is released as source code tarballs with corresponding binary tarballs for convenience, distributed via mirror sites and checked for tampering using GPG or SHA-512. A table lists the available versions. The table has five columns: Version, Release date, Source download, Binary download, and Release notes. The rows are for versions 2.10.0, 3.1.3, 3.2.1, and 2.9.2. In the row for version 3.2.1, the "binary (checksum signature)" link is highlighted with a red box. Below the table, there are instructions on how to verify releases using GPG and how to perform a quick check using SHA-512. At the bottom, it mentions that all previous releases are available from the Apache release archive site and that many third parties distribute products that include Apache Hadoop and related tools.

The steps outlined in this tutorial use the Binary download for **Hadoop Version 3.3.4**.

Select your preferred option, and you are presented with a mirror link that allows you to download the **Hadoop tar package**.



COMMUNITY-LED DEVELOPMENT "THE APACHE WAY"

[Projects](#) [People](#) [Community](#) [License](#) [Sponsors](#)



We suggest the following mirror site for your download:

<https://downloads.apache.org/hadoop/common/hadoop-3.2.1/hadoop-3.2.1.tar.gz>

Other mirror sites are suggested below.

It is essential that you verify the integrity of the downloaded file using the PGP signature (`.asc` file) or a hash (`.md5` or `.sha*` file).

Please only use the backup mirrors to download KEYS, PGP signatures and hashes (SHA\* etc) -- or if no other mirrors are working.

## HTTP

<https://downloads.apache.org/hadoop/common/hadoop-3.2.1/hadoop-3.2.1.tar.gz>

## BACKUP SITES

Please only use the backup mirrors to download KEYS, PGP signatures and hashes (SHA\* etc) -- or if no other mirrors are working.

<https://downloads.apache.org/hadoop/common/hadoop-3.2.1/hadoop-3.2.1.tar.gz>

The full listing of mirror sites is also available.

**Note:** It is sound practice to verify Hadoop downloads originating from mirror sites. The instructions for using GPG or SHA-512 for verification are provided on the official download page.

Use the provided mirror link and download the Hadoop package with the `wget` command:

```
wget https://downloads.apache.org/hadoop/common/hadoop-3.3.4/hadoop-3.3.4.tar.gz
```

```
hadoop@pnap-VirtualBox:~$ wget https://downloads.apache.org/hadoop/common/hadoop-3.2.1/hadoop-3.2.1.tar.gz
--2020-04-22 09:28:51-- https://downloads.apache.org/hadoop/common/hadoop-3.2.1/hadoop-3.2.1.tar.gz
Resolving downloads.apache.org (downloads.apache.org)... 88.99.95.219, 2a01:4f8:10a:201a::2
Connecting to downloads.apache.org (downloads.apache.org)[88.99.95.219]:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 359196911 (343M) [application/x-gzip]
Saving to: 'hadoop-3.2.1.tar.gz'

hadoop-3.2.1.tar.gz  100%[=====>] 342.56M  10.2MB/s   in 31s

2020-04-22 09:29:23 (10.9 MB/s) - 'hadoop-3.2.1.tar.gz' saved [359196911/359196911]
```

Once the download is complete, extract the files to initiate the Hadoop installation:

```
tar xzf hadoop-3.3.4.tar.gz
```

The Hadoop binary files are now located within the `hadoop-3.3.4` directory.

# Single Node Hadoop Deployment (Pseudo-Distributed Mode)

Hadoop excels when deployed in a **fully distributed mode** on a large cluster of networked servers. However, if you are new to Hadoop and want to explore basic commands or test applications, you can configure Hadoop on a single node.

This setup, also called **pseudo-distributed mode**, allows each Hadoop daemon to run as a single Java process. A Hadoop environment is configured by editing a set of configuration files:

- `bashrc`
- `hadoop-env.sh`
- `core-site.xml`
- `hdfs-site.xml`
- `mapred-site.xml`
- `yarn-site.xml`

## Configure Hadoop Environment Variables (`bashrc`)

Edit the `.bashrc` shell configuration file using a text editor of your choice (we will be using `nano`):

```
sudo nano .bashrc
```

Define the Hadoop environment variables by adding the following content to the end of the file:

```
#Hadoop Related Options
export HADOOP_HOME=/home/hadoop/hadoop-3.3.4 - should be your user
export HADOOP_INSTALL=$HADOOP_HOME
export HADOOP_MAPRED_HOME=$HADOOP_HOME
export HADOOP_COMMON_HOME=$HADOOP_HOME
export HADOOP_HDFS_HOME=$HADOOP_HOME
export YARN_HOME=$HADOOP_HOME
export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/lib/native
export PATH=$PATH:$HADOOP_HOME/sbin:$HADOOP_HOME/bin
export HADOOP_OPTS="-Djava.library.path=$HADOOP_HOME/lib/native"
```

Once you add the variables, save and exit the `.bashrc` file.

```
GNU nano 2.9.3 .bashrc

if ! shopt -oq posix; then
  if [ -f /usr/share/bash-completion/bash_completion ]; then
    . /usr/share/bash-completion/bash_completion
  elif [ -f /etc/bash_completion ]; then
    . /etc/bash_completion
  fi
fi

#Hadoop Related Options

export HADOOP_HOME=/home/hdoop/hadoop-3.2.1
export HADOOP_INSTALL=$HADOOP_HOME
export HADOOP_MAPRED_HOME=$HADOOP_HOME
export HADOOP_COMMON_HOME=$HADOOP_HOME
export HADOOP_HDFS_HOME=$HADOOP_HOME
export YARN_HOME=$HADOOP_HOME
export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/lib/native
export PATH=$PATH:$HADOOP_HOME/sbin:$HADOOP_HOME/bin
export HADOOP_OPTS="-Djava.library.path=$HADOOP_HOME/lib/native"

^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
```

It is vital to apply the changes to the current running environment by using the following command:

```
source ~/.bashrc
```

## Edit `hadoop-env.sh` File

The `hadoop-env.sh` file serves as a master file to configure YARN, [HDFS](#), [MapReduce](#), and Hadoop-related project settings.

When setting up a **single node Hadoop cluster**, you need to define which Java implementation is to be utilized. Use the previously created `$HADOOP_HOME` variable to access the `hadoop-env.sh` file:

```
sudo nano $HADOOP_HOME/etc/hadoop/hadoop-env.sh
```

Uncomment the `$JAVA_HOME` variable (i.e., remove the `#` sign) and add the full path to the OpenJDK installation on your system. If you have installed the same version as presented in the first part of this tutorial, add the following line:

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

The path needs to match the location of the Java installation on your system.

```
GNU nano 2.9.3 /home/hdoop/hadoop-3.2.1/etc/hadoop/hadoop-env.sh

###

# Technically, the only required environment variable is JAVA_HOME.
# All others are optional.  However, the defaults are probably not
# preferred.  Many sites configure these options outside of Hadoop,
# such as in /etc/profile.d

# The java implementation to use.  By default, this environment
# variable is REQUIRED on ALL platforms except OS X!
export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64

# Location of Hadoop.  By default, Hadoop will attempt to determine
# this location based upon its execution path.
# export HADOOP_HOME=

# Location of Hadoop's configuration information.  i.e., where this
# file is living.  If this is not defined, Hadoop will attempt to
# locate it based upon its execution path.
#

# NOTE: It is recommend that this variable not be set here but in
# /etc/profile.d or equivalent.  Some options (such as

^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 Linter  ^_ Go To Line
```

If you need help to locate the correct Java path, run the following command in your terminal window:

```
which javac
```

The resulting output provides the path to the Java binary directory.

```
hdoop@pnap-VirtualBox:~$ which javac
/usr/bin/javac
```

Use the provided path to find the OpenJDK directory with the following command:

```
readlink -f /usr/bin/javac
```

The section of the path just before the `/bin/javac` directory needs to be assigned to the `$JAVA_HOME` variable.



```
hadoop@pnap-VirtualBox:~$ readlink -f /usr/bin/javac  
/usr/lib/jvm/java-8-openjdk-amd64/bin/javac
```

## Edit core-site.xml File

The *core-site.xml* file defines HDFS and Hadoop core properties.

To set up Hadoop in a pseudo-distributed mode, you need to **specify the URL** for your NameNode, and the temporary directory Hadoop uses for the map and reduce process.

Open the *core-site.xml* file in a text editor:

```
sudo nano $HADOOP_HOME/etc/hadoop/core-site.xml
```

Add the following configuration to override the default values for the temporary directory and add your HDFS URL to replace the default local file system setting:

```
<configuration>  
<property>  
  <name>hadoop.tmp.dir</name>  
  <value>/home/hadoop/tmpdata</value>  
</property>  
<property>  
  <name>fs.default.name</name>  
  <value>hdfs://127.0.0.1:9000</value>  
</property>  
</configuration>
```

This example uses values specific to the local system. You should use values that match your systems requirements. The data needs to be consistent throughout the configuration process.

```
GNU nano 2.9.3 /home/hdoop/hadoop-3.2.1/etc/hadoop/core-site.xml

WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License. See accompanying LICENSE file.
-->

<!-- Put site-specific property overrides in this file. -->

<configuration>
<property>
  <name>hadoop.tmp.dir</name>
  <value>/home/hdoop/tmpdata</value>
</property>
<property>
  <name>fs.default.name</name>
  <value>hdfs://127.0.0.1:9000</value>
</property>
</configuration>

^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
```

Do not forget to [create a Linux directory](#) in the location you specified for your temporary data.

## Edit hdfs-site.xml File

The properties in the *hdfs-site.xml* file govern the location for storing node metadata, fsimage file, and edit log file. Configure the file by defining the **NameNode** and **DataNode storage directories**.

Additionally, the default `dfs.replication` value of 3 needs to be changed to 1 to match the single node setup.

Use the following command to open the *hdfs-site.xml* file for editing:

```
sudo nano $HADOOP_HOME/etc/hadoop/hdfs-site.xml
```

Add the following configuration to the file and, if needed, adjust the NameNode and DataNode directories to your custom locations:

```
<configuration>
<property>
  <name>dfs.data.dir</name>
  <value>/home/hdoop/dfsdata/namenode</value>
</property>
```

```
<property>
  <name>dfs.data.dir</name>
  <value>/home/hadoop/dfsdata/datanode</value>
</property>
<property>
  <name>dfs.replication</name>
  <value>1</value>
</property>
</configuration>
```

If necessary, create the specific directories you defined for the `dfs.data.dir` value.

```
GNU nano 2.9.3 /home/hadoop/hadoop-3.2.1/etc/hadoop/hdfs-site.xml

WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License. See accompanying LICENSE file.
-->

<!-- Put site-specific property overrides in this file. -->

<configuration>
<property>
  <name>dfs.data.dir</name>
  <value>/home/hadoop/dfsdata/namenode</value>
</property>
<property>
  <name>dfs.data.dir</name>
  <value>/home/hadoop/dfsdata/datanode</value>
</property>
<property>
  <name>dfs.replication</name>
  <value>1</value>
</property>
</configuration>
```

^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

## Edit mapred-site.xml File

Use the following command to access the *mapred-site.xml* file and **define MapReduce values**:

```
sudo nano $HADOOP_HOME/etc/hadoop/mapred-site.xml
```

Add the following configuration to change the default MapReduce framework name value to `yarn`:

```
<configuration>
```

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

```
GNU nano 2.9.3 /home/hdoop/hadoop-3.2.1/etc/hadoop/mapred-site.xml Modified
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License. See accompanying LICENSE file.
-->

<!-- Put site-specific property overrides in this file. -->

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

## Edit yarn-site.xml File

The *yarn-site.xml* file is used to define settings relevant to **YARN**. It contains configurations for the **Node Manager**, **Resource Manager**, **Containers**, and **Application Master**.

Open the *yarn-site.xml* file in a text editor:

```
sudo nano $HADOOP_HOME/etc/hadoop/yarn-site.xml
```

Append the following configuration to the file:

```
<configuration>
<property>
  <name>yarn.nodemanager.aux-services</name>
  <value>mapreduce_shuffle</value>
</property>
<property>
  <name>yarn.nodemanager.aux-services.mapreduce.shuffle.class</name>
  <value>org.apache.hadoop.mapred.ShuffleHandler</value>
</property>
<property>
```

```

    <name>yarn.resourcemanager.hostname</name>
    <value>127.0.0.1</value>
  </property>
</property>
  <name>yarn.acl.enable</name>
  <value>0</value>
</property>
<property>
  <name>yarn.nodemanager.env-whitelist</name>

  <value>JAVA_HOME,HADOOP_COMMON_HOME,HADOOP_HDFS_HOME,HADOOP_CONF_
  DIR,CLASSPATH_PERPEND_DISTCACHE,HADOOP_YARN_HOME,HADOOP_MAPRED_HO
  ME</value>
</property>
</configuration>

```



```

GNU nano 2.9.3 /home/hadoop/hadoop-3.2.1/etc/hadoop/yarn-site.xml
<configuration>
<property>
  <name>yarn.nodemanager.aux-services</name>
  <value>mapreduce_shuffle</value>
</property>
<property>
  <name>yarn.nodemanager.aux-services.mapreduce.shuffle.class</name>
  <value>org.apache.hadoop.mapred.ShuffleHandler</value>
</property>
<property>
  <name>yarn.resourcemanager.hostname</name>
  <value>127.0.0.1</value>
</property>
<property>
  <name>yarn.acl.enable</name>
  <value>0</value>
</property>
<property>
  <name>yarn.nodemanager.env-whitelist</name>
  <value>JAVA_HOME,HADOOP_COMMON_HOME,HADOOP_HDFS_HOME,HADOOP_CONF_DIR,CLASSPATH_PERPEND_DISTCACHE,HADOOP_YARN_HOME,HADOOP_MAPRED_HOME</value>
</property>
</configuration>

```

## Format HDFS NameNode

It is important to **format the NameNode** before starting Hadoop services for the first time:

```
hdfs namenode -format
```

The shutdown notification signifies the end of the NameNode format process.

cd

```
hadoop@pnap-VirtualBox:~$ hdfs namenode -format
WARNING: /home/hadoop/hadoop-3.2.1/logs does not exist. Creating.
2020-04-23 06:08:13,322 INFO namenode.NameNode: STARTUP_MSG:
/*****
STARTUP_MSG: Starting NameNode
STARTUP_MSG: host = pnap-VirtualBox/127.0.1.1
STARTUP_MSG: args = [-format]
STARTUP_MSG: version = 3.2.1
STARTUP_MSG: classpath = /home/hadoop/hadoop-3.2.1/etc/hadoop:/home/hadoop/hadoop-3.2.1/
hare/hadoop/common/lib/httpcore-4.4.10.jar:/home/hadoop/hadoop-3.2.1/share/hadoop/common/
lib/curator-recipes-2.13.0.jar:/home/hadoop/hadoop-3.2.1/share/hadoop/common/lib/kerby-asn
2020-04-23 06:08:17,966 INFO namenode.NNStorageRetentionManager: Going to retain 1 images
with txid >= 0
2020-04-23 06:08:18,036 INFO namenode.FSImage: FSImageSaver clean checkpoint: txid=0 wher
meet shutdown.
2020-04-23 06:08:18,036 INFO namenode.NameNode: SHUTDOWN_MSG:
/*****
SHUTDOWN_MSG: Shutting down NameNode at pnap-VirtualBox/127.0.1.1
*****/
```

## Start Hadoop Cluster

Navigate to the *hadoop-3.3.4/sbin* directory and execute the following commands to start the NameNode and DataNode:

```
./start-dfs.sh
```

The system takes a few moments to initiate the necessary nodes.

```
hadoop@pnap-VirtualBox:~/hadoop-3.2.1/sbin$ ./start-dfs.sh
Starting namenodes on [localhost]
Starting datanodes
Starting secondary namenodes [pnap-VirtualBox]
```

Once the namenode, datanodes, and secondary namenode are up and running, start the YARN resource and nodemanagers by typing:

```
./start-yarn.sh
```

As with the previous command, the output informs you that the processes are starting.

```
hadoop@pnap-VirtualBox:~/hadoop-3.2.1/sbin$ ./start-yarn.sh
Starting resourcemanager
Starting nodemanagers
```

Type this simple command to check if all the daemons are active and running as Java processes:

**jps**

If everything is working as intended, the resulting list of running Java processes contains all the HDFS and YARN daemons.

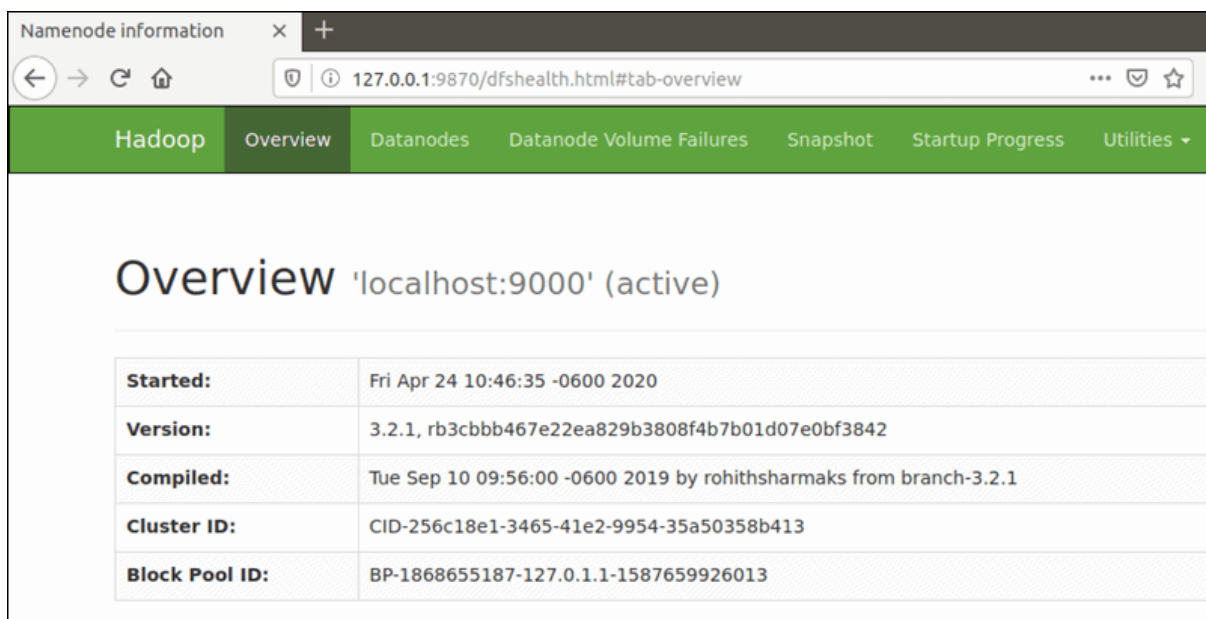
```
hadoop@pnap-VirtualBox:~/hadoop-3.2.1/sbin$ jps
469 DataNode
742 SecondaryNameNode
32759 NameNode
31180 NodeManager
31020 ResourceManager
988 Jps
```

## Access Hadoop UI from Browser

[Use your preferred browser](#) and navigate to your localhost URL or IP. The default port number **9870** gives you access to the Hadoop NameNode UI:

**<http://localhost:9870>**

The NameNode user interface provides a comprehensive overview of the entire cluster.

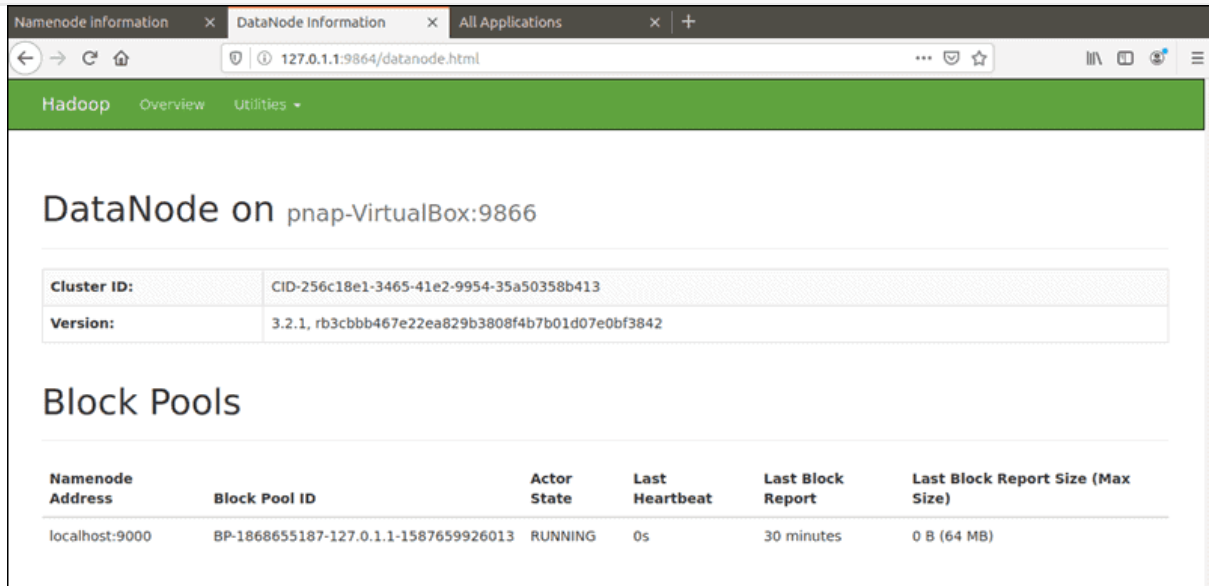


The screenshot shows a web browser window with the title 'Namenode information'. The address bar displays '127.0.0.1:9870/dfshealth.html#tab-overview'. The page has a green navigation bar with tabs: 'Hadoop', 'Overview' (selected), 'Datanodes', 'Datanode Volume Failures', 'Snapshot', 'Startup Progress', and 'Utilities'. The main content area is titled 'Overview 'localhost:9000' (active)'. Below the title is a table with the following information:

<b>Started:</b>	Fri Apr 24 10:46:35 -0600 2020
<b>Version:</b>	3.2.1, rb3cbbb467e22ea829b3808f4b7b01d07e0bf3842
<b>Compiled:</b>	Tue Sep 10 09:56:00 -0600 2019 by rohithsharmaks from branch-3.2.1
<b>Cluster ID:</b>	CID-256c18e1-3465-41e2-9954-35a50358b413
<b>Block Pool ID:</b>	BP-1868655187-127.0.1.1-1587659926013

The default port **9864** is used to access individual DataNodes directly from your browser:

<http://localhost:9864>



The screenshot shows the Hadoop DataNode Information page. The browser address bar displays `127.0.1.1:9864/datanode.html`. The page title is "DataNode on pnap-VirtualBox:9866".

**Cluster Information:**

Cluster ID:	CID-256c18e1-3465-41e2-9954-35a50358b413
Version:	3.2.1, rb3cbbb467e22ea829b3808f4b7b01d07e0bf3842

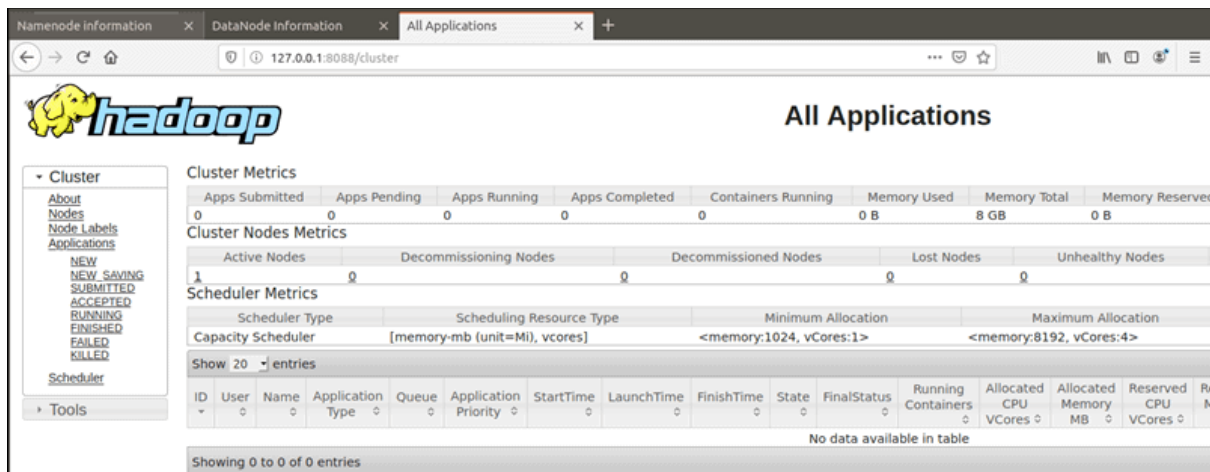
**Block Pools**

Namenode Address	Block Pool ID	Actor State	Last Heartbeat	Last Block Report	Last Block Report Size (Max Size)
localhost:9000	BP-1868655187-127.0.1.1-1587659926013	RUNNING	0s	30 minutes	0 B (64 MB)

The YARN Resource Manager is accessible on port **8088**:

<http://localhost:8088>

The Resource Manager is an invaluable tool that allows you to monitor all running processes in your Hadoop cluster.



The screenshot shows the Hadoop All Applications page. The browser address bar displays `127.0.0.1:8088/cluster`. The page title is "All Applications".

**Cluster Metrics**

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved
0	0	0	0	0	0 B	8 GB	0 B

**Cluster Nodes Metrics**

Active Nodes	Decommissioning Nodes	Decommissioned Nodes	Lost Nodes	Unhealthy Nodes
1	0	0	0	0

**Scheduler Metrics**

Scheduler Type	Scheduling Resource Type	Minimum Allocation	Maximum Allocation
Capacity Scheduler	[memory-mb (unit=Mi), vcores]	<memory:1024, vCores:1>	<memory:8192, vCores:4>

**Applications Table**

ID	User	Name	Application Type	Queue	Application Priority	StartTime	LaunchTime	FinishTime	State	FinalStatus	Running Containers	Allocated CPU VCoers	Allocated Memory MB	Reserved CPU VCoers
No data available in table														

Showing 0 to 0 of 0 entries

## Conclusion

You have successfully installed Hadoop on Ubuntu and deployed it in a pseudo-distributed mode. A single node Hadoop deployment is an excellent starting point to



explore basic HDFS commands and acquire the experience you need to design a fully distributed Hadoop cluster

Finally all nodes should be stopped...so use command

`./stop-all.sh`

---