



Published in AWS Tip



Dineshvarma Guduru [Follow](#)  
Dec 27, 2021 · 7 min read · [Listen](#)

Save



...

# Setting up Multi-Node Apache Hadoop Cluster on AWS EC2 from scratch

This article will be a demonstration of installing and running Apache Hadoop on AWS EC2 cluster.

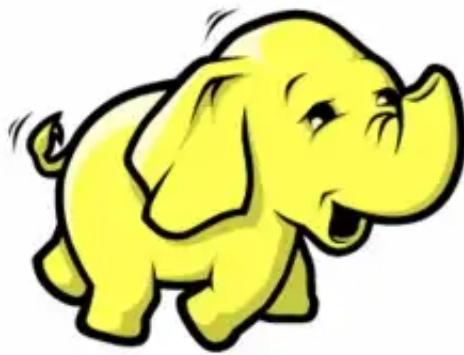
If you are looking for video resource, Checkout [this link](#).

## Pre-Requisites

AWS Account is required for following this blog. If you don't have an account already Sign Up for one from [here](#) or check out [this video](#) for full steps.



Amazon EC2



This set up involves Namenode, Secondary Namenode and 2 Datanodes.

#### Create AWS EC2 Instances

Open AWS Management Console and select Services -> Computing -> EC2 from the on top left corner.

S Services Search for services, features, blogs, docs, and more [Option+S]

## Recently visited

## Favorites

## All services

- Analytics
- Application Integration
- AR & VR
- AWS Cost Management
- Blockchain
- Business Applications
- Compute
- Containers
- Customer Enablement
- Database
- Developer Tools
- End User Computing
- Front-end Web & Mobile

## Compute

### AWS App Runner

Build and run production web applications at scale

### Batch

Fully managed batch processing at any scale

**EC2**  
Virtual Servers in the Cloud

### EC2 Image Builder

A managed service to automate build, customize and deploy OS images

### Elastic Beanstalk

Run and Manage Web Apps

### Lambda

Run Code without Thinking about Servers

### Lightsail

Launch and Manage Virtual Private Servers

### AWS Outposts

Now select Instances and Launch instances.

S Services Search for services, features, blogs, docs, and more [Option+S]

New EC Experience Tell us what you think

EC2 Dashboard

EC2 Global View

Events

Tags

Limits

Instances 1

Instances 1

Instance types

Launch Templates

Spot Requests

Savings Plans

Reserved Instances

Dedicated Hosts

Capacity Reservations

### Instances Info

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Put
You do not have any instances in this region							

Select an instance

Launch instances

Select any Amazon Machine Image (AMI) which has free tier eligible tag on it. For Hadoop installation I prefer Ubuntu Server. Select **Ubuntu Server 20.04 LTS (HVM)**.

Step 1: Choose an Amazon Machine Image (AMI)

Cancel and Exit

<input checked="" type="radio"/> Ubuntu Server 20.04 LTS (HVM), SSD Volume Type - ami-0fb653ca2d3203ac1 (64-bit x86) / ami-02af65b2d1ebdfefc (64-bit Arm)	<input type="button" value="Select"/>
Ubuntu Server 20.04 LTS (HVM), EBS General Purpose (SSD) Volume Type. Support available from Canonical ( <a href="http://www.ubuntu.com/cloud/services">http://www.ubuntu.com/cloud/services</a> ).	
Root device type: ebs    Virtualization type: hvm    ENA Enabled: Yes	
<input checked="" type="radio"/> Microsoft Windows Server 2019 Base - ami-0c95aae83dc5a60ec	<input type="button" value="Select"/>
Microsoft Windows 2019 Datacenter edition, [English]	
Root device type: obs    Virtualization type: hvm    ENA Enabled: Yes	64-bit (x86)
<input checked="" type="radio"/> Microsoft Windows Server 2019 Base with Containers - ami-0ba02e7ab1c9dd052	<input type="button" value="Select"/>
Microsoft Windows 2019 Datacenter edition with Containers, [English]	
Root device type: obs    Virtualization type: hvm    ENA Enabled: Yes	64-bit (x86)

## Next: Choose an Instance Type

Choose the instance that has Free tier eligible on it. We only have **t2.micro** with only 1 CPU and 1 GB Memory. If you want better configuration, you might get charged for using the other instance types.

### Step 2: Choose an Instance Type

Amazon EC2 provides a wide selection of instance types optimized to fit different use cases. Instances are virtual servers that can run applications. They have varying combinations of CPU, memory, storage, and networking capacity, and give you the flexibility to choose the appropriate mix of resources for your applications. [Learn more about instance types and how they can meet your computing needs.](#)

Filter by: All Instance families ▾ Current generation ▾ Show/Hide Columns

Currently selected: t2.micro (- ECUs, 1 vCPUs, 2.5 GHz, ~ 1 GiB memory, EBS only)

Family	Type	vCPUs	Memory (GiB)	Instance Storage (GB)	EBS-Optimized Available	Network Performance	IPv6 Support
t2	t2.nano	1	0.5	EBS only	-	Low to Moderate	Yes
t2	<b>t2.micro</b> Free tier eligible	1	1	EBS only	-	Low to Moderate	Yes

## Next: Configure Instance Details

Provide the number of instances as 4 and select one one of the available subnet options so that all the 4 instances will be in the same region and leave the rest of the options as same

1. Choose AMI    2. Choose Instance Type    3. Configure Instance    4. Add Storage    5. Add Tags    6. Configure Security Group    7. Review

### Step 3: Configure Instance Details

Configure the instance to suit your requirements. You can launch multiple instances from the same AMI, request Spot instances to take advantage of the lower pricing, assign an access management role to the instance, or more.

Number of instances  Launch into Auto Scaling Group

You may want to consider launching these instances into an Auto Scaling Group to help you maintain application availability and for easy scaling in the future. [Learn how Auto Scaling can help your application stay healthy and cost effective.](#)

Purchasing option  Request Spot Instances

Network    Create new VPC

Subnet    Create new subnet  
4091 IP Addresses available

Auto-assign Public IP  Use subnet setting (Enable)

## Next: Add Storage

By default 8 GB of storage will be allotted. If you need more storage you may Add New Volume or change the size based on the requirements. For Hadoop installation and learning purpose 8 GB will be sufficient.

1. Choose AMI    2. Choose Instance Type    3. Configure Instance    4. Add Storage    5. Add Tags    6. Configure Security Group    7. Review

#### Step 4: Add Storage

Your instance will be launched with the following storage device settings. You can attach additional EBS volumes and Instance store volumes to your instance, or edit the settings of the root volume. You can also attach additional EBS volumes after launching an instance, but not instance store volumes. Learn more about storage options in Amazon EC2.

Volume Type	Device	Snapshot	Size (GiB)	Volume Type	IOPS	Throughput (MB/s)	Delete on Termination	Encryption
Root	/dev/sda1	snap-092498c2cc2e3eb82	8	General Purpose SSD (gp2)	100 / 3000	N/A	<input checked="" type="checkbox"/>	Not Encrypted

[Add New Volume](#)

#### Next: Add Tags

Give some tag to the instance to identify the instance with the name we choose, We can later change the Name “Hadoop” to Namenode, SNN and Datanodes once instances are launched

1. Choose AMI    2. Choose Instance Type    3. Configure Instance    4. Add Storage    5. Add Tags    6. Configure Security Group    7. Review

#### Step 5: Add Tags

A tag consists of a case-sensitive key-value pair. For example, you could define a tag with key = Name and value = Webserver.

A copy of a tag can be applied to volumes, instances or both.

Tags will be applied to all instances and volumes. Learn more about tagging your Amazon EC2 resources.

Key	(128 characters maximum)	Value	(256 characters maximum)	Instances	Volumes	Network Interfaces
Name		Hadoop		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

#### Next: Configure Security Group

Select Create a new security group and type as All traffic so the security group will be open for testing.

1. Choose AMI    2. Choose Instance Type    3. Configure Instance    4. Add Storage    5. Add Tags    6. Configure Security Group    7. Review

#### Step 6: Configure Security Group

A security group is a set of firewall rules that control the traffic for your instance. On this page, you can add rules to allow specific traffic to reach your instance. For example, if you want to set up a web server and allow Internet traffic to reach your instance, add rules that allow unrestricted access to the HTTP and HTTPS ports. You can create a new security group or select from an existing one below. Learn more about Amazon EC2 security groups.

Assign a security group:  Create a new security group

Select an existing security group

Security group name: launch-wizard-5

Description: launch-wizard-5 created 2021-12-27T14:44:30.938+05:30

Type	Protocol	Port Range	Source	Description
All traffic	All	0 - 65535	Custom 0.0.0.0/0	e.g. SSH for Admin Desktop

**Review and Launch :** Review all the information selected again and proceed with **Launch** then you will be prompted with the below screen.

Select **Create new key pair** → Type as **RSA** → Give a name to the key pair and click on download the key-pair. It will download .pem file which is used to access the EC2 instances then **Launch Instances**

## Select an existing key pair or create a new key pair X

A key pair consists of a **public key** that AWS stores, and a **private key file** that you store. Together, they allow you to connect to your instance securely. For Windows AMIs, the private key file is required to obtain the password used to log into your instance. For Linux AMIs, the private key file allows you to securely SSH into your instance. Amazon EC2 supports ED25519 and RSA key pair types.

Note: The selected key pair will be added to the set of keys authorized for this instance. Learn more about [removing existing key pairs from a public AMI](#).

▼

**Key pair type**  
 RSA  ED25519

**Key pair name**

You have to download the **private key file** (\*.pem file) before you can continue. **Store it in a secure and accessible location.** You will not be able to download the file again after it's created.

[Cancel](#) Launch Instances

Click on **View instances** and select instance to see the properties through which we can see the ip address and dns of our instance.

Instances (1/4) [Info](#)

C Connect Instance state Actions Launch instances

Search

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
<input checked="" type="checkbox"/> Hadoop	i-03b06b6009b922dea	<span>Running</span>	t2.micro	-	No alarms	+ us-east-2c
<input type="checkbox"/> Hadoop	i-0ea7076291adc\$27c	<span>Running</span>	t2.micro	-	No alarms	+ us-east-2c
<input type="checkbox"/> Hadoop	i-0c88c3376ef6f9aac	<span>Running</span>	t2.micro	-	No alarms	+ us-east-2c
<input type="checkbox"/> Hadoop	i-06471d296b7bcfa7a	<span>Running</span>	t2.micro	-	No alarms	+ us-east-2c

Instance: i-03b06b6009b922dea (Hadoop)

Details Security Networking Storage Status checks Monitoring Tags

Instance summary [Info](#)

Instance ID  
 i-03b06b6009b922dea (Hadoop)

IPv6 address -

Public IPv4 address  
 3.138.109.90 | [open address](#) [🔗](#)

Instance state Running

Private IPv4 addresses  
 172.31.33.110

Public IPv4 DNS  
 ec2-3-138-109-90.us-east-2.compute.amazonaws.com | [open address](#) [🔗](#)

Click on Edit option near to the Name and rename the instances as **Namenode**, **SNN**, **Datanode1** and **Datanode2**.

Instances (4)		Info		Connect	Instance state	Actions	Launch Instances			
			Search							
	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Pub		
<input type="checkbox"/>	Namenode	i-03b06b6009b922dea	Running		t2.micro	-	No alarms		us-east-2c	ec2-
<input type="checkbox"/>	SNN	i-0ea7076291adc327c	Running		t2.micro	-	No alarms		us-east-2c	ec2-
<input type="checkbox"/>	Datanode1	i-0c88c3376ef6f9aac	Running		t2.micro	-	No alarms		us-east-2c	ec2-
<input type="checkbox"/>	Datanode2	i-06471d296b7bcfa7a	Running		t2.micro	-	No alarms		us-east-2c	ec2-

Now that the instances are up and running, We need to setup Hadoop in these instances. Before setting up Hadoop in this instances make sure you are able to connect to these EC2 instances with SSH.

If you are not aware of how to SSH, follow any of the below article/video

## Connecting to EC2 instance using SSH from Mac or Linux

[Video on Creating AWS Account, launch EC2 instance and connect using SSH from Mac/ Linux](#)

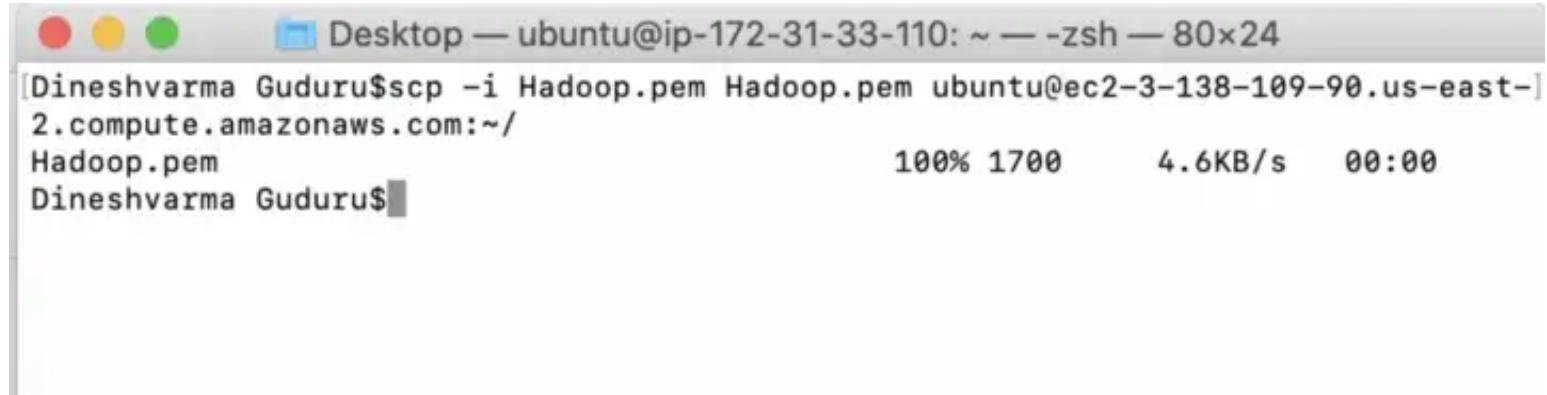
## Video on establishing SSH to EC2 instance from Windows

## Copy Keypair (xxxx.pem) from Local to Namenode

We need the Keypair (xxxx.pem) in the Namenode for the namenode to access the Datanodes

```
scp -i <path_to_keypair> <path_to_keypair> user@<ec2_namenode_dns>:<path_at_ec2>
```

```
scp -i Hadoop.pem Hadoop.pem ubuntu@ec2-3-138-109-90.us-east-2.compute.amazonaws.com:/
```



A screenshot of a terminal window titled "Desktop — ubuntu@ip-172-31-33-110: ~ — zsh — 80x24". The command "scp -i Hadoop.pem Hadoop.pem ubuntu@ec2-3-138-109-90.us-east-2.compute.amazonaws.com:~/Hadoop.pem" is being typed. The progress bar shows 100% completion at 1700 bytes, with a transfer rate of 4.6KB/s and a total time of 00:00. The prompt "Dineshvarma Guduru\$ " is visible at the bottom.

### All Nodes:

The below activities to be performed in all the nodes one by one.

**Edit the hosts File:** Replace the first line with

```
<ec2_private_ipv4> <ec2_dns>
```

```
ubuntu@ip-172-31-33-110:~$ sudo vi /etc/hosts  
172.31.33.110 ec2-3-138-109-90.us-east-2.compute.amazonaws.com
```

● ● ● Desktop — ubuntu@ip-172-31-33-110: ~ — ssh -i Hadoop.pem ubuntu@ec2-...

172.31.33.110 ec2-3-138-109-90.us-east-2.compute.amazonaws.com

Update OS with latest available patches in all nodes

```
sudo apt-get update && sudo apt-get -y dist-upgrade
```

See "man sudo\_root" for details.

```
ubuntu@ip-172-31-35-238:~$ sudo apt-get update && sudo apt-get -y dist-upgrade [1]
Hit:1 http://us-east-2.ec2.archive.ubuntu.com/ubuntu focal InRelease
Get:2 http://us-east-2.ec2.archive.ubuntu.com/ubuntu focal-updates InRelease [114 kB]
Get:3 http://us-east-2.ec2.archive.ubuntu.com/ubuntu focal-backports InRelease [108 kB]
Get:4 http://us-east-2.ec2.archive.ubuntu.com/ubuntu focal/universe amd64 Packages [8628 kB]
Get:5 http://security.ubuntu.com/ubuntu focal-security InRelease [114 kB]
Get:6 http://us-east-2.ec2.archive.ubuntu.com/ubuntu focal/universe Translation-en [5124 kB]
Get:7 http://us-east-2.ec2.archive.ubuntu.com/ubuntu focal/universe amd64 c-n-f Metadata [265 kB]
Get:8 http://us-east-2.ec2.archive.ubuntu.com/ubuntu focal/multiverse amd64 Packages [144 kB]
Get:9 http://us-east-2.ec2.archive.ubuntu.com/ubuntu focal/multiverse Translation-en [104 kB]
Get:10 http://us-east-2.ec2.archive.ubuntu.com/ubuntu focal/multiverse amd64 c-n-f Metadata [9136 B]
Get:11 http://us-east-2.ec2.archive.ubuntu.com/ubuntu focal-updates/main amd64 Packages [1400 kB]
```

Install Java in all nodes

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

```
guduru.varma — ubuntu@ip-172-31-35-238: ~ — ssh -i ~/Desktop/Hadoop.pe...
[ubuntu@ip-172-31-35-238:~]$ sudo apt-get -y install openjdk-8-jdk-headless
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  ca-certificates-java fontconfig-config fonts-dejavu-core java-common
  libavahi-client3 libavahi-common-data libavahi-common3 libcups2
  libfontconfig1 libjpeg-turbo8 libjpeg8 liblcms2-2 libpcsslite1 libxi6
  libxrender1 libxtst6 openjdk-8-jre-headless x11-common
Suggested packages:
  default-jre cups-common liblcms2-utils pcscd openjdk-8-demo openjdk-8-source
  libnss-mdns fonts-dejavu-extra fonts-ipafont-gothic fonts-ipafont-mincho
  fonts-wqy-microhei fonts-wqy-zenhei fonts-indic
The following NEW packages will be installed:
  ca-certificates-java fontconfig-config fonts-dejavu-core java-common
  libavahi-client3 libavahi-common-data libavahi-common3 libcups2
  libfontconfig1 libjpeg-turbo8 libjpeg8 liblcms2-2 libpcsslite1 libxi6
  libxrender1 libxtst6 openjdk-8-jdk-headless openjdk-8-jre-headless
  x11-common
0 upgraded, 19 newly installed, 0 to remove and 0 not upgraded.
Need to get 38.4 MB of archives.
After this operation, 150 MB of additional disk space will be used.
Get:1 http://us-east-2.ec2.archive.ubuntu.com/ubuntu focal/main amd64 java-commo
n all 0.72 [6816 B]
```

Download and install Hadoop in all nodes

```
step1: wget https://downloads.apache.org/hadoop/common/hadoop-3.2.2/hadoop-
3.2.2.tar.gz
step2: tar xvzf hadoop-3.2.2.tar.gz
step3: mv hadoop-3.2.2 hadoop
```

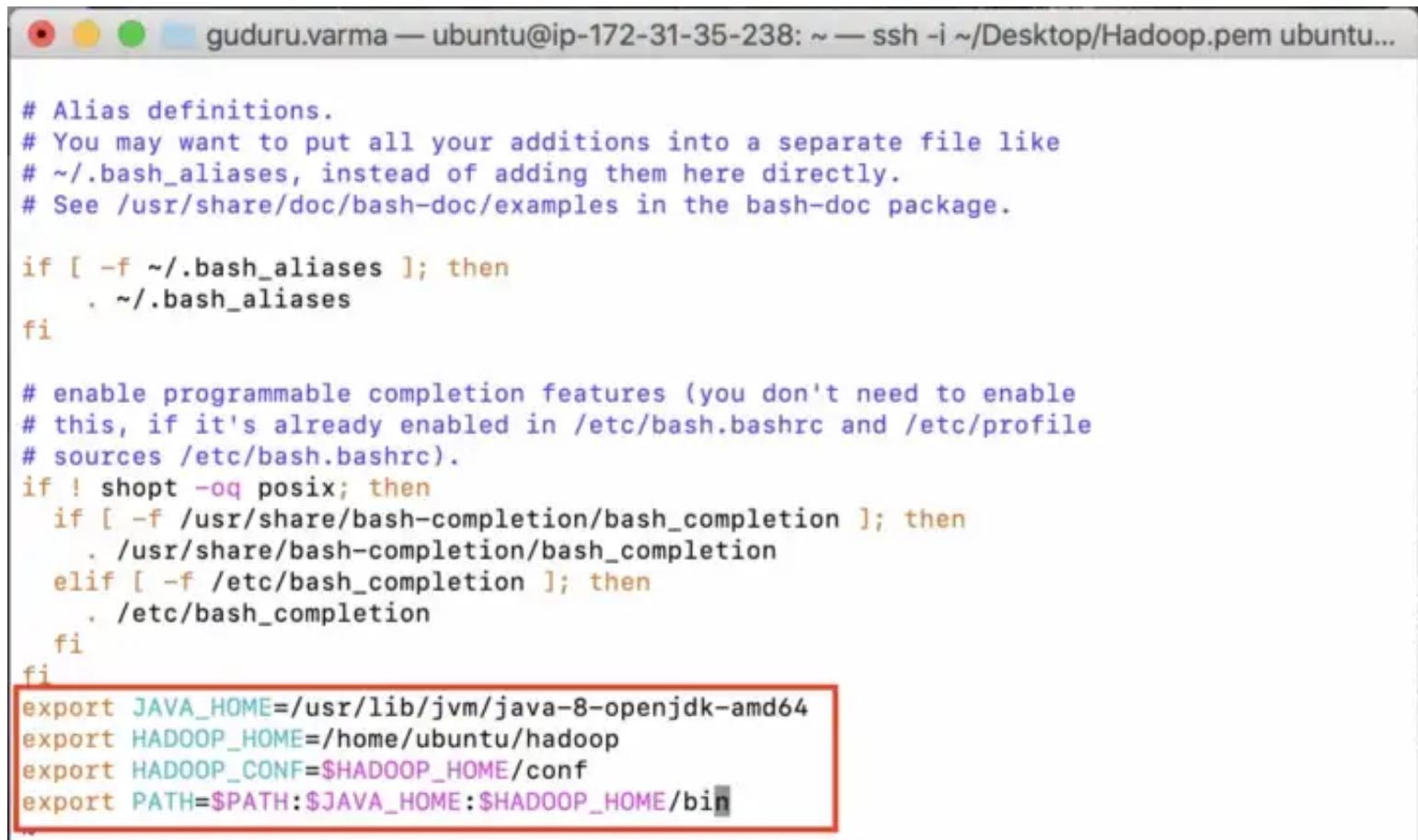
```
[ubuntu@ip-172-31-35-238:~]$ wget https://downloads.apache.org/hadoop/common/hadoop-3.2.2/
hadoop-3.2.2.tar.gz
--2021-12-28 02:08:24-- https://downloads.apache.org/hadoop/common/hadoop-3.2.2/hadoop-
3.2.2.tar.gz
Resolving downloads.apache.org (downloads.apache.org)... 135.181.214.104, 88.99.95.219,
2a01:4f8:10a:201a::2, ...
Connecting to downloads.apache.org (downloads.apache.org)|135.181.214.104|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 395448622 (377M) [application/x-gzip]
Saving to: 'hadoop-3.2.2.tar.gz'

hadoop-3.2.2.tar.gz    100%[=====] 377.13M  11.2MB/s    in 32s

2021-12-28 02:08:57 (11.7 MB/s) - 'hadoop-3.2.2.tar.gz' saved [395448622/395448622]
```

Modify `~/.bashrc` in all nodes and add below lines at the end

```
export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
export HADOOP_HOME=/home/ubuntu/hadoop
export HADOOP_CONF=$HADOOP_HOME/conf
export PATH=$PATH:$JAVA_HOME:$HADOOP_HOME/bin
```



A screenshot of a terminal window titled "guduru.varma — ubuntu@ip-172-31-35-238: ~ — ssh -i ~/Desktop/Hadoop.pem ubuntu...". The window displays the contents of the .bashrc file. A red box highlights the four export commands at the bottom of the file:

```
# Alias definitions.
# You may want to put all your additions into a separate file like
# ~/.bash_aliases, instead of adding them here directly.
# See /usr/share/doc/bash-doc/examples in the bash-doc package.

if [ -f ~/.bash_aliases ]; then
    . ~/.bash_aliases
fi

# enable programmable completion features (you don't need to enable
# this, if it's already enabled in /etc/bash.bashrc and /etc/profile
# sources /etc/bash.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

export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
export HADOOP_HOME=/home/ubuntu/hadoop
export HADOOP_CONF=$HADOOP_HOME/conf
export PATH=$PATH:$JAVA_HOME:$HADOOP_HOME/bin
```

Now run *source ~/.bashrc* in all nodes

Run **hadoop** command to check if hadoop is installed correctly in all nodes

You should now see something like below, which means hadoop is installed correctly.

```
[ubuntu@ip-172-31-33-110:~$ hadoop
Usage: hadoop [OPTIONS] SUBCOMMAND [SUBCOMMAND OPTIONS]
or      hadoop [OPTIONS] CLASSNAME [CLASSNAME OPTIONS]
where CLASSNAME is a user-provided Java class

OPTIONS is none or any of:

--config dir          Hadoop config directory
--debug               turn on shell script debug mode
--help                usage information
buildpaths           attempt to add class files from build tree
hostnames list[,of,host,names] hosts to use in slave mode
hosts filename       list of hosts to use in slave mode
loglevel level       set the log4j level for this command
workers              turn on worker mode
```

## Setup Password-less SSH on servers

On Namenode run **ssh-keygen** and use the default location for the key and hit enter twice for an empty passphrase. The **rsa\_id** will now be available on Namenode as **/home/ubuntu/.ssh/id\_rsa.pub**

```
[ubuntu@ip-172-31-35-238:~$ ssh-keygen
Generating public/private rsa key pair.
[Enter file in which to save the key (/home/ubuntu/.ssh/id_rsa):
[Enter passphrase (empty for no passphrase):
[Enter same passphrase again:
Your identification has been saved in /home/ubuntu/.ssh/id_rsa
Your public key has been saved in /home/ubuntu/.ssh/id_rsa.pub
The key fingerprint is.
SHA256:kUW2j4I3YWK2m57GseqhnJ4pBv43pnGd05KEQk+KgA ubuntu@ip-172-31-35-238
The key's randomart image is:
+---[RSA 3072]----+
|E .     ..+
|... . o + .
|. o o + o .
|.. o + o . o
|o . o . S ..
|oo..... o
|=oo+o
|==++*o.
|=BB==o.
+---[SHA256]----+
```

Copy the above file to all the Nodes. Command to copy is shown below which needs to be executed in Namenode thrice, once for each node.

```
scp -i Hadoop.pem /home/ubuntu/.ssh/id_rsa.pub ubuntu@ec2-3-139-95-172.us-east-2.compute.amazonaws.com:/home/ubuntu/.ssh/id_rsa.pub
```

## On Datanodes, Move the rsa\_id to the authorized keys

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

**Setup SSH config in Namenode:** Add below lines in `~/.ssh/config`

```
vi ~/.ssh/config
```

```
Host nnode
  HostName <nnode dns>
  User ubuntu
  IdentityFile ~/.ssh/id_rsa

Host snode
  HostName <snn dns>
  User ubuntu
  IdentityFile ~/.ssh/id_rsa

Host dnode2
  HostName <dnode1 dns>
  User ubuntu
  IdentityFile ~/.ssh/id_rsa

Host dnode3
  HostName <dnode2 dns>
  User ubuntu
  IdentityFile ~/.ssh/id_rsa
```

The screenshot shows a terminal window with the following content:

```
guduru.varma — ubuntu@ip-172-31-35-238: ~ — ssh -i ~/Desktop/Hadoop.pem ubuntu...
```

The configuration file contains four host definitions:

- Host nnode**:  
HostName ec2-3-19-185-196.us-east-2.compute.amazonaws.com (highlighted)  
User ubuntu  
IdentityFile ~/.ssh/id\_rsa
- Host snode**:  
HostName ec2-3-16-107-62.us-east-2.compute.amazonaws.com (highlighted)  
User ubuntu  
IdentityFile ~/.ssh/id\_rsa
- Host dnode1**:  
HostName ec2-18-117-115-147.us-east-2.compute.amazonaws.com (highlighted)  
User ubuntu  
IdentityFile ~/.ssh/id\_rsa
- Host dnode2**:  
HostName ec2-3-139-95-172.us-east-2.compute.amazonaws.com (highlighted)  
User ubuntu  
IdentityFile ~/.ssh/id\_rsa

Annotations with green boxes and arrows point from the host names in the configuration to their corresponding DNS addresses in the terminal window:

- An arrow points from "Namenode dns" to the "nnode" host definition.
- An arrow points from "Secondary Namenode dns" to the "snode" host definition.
- An arrow points from "Datanode1 dns" to the "dnode1" host definition.
- An arrow points from "Datanode2 dns" to the "dnode2" host definition.

Create data directories in all nodes

```
sudo mkdir -p /usr/local/hadoop/hdfs/data  
sudo chown -R ubuntu:ubuntu /usr/local/hadoop/hdfs/data
```

```
[ubuntu@ip-172-31-35-238:~$ sudo mkdir -p /usr/local/hadoop/hdfs/data  
ubuntu@ip-172-31-35-238:~$ sudo chown -R ubuntu:ubuntu /usr/local/hadoop/hdfs/data
```

## Set up Hadoop cluster

Hadoop cluster needs below 7 files to be configured

```
$HADOOP_HOME/etc/hadoop/hadoop-env.sh  
$HADOOP_HOME/etc/hadoop/core-site.xml  
$HADOOP_HOME/etc/hadoop/hdfs-site.xml  
$HADOOP_HOME/etc/hadoop/mapred-site.xml  
$HADOOP_HOME/etc/hadoop/yarn-site.xml  
$HADOOP_HOME/etc/hadoop/masters  
$HADOOP_HOME/etc/hadoop/slaves
```

Add below line in \$HADOOP\_HOME/etc/hadoop/hadoop-env.sh

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

```
guduru.varma — ubuntu@ip-172-31-35-238: ~ — ssh -i ~/Desktop/Hadoop.pem ubuntu...  
###  
# Generic settings for HADOOP  
###  
  
# 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
```

Add below lines in \$HADOOP\_HOME/etc/hadoop/core-site.xml

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

```
guduru.varma — ubuntu@ip-172-31-35-238: ~ — ssh -i ~/Desktop/Hadoop.pem ubuntu...
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<!--
 Licensed under the Apache License, Version 2.0 (the "License");
 you may not use this file except in compliance with the License.
 You may obtain a copy of the License at

 http://www.apache.org/licenses/LICENSE-2.0

 Unless required by applicable law or agreed to in writing, software
 distributed under the License is distributed on an "AS IS" BASIS,
 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>fs.defaultFS</name>
    <value>hdfs://ec2-18-223-131-252.us-east-2.compute.amazonaws.com:9000</value>
  </property>
</configuration>
```

Add below lines in \$HADOOP\_HOME/etc/hadoop/hdfs-site.xml

```
<configuration>
  <property>
    <name>dfs.replication</name>
    <value>3</value>
  </property>
  <property>
    <name>dfs.namenode.name.dir</name>
    <value>file:///usr/local/hadoop/hdfs/data</value>
  </property>
</configuration>
```

```
<!--  
 Licensed under the Apache License, Version 2.0 (the "License");  
 you may not use this file except in compliance with the License.  
 You may obtain a copy of the License at  
  
 http://www.apache.org/licenses/LICENSE-2.0  
  
 Unless required by applicable law or agreed to in writing, software  
 distributed under the License is distributed on an "AS IS" BASIS,  
 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.replication</name>  
    <value>3</value>  
  </property>  
  <property>  
    <name>dfs.namenode.name.dir</name>  
    <value>file:///usr/local/hadoop/hdfs/data</value>  
  </property>  
</configuration>
```

Add below lines in \$HADOOP\_HOME/etc/hadoop/mapred-site.xml

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



```
guddu@varma:~$ ssh -T ~/_Desktop/Hadoop.pem ubuntu...
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<!--
 Licensed under the Apache License, Version 2.0 (the "License");
 you may not use this file except in compliance with the License.
 You may obtain a copy of the License at

 http://www.apache.org/licenses/LICENSE-2.0

 Unless required by applicable law or agreed to in writing, software
 distributed under the License is distributed on an "AS IS" BASIS,
 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>
```

Add below lines in \$HADOOP\_HOME/etc/hadoop/yarn-site.xml

```
<configuration>
  <!-- Site specific YARN configuration properties -->
  <property>
    <name>yarn.nodemanager.aux-services</name>
    <value>mapreduce_shuffle</value>
  </property>
  <property>
    <name>yarn.resourcemanager.hostname</name>
    <value><nnode></value>
  </property>
</configuration>
```

guduru.varma — ubuntu@ip-172-31-35-238: ~ — ssh -i ~/Desktop/Hadoop.pem ubuntu...

Licensed under the Apache License, Version 2.0 (the "License");  
you may not use this file except in compliance with the License.  
You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software  
distributed under the License is distributed on an "AS IS" BASIS,  
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.

-->

<configuration>

```
<!-- Site specific YARN configuration properties -->
<property>
  <name>yarn.nodemanager.aux-services</name>
  <value>mapreduce_shuffle</value>
</property>
<property>
  <name>yarn.resourcemanager.hostname</name>
  <value>ec2-18-223-131-252.us-east-2.compute.amazonaws.com</value>
</property>
```

</configuration>

## Move configuration files to Slaves

```
scp hadoop-env.sh core-site.xml hdfs-site.xml mapred-site.xml yarn-site.xml
<user>@<dnode>:/home/ubuntu/hadoop/conf
```

```
scp hadoop-env.sh core-site.xml hdfs-site.xml mapred-site.xml yarn-site.xml
ubuntu@ec2-54-209-221-47.compute-1.amazonaws.com:/home/ubuntu/hadoop/conf
```

## Now on Namenode and Secondary Namenode edit below 2 files

Add below lines in \$HADOOP\_HOME/etc/hadoop/masters

```
<nnode>
<snnode>
```

```
guduru.varma — ubuntu@ip-172-31-35-238: ~ — ssh -i ~/Desktop/Hadoop.pem ubuntu...
ec2-3-19-185-196.us-east-2.compute.amazonaws.com dns of Namenode
ec2-3-16-107-62.us-east-2.compute.amazonaws.com dns of Secondary Namenode
```

Add below lines in \$HADOOP\_HOME/etc/hadoop/slaves

```
<dnode1>
<dnode2>
```

```
guduru.varma — ubuntu@ip-172-31-35-238: ~ — ssh -i ~/Desktop/Hadoop.pem ubuntu...
ec2-18-117-115-147.us-east-2.compute.amazonaws.com dns of Datanode1
ec2-3-139-95-172.us-east-2.compute.amazonaws.com dns of Datanode1
```

## Datanode

\$HADOOP\_HOME/etc/hadoop/masters will be empty in Datanodes

Add below lines in \$HADOOP\_HOME/etc/hadoop/slaves

```
<dnode_dns>
```

```
guduru.varma — ubuntu@ip-172-31-35-238: ~ — ssh -i ~/Desktop/Hadoop.pem ubuntu...
ec2-18-117-115-147.us-east-2.compute.amazonaws.com dns of datanode
```

Each Datanode will have only it's own dns in slaves file

# Hadoop Daemon Startup

```
Namenode: hdfs namenode -format  
Namenode: $HADOOP_HOME/sbin/start-dfs.sh  
Namenode: $HADOOP_HOME/sbin/start-yarn.sh  
Namenode: $HADOOP_HOME/sbin/mr-jobhistory-daemon.sh start historyserver
```

```
[ubuntu@ip-172-31-35-238:~$ $HADOOP_HOME/sbin/start-dfs.sh  
Starting namenodes on [ec2-18-223-131-252.us-east-2.compute.amazonaws.com]  
ec2-18-223-131-252.us-east-2.compute.amazonaws.com: Warning: Permanently added 'ec2-18-  
23-131-252.us-east-2.compute.amazonaws.com' (ECDSA) to the list of known hosts.  
Starting datanodes  
Starting secondary namenodes [ip-172-31-35-238]  
[ubuntu@ip-172-31-35-238:~$ $HADOOP_HOME/sbin/start-yarn.sh  
Starting resourcemanager  
Starting nodemanagers  
[ubuntu@ip-172-31-35-238:~$ jps  
2432 Jps  
2313 NodeManager  
1802 DataNode  
1998 SecondaryNameNode
```

Now that we have started Hadoop in Cluster mode and will be able to check the services running in all the nodes by using jps command as shown above

Once the cluster is running, we can check the web UI for the status of the data nodes. Go to <nnode>:50070 for the Web UI and verify that the datanodes added are online

Hadoop services can be stopped using \$HADOOP\_HOME/sbin/stop-all.sh command.

```
[ubuntu@ip-172-31-35-238:~$ $HADOOP_HOME/sbin/stop-all.sh  
WARNING: Stopping all Apache Hadoop daemons as ubuntu in 10 seconds.  
WARNING: Use CTRL-C to abort.  
Stopping namenodes on [ec2-18-223-131-252.us-east-2.compute.amazonaws.com]  
Stopping datanodes  
Stopping secondary namenodes [ip-172-31-35-238]  
Stopping nodemanagers  
localhost: WARNING: nodemanager did not stop gracefully after 5 seconds: Trying to kill  
with kill -9  
Stopping resourcemanager
```

More detailed video demonstration is available in [this link](#).