
Install And Configure Jenkins on AWS

[Edition 04]

[Last Update 220920]

K21Academy

Contents

1	Introduction	3
2	Documentation Links.....	4
3	Pre-requisite	5
4	Launching a Linux Instance Using EC2.....	6
4.1	Creating an EC2 Instance.....	6
4.2	Accessing EC2 Instance Using Browser SSH Connection.....	14
4.3	Accessing EC2 Instance Using Putty	16
5	Install And Configure Jenkins.....	29
5.1	Connect To Your Linux Instance.....	29
5.2	Install JDK on Linux Instance.....	29
5.3	Download And Install Jenkins.....	Error! Bookmark not defined.
5.4	Configure Jenkins.....	Error! Bookmark not defined.
5.5	Useful Commands (Start/Stop, Update, Uninstall)	38
6	Deleting/Cleanup	39
6.1	Stopping the Linux Instance.....	39
6.2	Terminating the Linux Instance	42
7	Troubleshooting	44
8	Summary.....	46

1 INTRODUCTION

Jenkins is an open-source automation server that integrates with a number of AWS Services, such as AWS CodeCommit, AWS CodeDeploy, Amazon EC2 Spot, and Amazon EC2 Fleet. You can use Amazon Elastic Compute Cloud (Amazon EC2) to deploy a Jenkins application on AWS in a matter of minutes.

This tutorial walks you through the process of deploying a Jenkins application. You will launch an EC2 instance, install Jenkins on that instance, and configure Jenkins to automatically spin up Jenkins build slave instances if build abilities need to be augmented on the instance.

Amazon EC2 presents a true virtual computing environment, allowing you to use web service interfaces to launch instances with a variety of operating systems, load them with your custom application environment, manage your network's access permissions, and run your image using as many or few systems as you desire.

This activity guide cover steps for:

- Step 1: Launch an EC2 Instance
- Step 2: Install JDK
- Step 3: Install and Configure Jenkins
- Step 4: Clean Up

2 DOCUMENTATION LINKS

1. Amazon EC2
<https://aws.amazon.com/ec2/>
2. Features of Amazon EC2
<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/concepts.html>
3. Amazon EC2 FAQs
<https://aws.amazon.com/ec2/faqs/>
4. Getting Started with Amazon EC2 linux Instances
https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EC2_GetStarted.html
5. Jenkins on AWS (whitepaper)
https://d0.awsstatic.com/whitepapers/DevOps/Jenkins_on_AWS.pdf
6. DevOps and AWS
<https://aws.amazon.com/devops/>

3 PRE-REQUISITE

1. An AWS Account (Free or Paid).

K21Academy

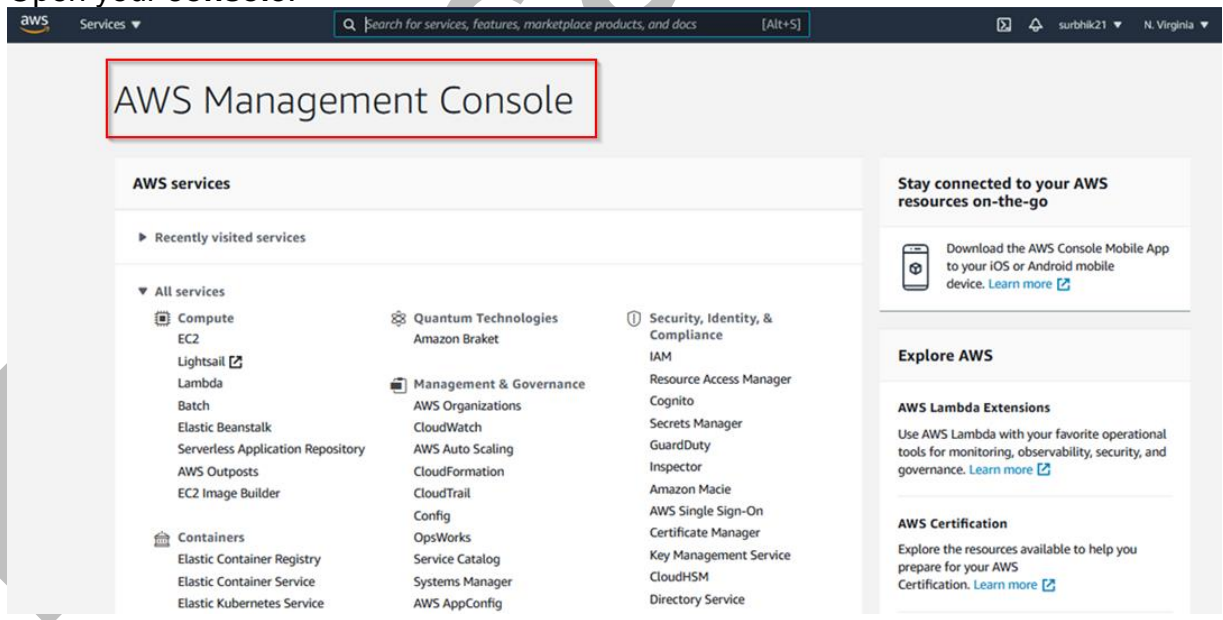
4 LAUNCHING A LINUX INSTANCE USING EC2

Note: Launching a Linux Instance means you will create a Virtual Machine and access it using tools like Putty, MobaXterm etc. You shall be accessing this Virtual machine using internet, accessing on internet means these machines are physically present in the datacenters that lie in different Regions in the world. A datacenter is a place used to house computer systems and associated components, such as routers, switches, firewalls, storage systems, servers etc. So, you are basically accessing the machine present in such datacenters from anywhere in the world via internet. The launching of an instance does not depend on the underlying infrastructure whether your Laptop is having Windows or macOS.

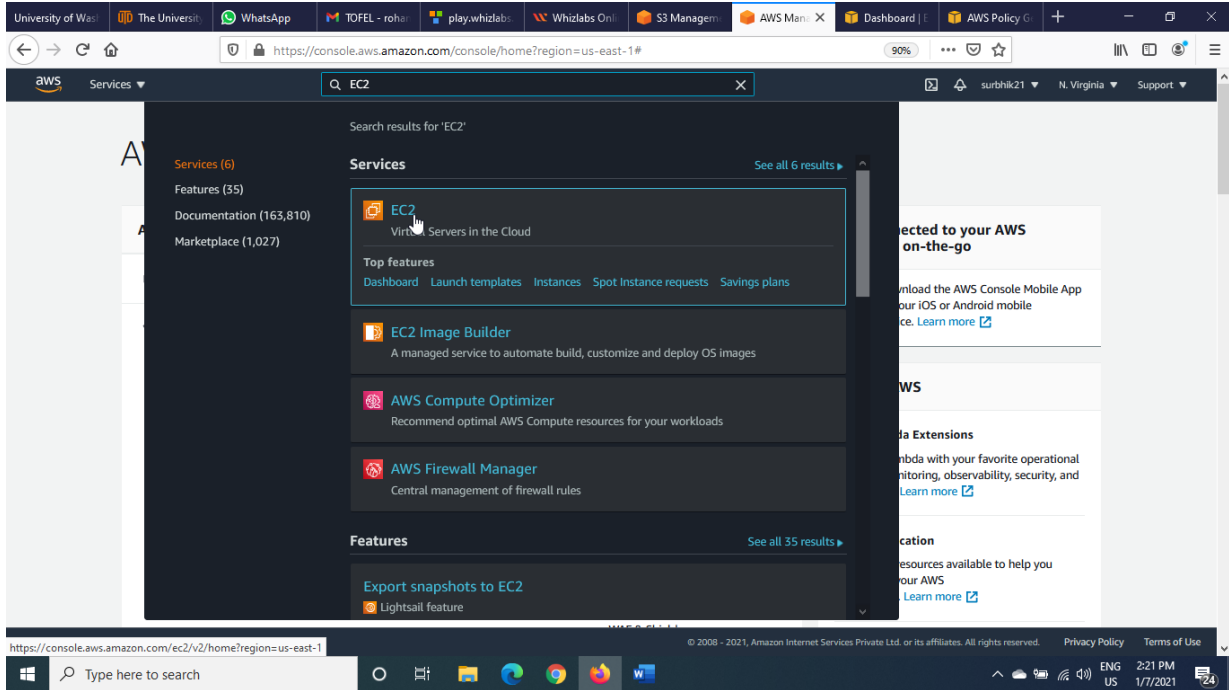
4.1 Creating an EC2 Instance

Note: Amazon EC2 basically provides a **Virtual Computing environment**, where they have pre-configured AMI (Amazon Machine Image) from which you can launch Virtual machines. In this section, we shall see the steps to create an configure the Linux Ec2 instance from a pre-configured AMI named Linux-2 AMI.

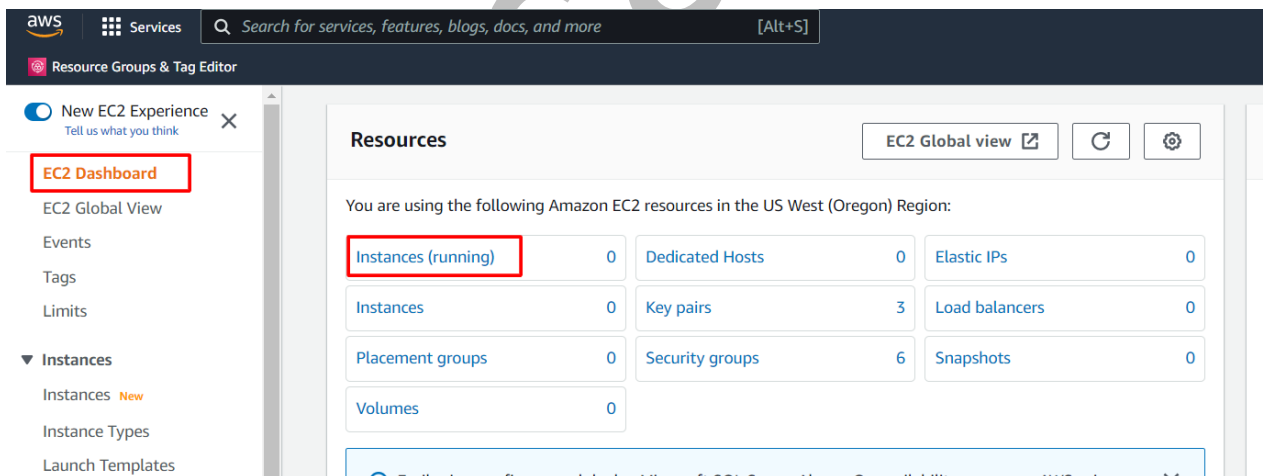
1. Open your **console**.



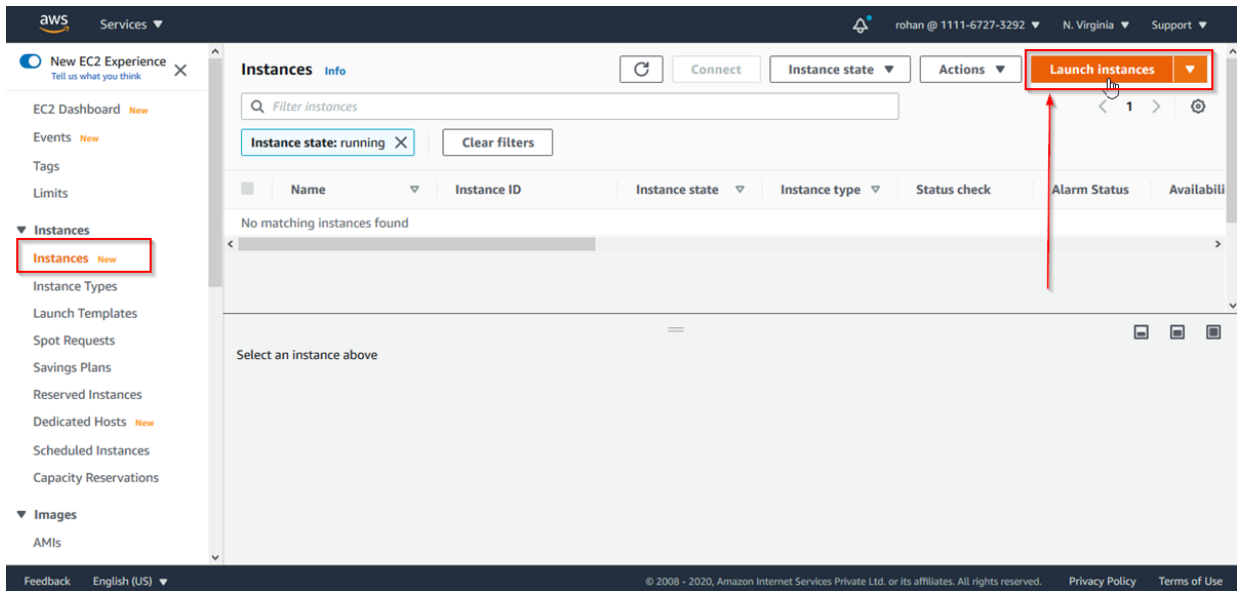
2. Search for **EC2** in the search bar and click on it.



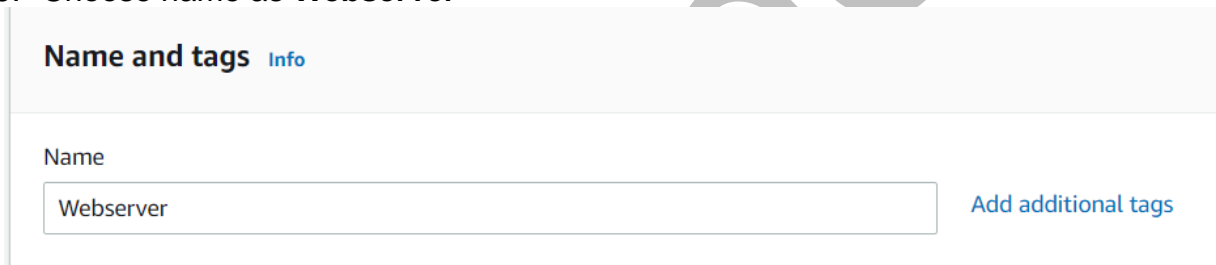
3. Click on **Instances (running)**.



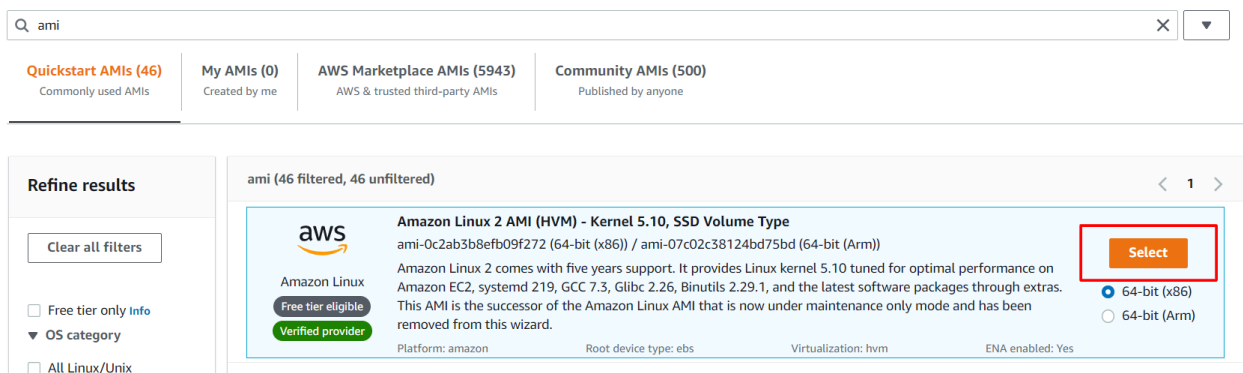
4. Click on **Launch Instances** on the right corner.



5. Choose name as **Webserver**



6. Choose **Amazon Linux 2 AMI**, select 64-bit (x86) architecture and Click on **Select**.



Note: Here, in the above step we are selecting the Amazon Linux 2 AMI which is basically preconfigured with some tools installed on it like Amazon Linux 2 AMI comes with AWS CLI pre-installed. AWS CLI is basically the Command Line Interface provided by AWS to access any of their services using CLI.

7. Choose an Instance type **t2.micro** as it is in free tier.

▼ Instance type [Info](#)

Instance type

t2.micro	Free tier eligible
Family: t2 1 vCPU 1 GiB Memory	▼
On-Demand Linux pricing: 0.0116 USD per Hour	
On-Demand Windows pricing: 0.0162 USD per Hour	

[Compare instance types](#)

Note: Here we must choose the instance type as **t2.micro**, because it is free tier eligible, i.e. you won't be charge for running the instance with this instance type as we need to perform the labs being in free tier. So, please make sure you select the t2.micro as the instance type else you shall be charged for running instances from Paid AMI's.

8. Select a new **keypair**.

▼ Key pair (login) [Info](#)

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

Key pair name - *required*

Select ▼

[Create new key pair](#)

Create key pair [X]

Key pairs allow you to connect to your instance securely.

Enter the name of the key pair below. When prompted, store the private key in a secure and accessible location on your computer. **You will need it later to connect to your instance.** [Learn more](#)

Key pair name
jenkins
The name can include upto 255 ASCII characters. It can't include leading or trailing spaces.

Key pair type
 RSA
RSA encrypted private and public key pair
 ED25519
ED25519 encrypted private and public key pair (Not supported for Windows instances)

Private key file format
 .pem
For use with OpenSSH
 .ppk
For use with PuTTY

Cancel [Create key pair] Launch inst...

- Now in networking, Click on **Create a new Security Group**, provide the security group a name and mention the same in description, then Click on **Add security Rule**.

▼ **Network settings** Info

VPC - required Info
vpc-027791e4cf8428650 (default) 172.31.0.0/16

Subnet Info
No preference Create new subnet

Auto-assign public IP Info
Enable

Firewall (security groups) Info
A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

Create security group Select existing security group

Security group name - required
k21-web-dmz
This security group will be added to all network interfaces. The name can't be edited after the security group is created. Max length is 255 characters. Valid characters: a-z, A-Z, 0-9, spaces, and _-/(/#+=&:[]\$*

Description - required Info
k21-web-dmz

Inbound security groups rules

▼ Security group rule 1 (TCP, 22, 0.0.0.0/0) Remove

Type <small>Info</small> ssh	Protocol <small>Info</small> TCP	Port range <small>Info</small> 22
Source type <small>Info</small> Anywhere	Source <small>Info</small> 0.0.0.0/0	Description - optional <small>Info</small> e.g. SSH for admin desktop

⚠ Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only. ×

Add security group rule

Note: In the above steps, we are creating and configuring rules for a security group. A security group is basically a virtual firewall for EC2 instances to control incoming and outgoing traffic to your instances.

10. Search for **HTTP** and press enter, port shall be **80**, source shall be **0.0.0.0/0 (IPv4), ::/0 (ipv6)** traffic
11. Click on **Add Security Rule**, Keep it Custom TCP & port no will be 8080.
12. Click on **Launch Instance**

The screenshot shows the AWS console interface for configuring security group rules and launching an instance. On the left, two security group rules are visible:

- Security group rule 2 (TCP, 80, Multiple sources):** Type: HTTP, Protocol: TCP, Port range: 80, Source type: Custom, Source: 0.0.0.0/0 and ::/0, Description: e.g. SSH for admin desktop.
- Security group rule 3 (TCP, 8080, Multiple sources):** Type: Custom TCP, Protocol: TCP, Port range: 8080, Source type: Custom, Source: 0.0.0.0/0 and ::/0, Description: e.g. SSH for admin desktop.

On the right, the instance configuration panel is shown with the following settings:

- Software Image (AMI):** Amazon Linux 2 AMI (HVM) - Ker...
- Virtual server type (instance type):** t2.micro
- Firewall (security group):** New security group
- Storage (volumes):** 1 volume(s) - 8 GiB

A red arrow points to the **Launch instance** button at the bottom right of the configuration panel.

Note: We add http rule here because we need to access our Webpage. As the **communication** for request and response between Webserver and Browser **happens on port 80** which is **http**.

Note: We have added 8080 port here because the **default port for Jenkins is 8080**

13. Now Click on **View All Instances**.

EC2 > Instances > Launch an instance

Success
Successfully initiated launch of instance (i-027ba060342207db8)

▶ Launch log

Next Steps

Get notified of estimated charges
Create [billing alerts](#) to get an email notification when estimated charges on your AWS bill exceed an amount you define (for example, if you exceed the free usage tier)

How to connect to your instance
Your instance is launching and it might be a few minutes until it is in the running state, when it will be ready for you to use
Click [View Instances](#) to monitor your instance's status. Once your instance is in the 'running' state, you can connect to it from the Instances screen. Find out [how to connect to your instance](#)

[View more resources to get you started](#)

View all Instances

14. Now Click on **View Instances**.

Launch Status

The following instance launches have been initiated. i-0003f420b02e90a0 view launch log

Get notified of estimated charges
Create [billing alerts](#) to get an email notification when estimated charges on your AWS bill exceed an amount you define (for example, if you exceed the free usage tier).

How to connect to your instances
Your instances are launching, and it may take a few minutes until they are in the **running** state, when they will be ready for you to use. Usage hours on your new instances will start immediately and continue to accrue until you stop or terminate your instances.
Click [View Instances](#) to monitor your instances' status. Once your instances are in the **running** state, you can [connect](#) to them from the Instances screen. [Find out](#) how to connect to your instances.

▼ Here are some helpful resources to get you started

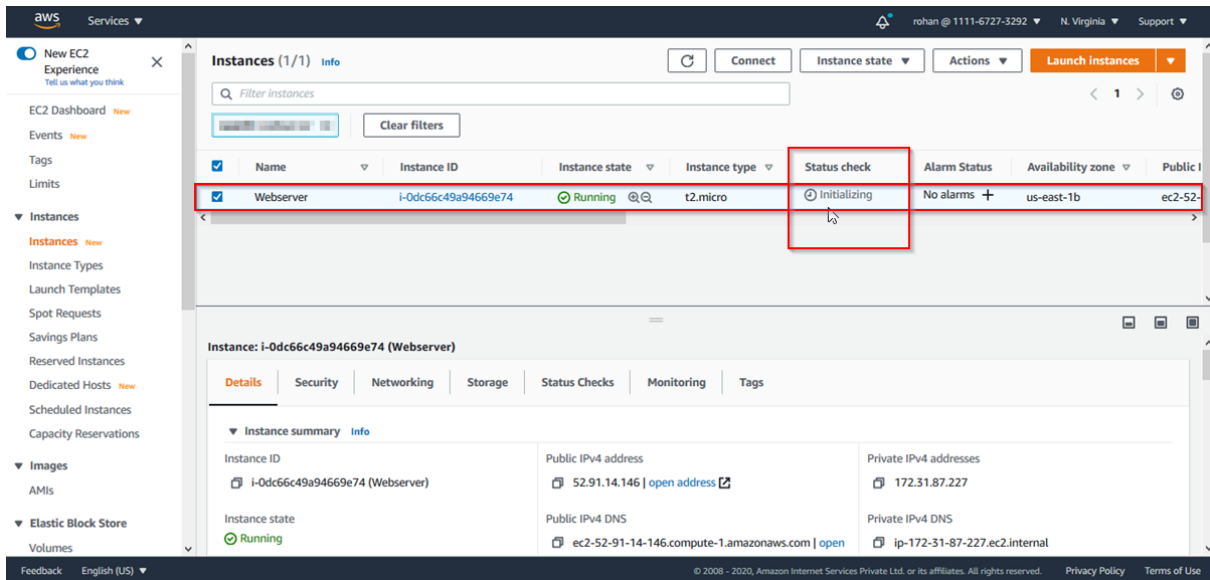
- How to connect to your Linux instance
- Learn about AWS Free Usage Tier
- Amazon EC2: User Guide
- Amazon EC2: Discussion Forum

While your instances are launching you can also

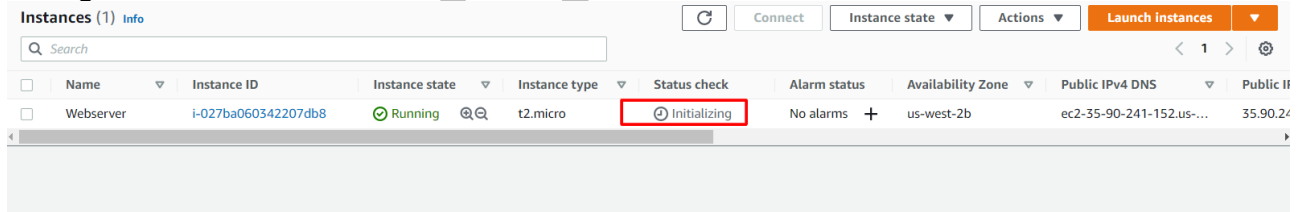
- Create [status check alarms](#) to be notified when these instances fail status checks. (Additional charges may apply)
- Create and attach [additional EBS volumes](#) (Additional charges may apply)
- Manage [security groups](#)

View Instances

15. Here, you shall see your instance is launching and the **Status check is Initializing**, wait for some time.



16. Refresh and you shall see your instance are Up and Running, and **Status check** has changed to **2/2 checks**.



Now, we have successfully created our first Linux Instance using EC2.

Note: Now, wait for the Status check of the instance to change to 2/2 checks pass as the instance is launching from an AMI and it needs time to install all the preconfigured things like AWS CLI and other tools. So, please wait till the Status check becomes 2/2 checks pass and then only proceed with further steps, and if status check has not passed then you shall get error in the further steps so please wait.

Note: Now we have launched the instance successfully, further we need to access the instance so, we have two ways for accessing an instance-

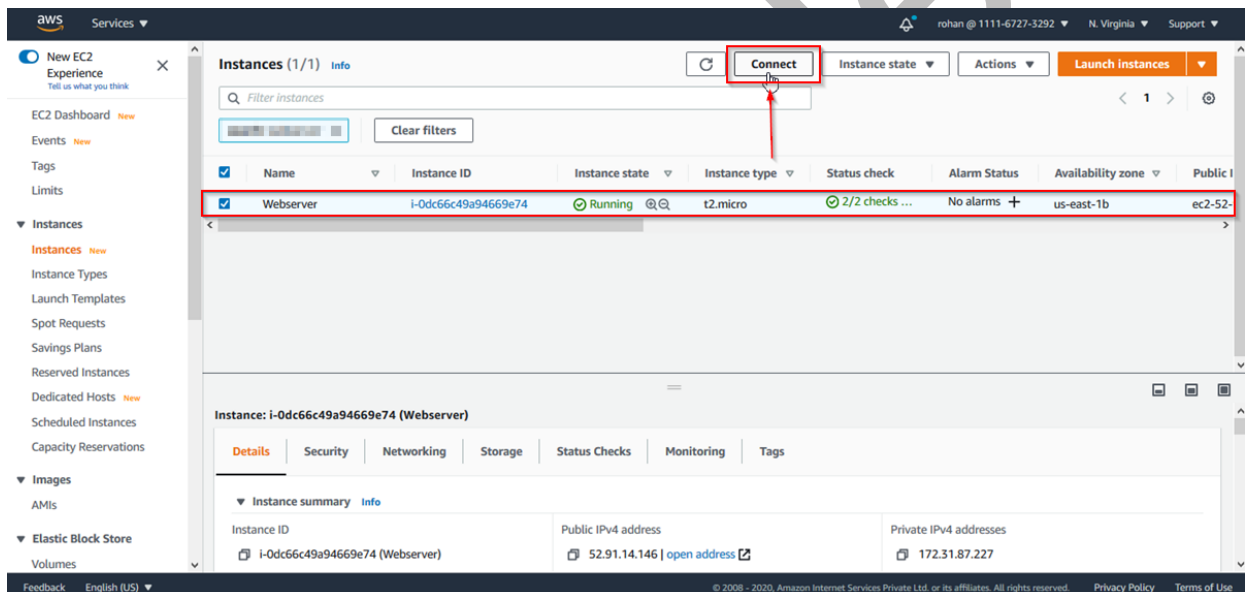
- a. Accessing via Browser using EC2 instance Connect.
- b. Accessing using Putty (a Secure Shell SSH client).

In further steps we shall see both the ways to access Linux EC2 instances.

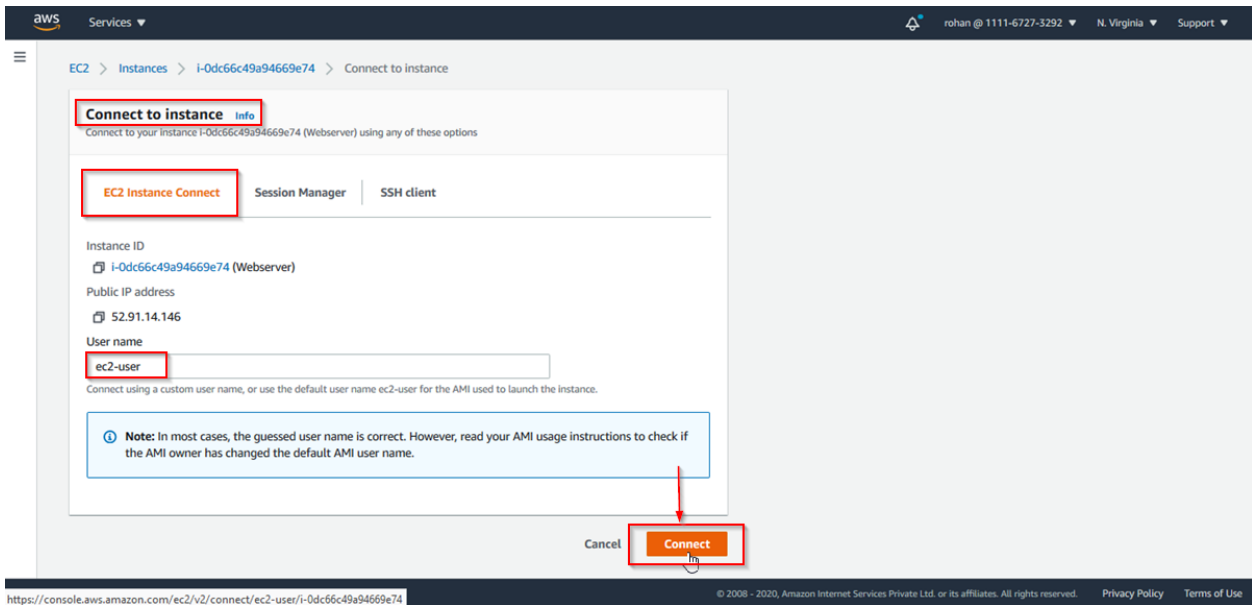
4.2 Accessing EC2 Instance Using Browser SSH Connection

Note: Till now, we have launched an instance successfully, but we need to access it to run commands so for accessing our instance the easy way is you can access the CLI directly from the browser using EC2 instance connect that we shall see in further steps.

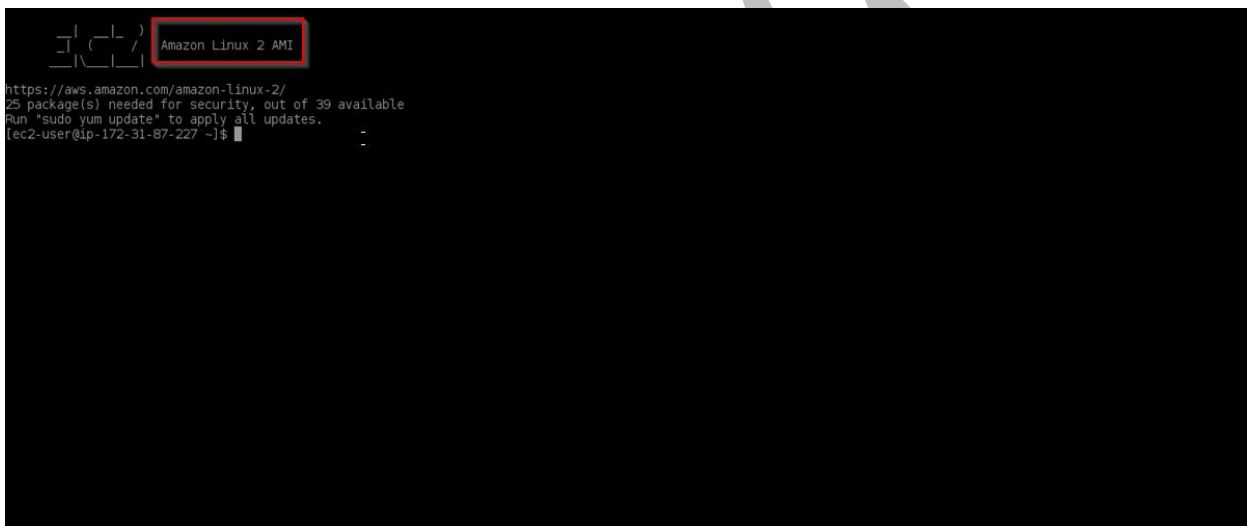
1. Select your Instance and Click on **Connect**.



2. Under EC2 Instance Connect, let everything default and Click on Connect.



3. Now you shall be redirected to new window and you shall see the Linux CLI.



i-0dc66c49a94669e74 (Webserver)
Public IPs: 52.91.14.146 Private IPs: 172.31.87.227

4.3 Accessing EC2 Instance Using Putty

Note: PuTTY is an SSH and telnet client. PuTTY is a free and open-source terminal emulator, serial console and network file transfer application. It supports several network protocols, including SCP, SSH, Telnet, rlogin, and raw socket connection. In this section, we will **download and install Putty** and will try to SSH (Secure Shell) into our instance using it.

1. Download and Install Putty from <http://www.chiark.greenend.org.uk/~sgtatham/putty/> Once you Click on the link you shall see this screen, Click on **Download it here**.

PuTTY: a free SSH and Telnet client

Home | [FAQ](#) | [Feedback](#) | [Licence](#) | [Updates](#) | [Mirrors](#) | [Keys](#) | [Links](#) | [Team](#)
Download: [Stable](#) · [Snapshot](#) | [Docs](#) | [Changes](#) | [Wishlist](#)

PuTTY is a free implementation of SSH and Telnet for Windows and Unix platforms, along with an xterm terminal emulator. It is written and maintained primarily by [Simon Tatham](#).

The latest version is 0.74. [Download it here](#).

LEGAL WARNING: Use of PuTTY, PSCP, PSFTP and Plink is illegal in countries where encryption is outlawed. We believe it is legal to use PuTTY, PSCP, PSFTP and Plink in England and Wales and in many other countries, but we are not lawyers, and so if in doubt you should seek legal advice before downloading it. You may find useful information at cryptolaw.org, which collects information on cryptography laws in many countries, but we can't vouch for its correctness.

Use of the Telnet-only binary (PuTTYtel) is unrestricted by any cryptography laws.

Latest news

2020-06-27 PuTTY 0.74 released

PuTTY 0.74, released today, is a bug-fix and security release. It fixes bugs in 0.73, including one possible vulnerability, and also adds a [new configuration option](#) to mitigate a minor information leak in SSH host key policy.

2019-09-29 PuTTY 0.73 released

PuTTY 0.73, released today, is a bug-fix release. It fixes a small number of bugs since 0.72, and a couple of them have potential security implications.

2019-07-20 PuTTY 0.72 released

PuTTY 0.72, released today, is a bug-fix release. It fixes a small number of further security issues found by the 2019 EU-funded HackerOne bug bounty, and a variety of other bugs introduced in 0.71.

2019-07-08 Bug bounty concluded

The EU-funded bug bounty programme is now closed. Many thanks to everybody who sent in reports!

2. Now Click on **64-bit** one and install it once downloaded.

Download PuTTY: latest release (0.74)

[Home](#) | [FAQ](#) | [Feedback](#) | [Licence](#) | [Updates](#) | [Mirrors](#) | [Keys](#) | [Links](#) | [Team](#)
Download: [Stable](#) · [Snapshot](#) | [Docs](#) | [Changes](#) | [Wishlist](#)

This page contains download links for the latest released version of PuTTY. Currently this is 0.74, released on 2020-06-27.

When new releases come out, this page will update to contain the latest, so this is a good page to bookmark or link to. Alternatively, here is a [permanent link to the 0.74 release](#).

Release versions of PuTTY are versions we think are reasonably likely to work well. However, they are often not the most up-to-date version of the code available. If you have a problem with this release, then it might be worth trying out the [development snapshots](#), to see if the problem has already been fixed in those versions.

Package files

You probably want one of these. They include versions of all the PuTTY utilities.
(Not sure whether you want the 32-bit or the 64-bit version? Read the [FAQ entry](#).)

MSI ('Windows Installer')

32-bit:	putty-0.74-installer.msi	(or by FTP)	(signature)
64-bit:	putty-64bit-0.74-installer.msi	(or by FTP)	(signature)

Unix source archive

.tar.gz:	putty-0.74.tar.gz	(or by FTP)	(signature)
----------	-----------------------------------	-----------------------------	-----------------------------

Alternative binary files

<https://the.earth.li/~sgtatham/putty/latest/w64/putty-64bit-0.74-installer.msi>

PuTTY will start downloading.

When new releases come out, this page will update to contain the latest, so th

Release versions of PuTTY are versions we think are reasonably likely to wo
this release, then it might be worth trying out the [development snapshots](#), to s

Package files

You probably want one of these. They include versions of all the PuT
(Not sure whether you want the 32-bit or the 64-bit version? Read the

MSI ('Windows Installer')

32-bit:	putty-0.74-installer.msi	(or by F
64-bit:	putty-64bit-0.74-installer.msi	(or by F

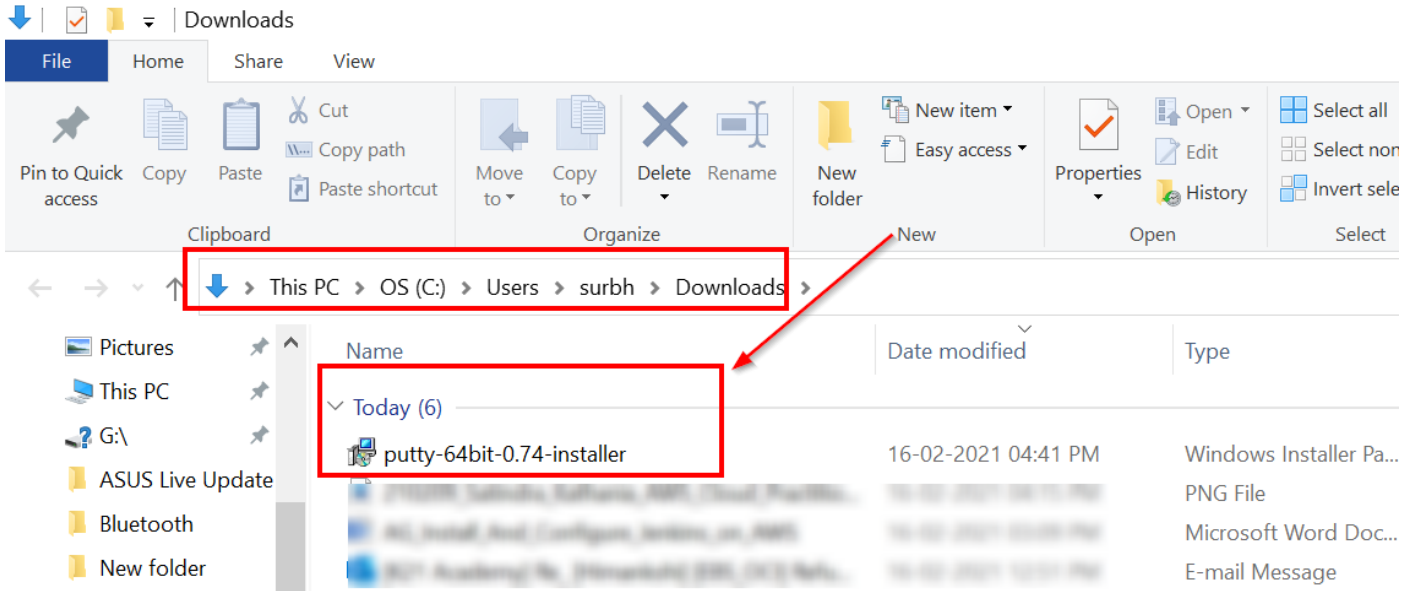
Unix source archive

	(or by F
--	--------------------------

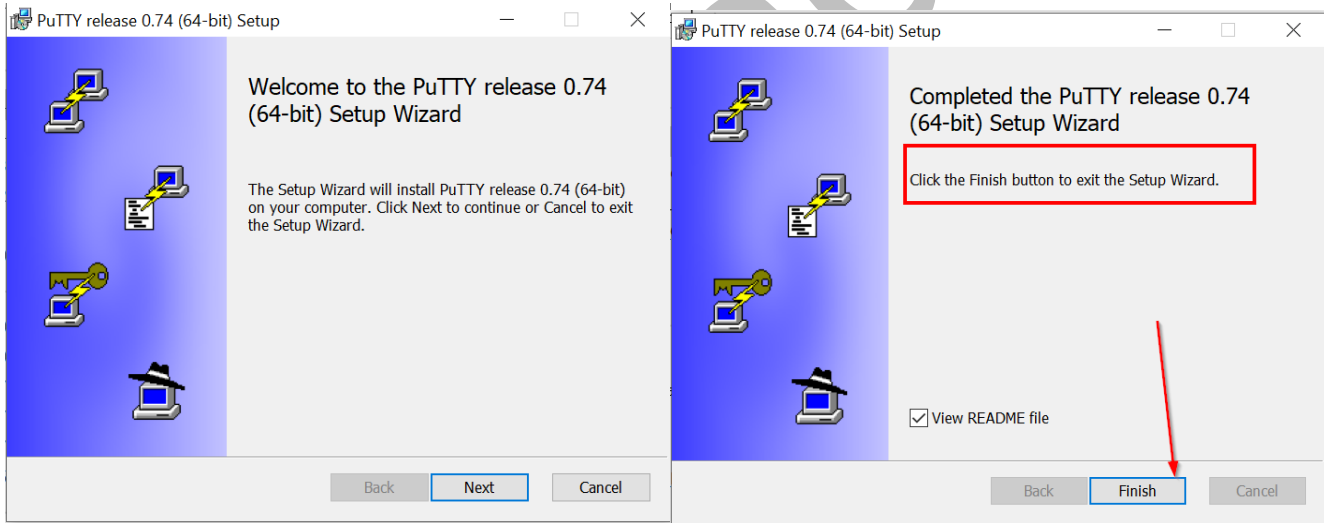
<https://the.earth.li/~sgtatham/putty/latest/w64/putty-64bit-0.74-installer.msi>

putty-64bit-0.74-i...msi
2.7/2.7 MB

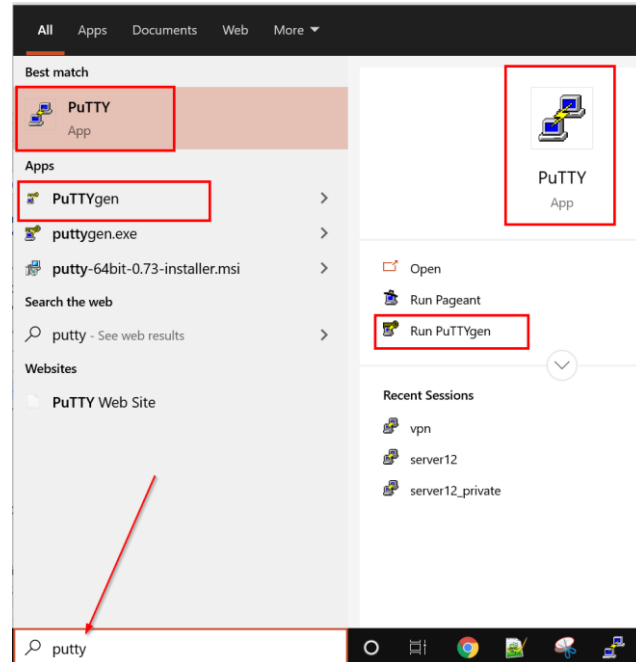
Once downloaded, go to the downloads window on the laptop & double click to install it.



It will open the setup wizard & then proceed to the next step, click **Next** to move ahead, complete all the steps to install & once done, click **Finish**



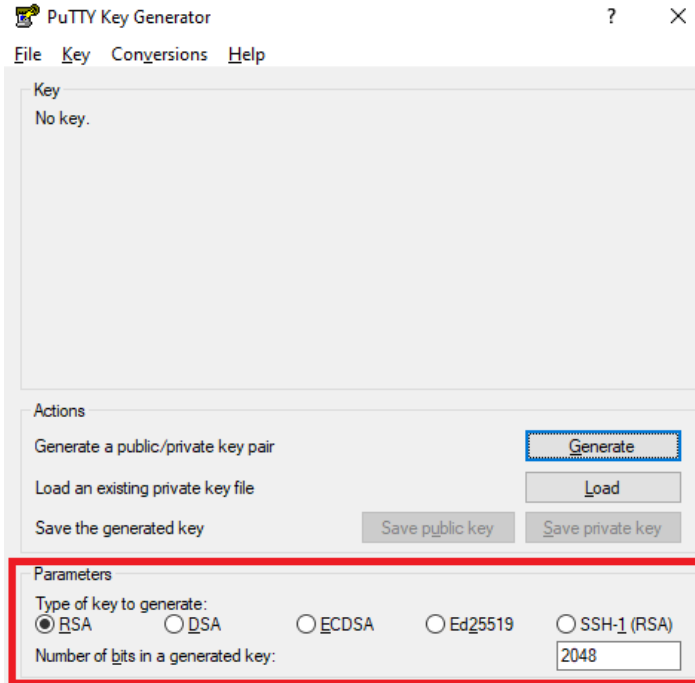
Note: When you download & install PuTTY, it will automatically install PuTTYgen too on your local laptop/PC



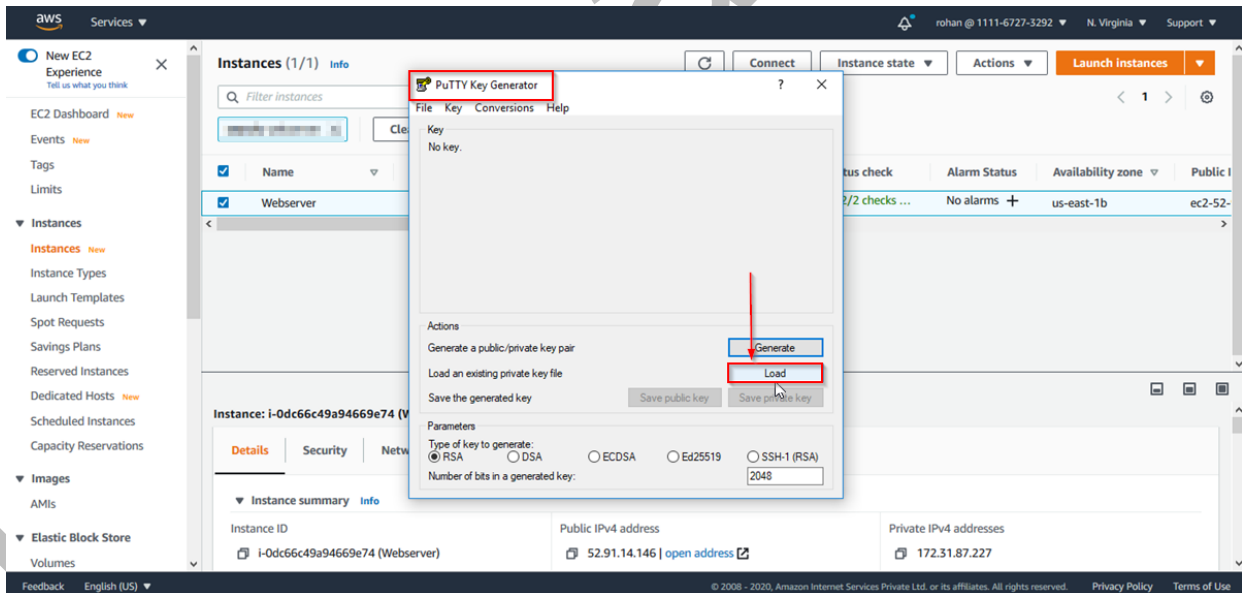
Note: Now, we have downloaded a key from the AWS console while creating Key-Pair, that key has an **.pem (privacy enhanced mail)** extension. However, putty reads key only with **.ppk (putty private key)** extension so need to convert our **.pem key** into **.ppk key** so that Putty can read and use the key to login in our instance.

For converting the key we need to load the Key in **PuTTYgen** which you can get when you search for it in the menu bar in your laptop, once the key is loaded we generate a key in **.ppk** extension using **PuttyGen**.

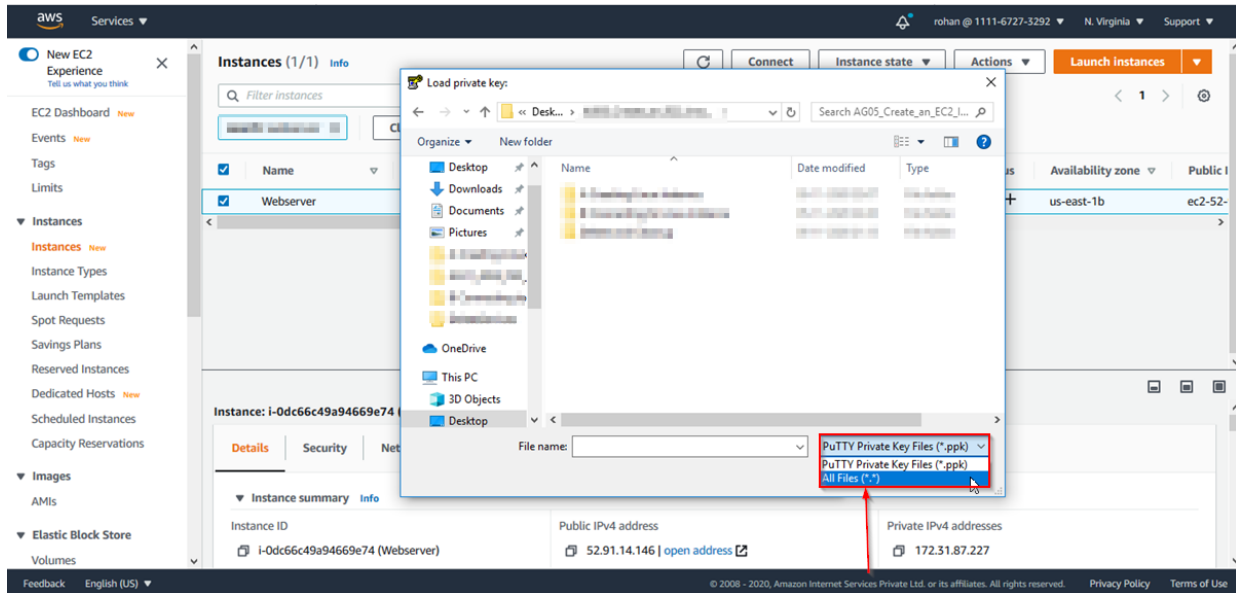
3. Once installed **Open PuTTYgen**, under Type of key to generate, make sure **RSA** is selected.



4. Now Click on **Load**.

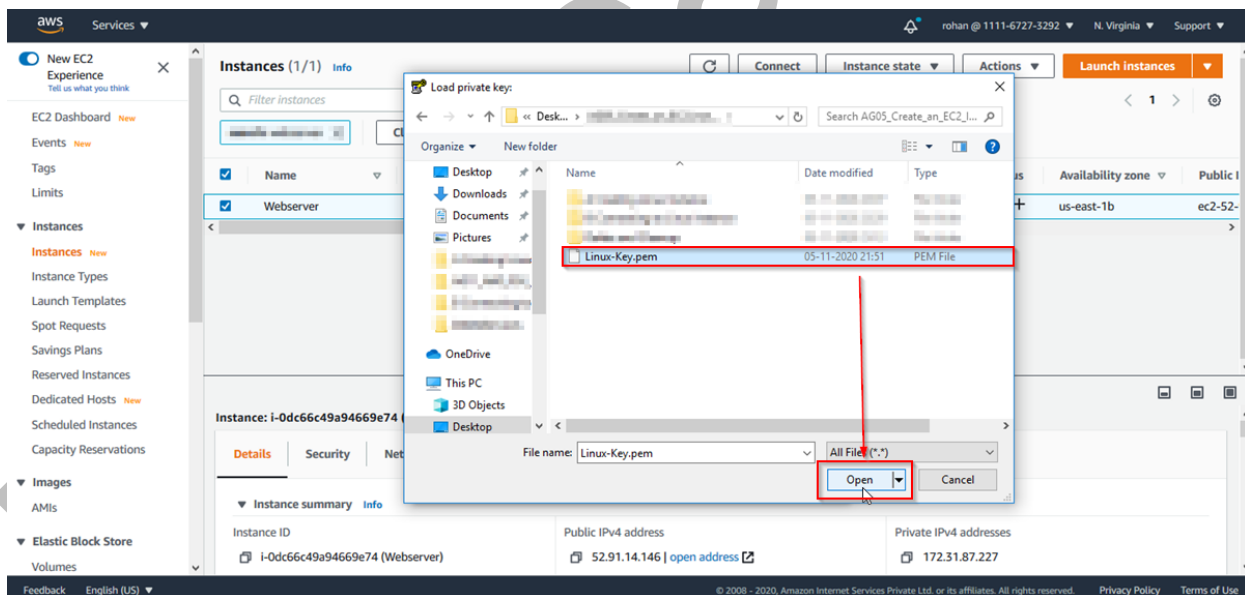


5. Click on **All Files** under the dropdown in the folder where your **.pem** file is saved.

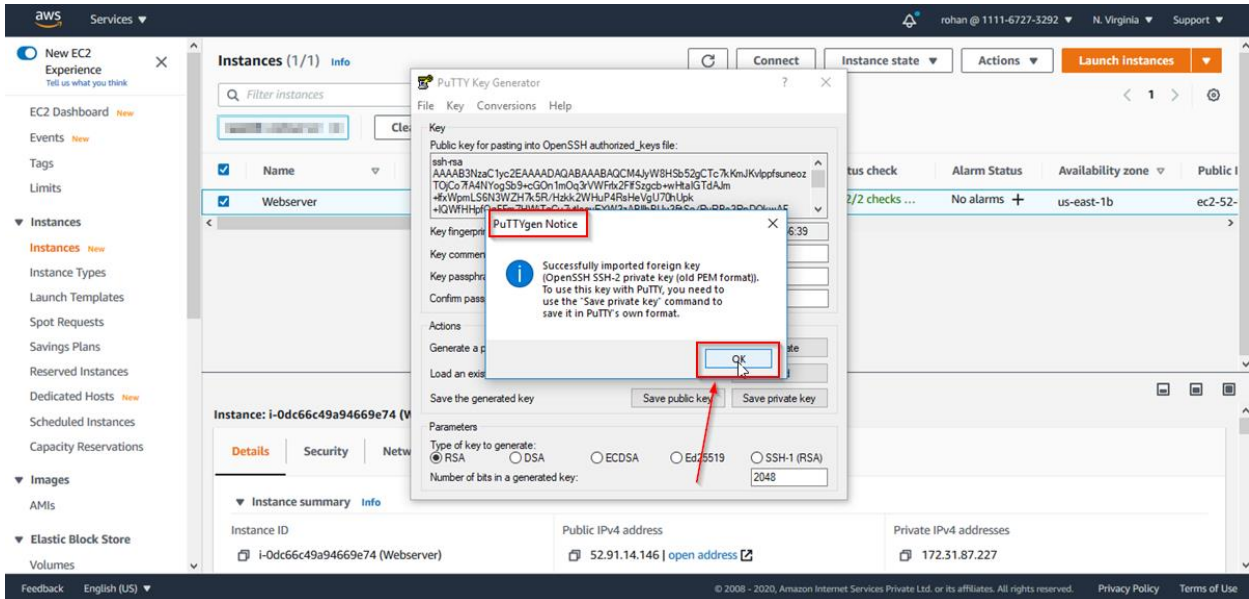


Note: If you don't select All files, you won't be able to see the .pem file. So please make sure you have selected All files from the Dropdown.

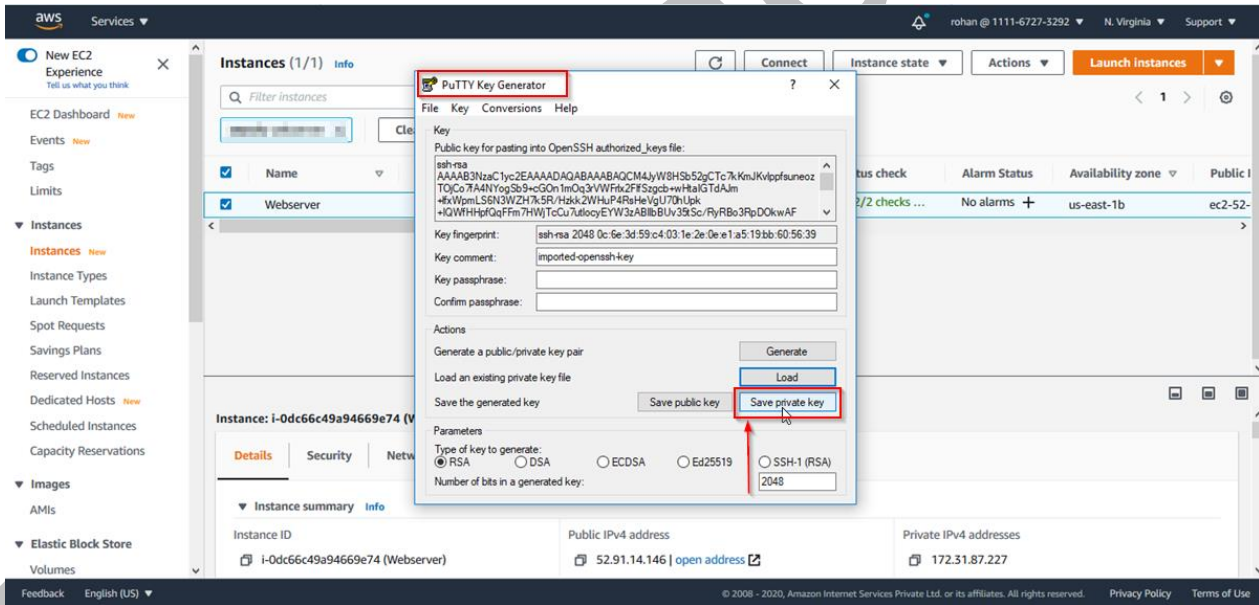
6. Select our .pem file and Click on **Open**.



7. Now Click on **OK**.

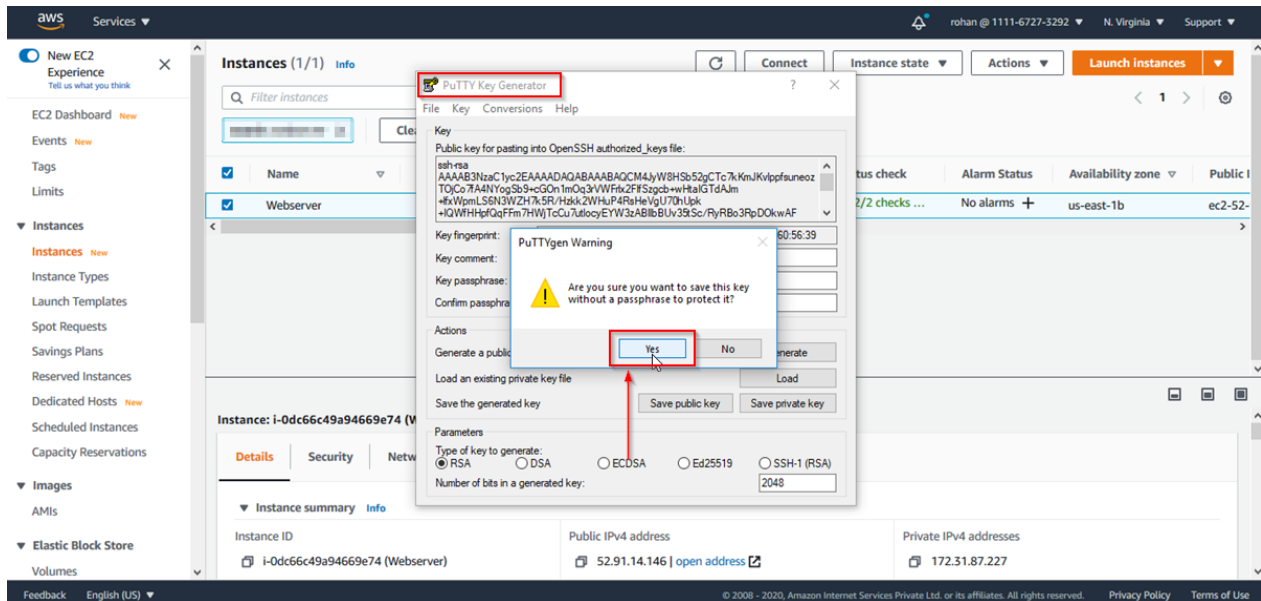


8. Click on **Save private key.**

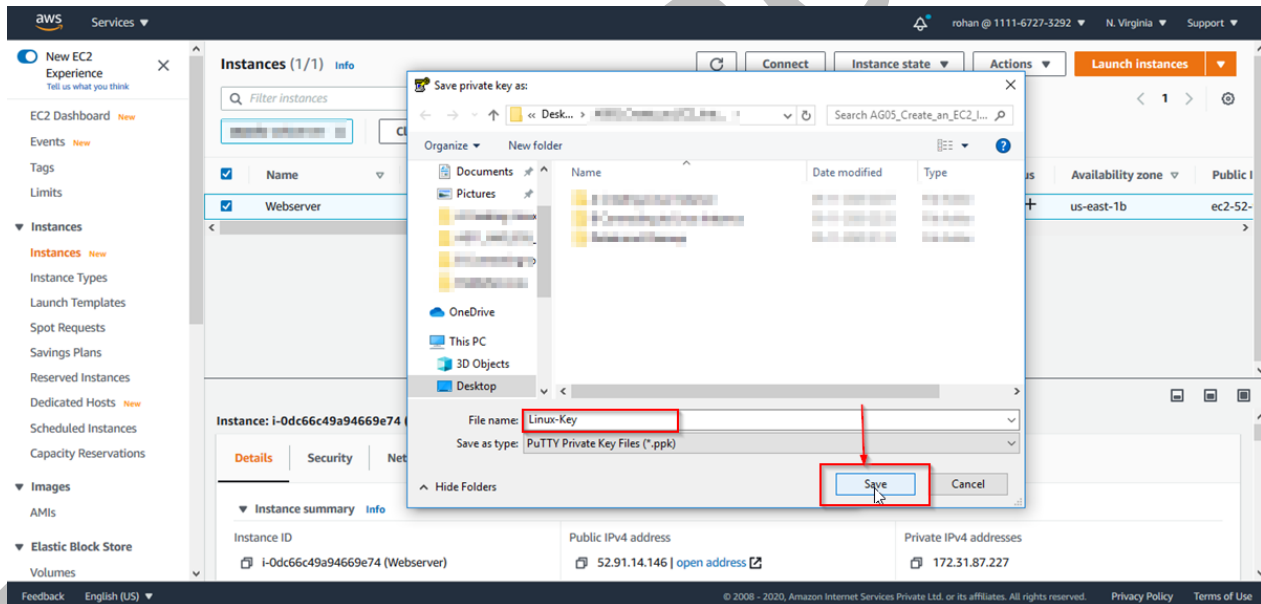


Note: Here when we select Save private key, we are basically saving the .ppk key that we require to login in our instance. **So, make sure you save it.**

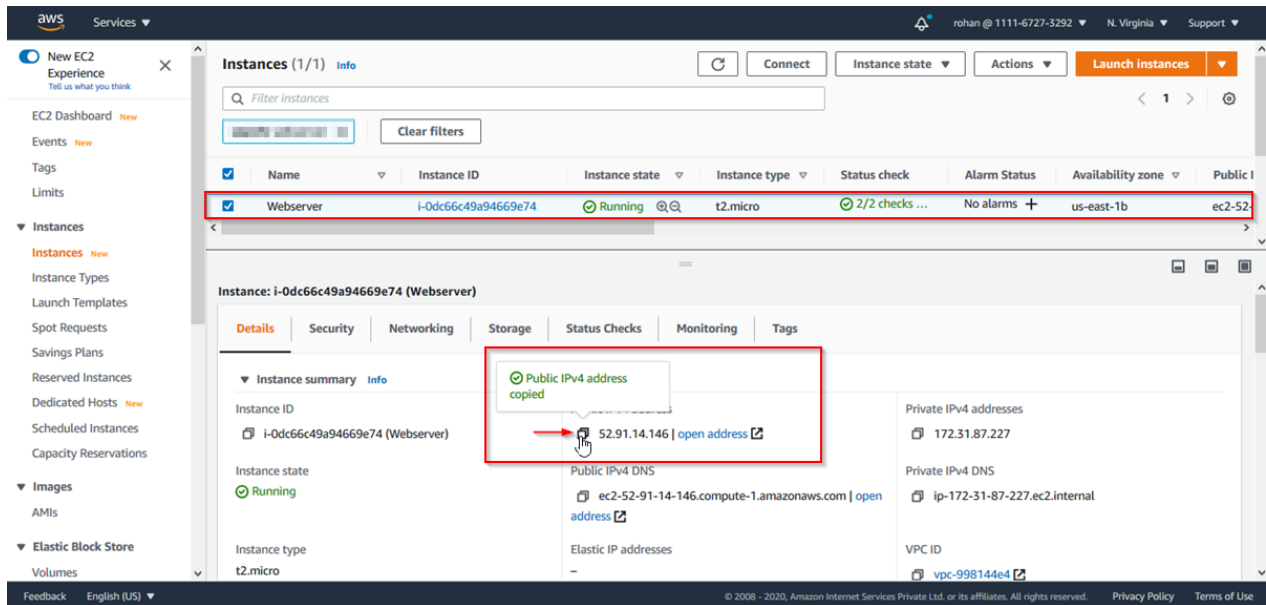
9. Click on **Yes.**



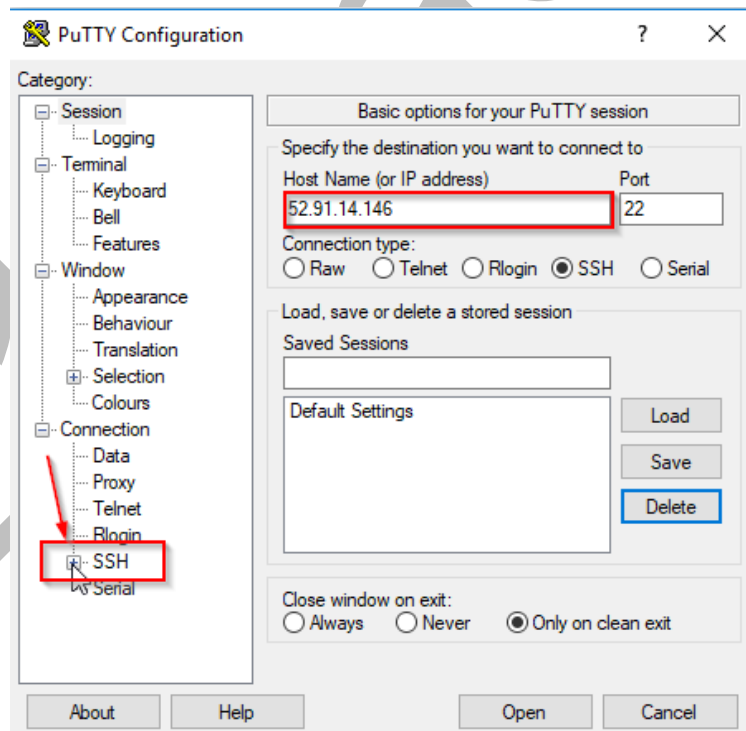
10. Enter **Linux-Key** as the name and Click on **Save**.



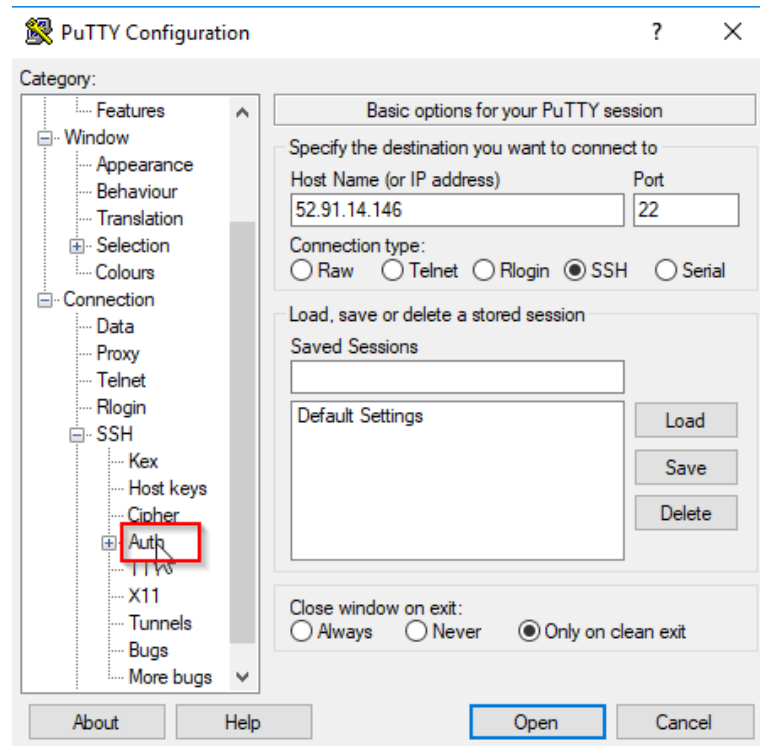
11. Now Select your instance and Copy the **public IPv4 address** under Details as shown below.



12. Now Open **Putty** and paste the Public-ip address under Host Name (or IP address) you copied and click on **plus mark on SSH**.

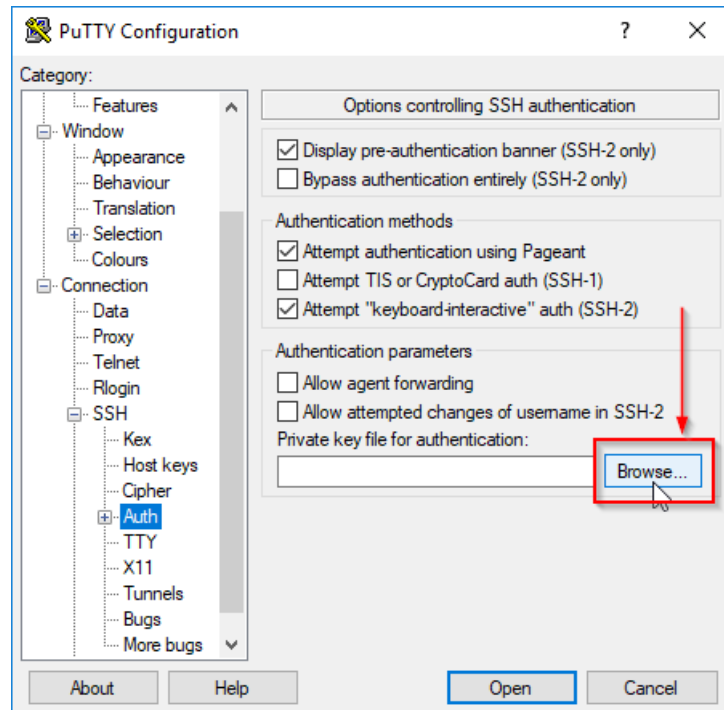


13. Under it now Click on **Auth**.

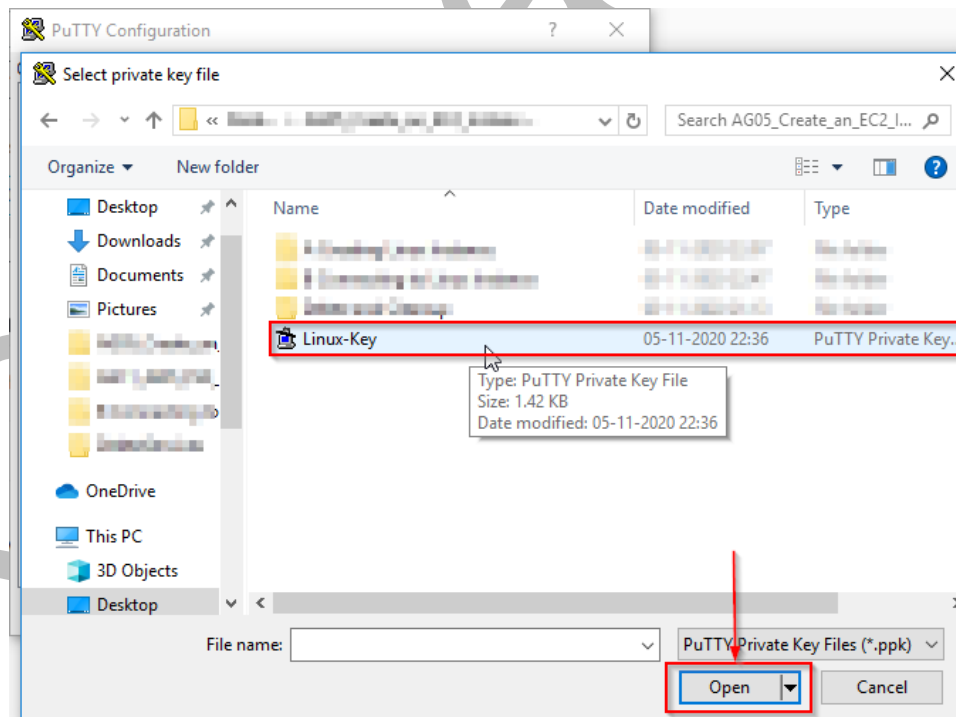


Note: Under Auth we shall provide the **.ppk key** that we generated earlier to authenticate us to login in our instance.

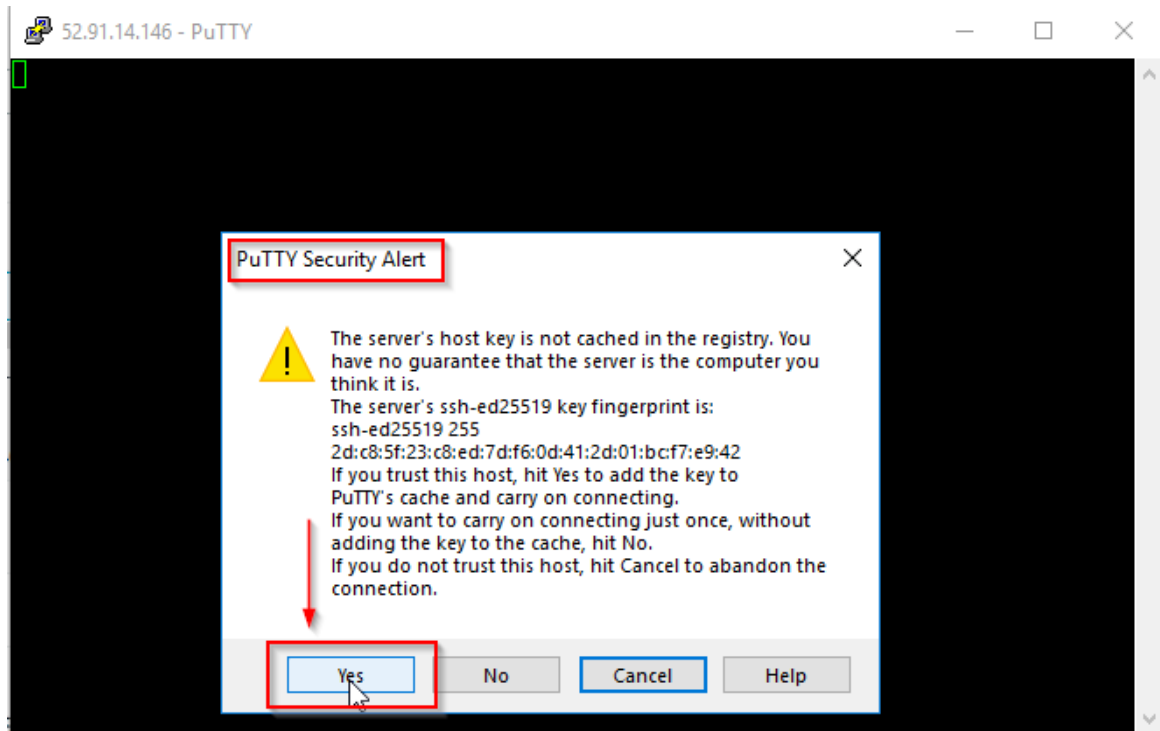
14. Now Click on **Browse**.



15. Now Select our **.ppk file** and Click on **Open**.



16. Now Click on **Yes**.

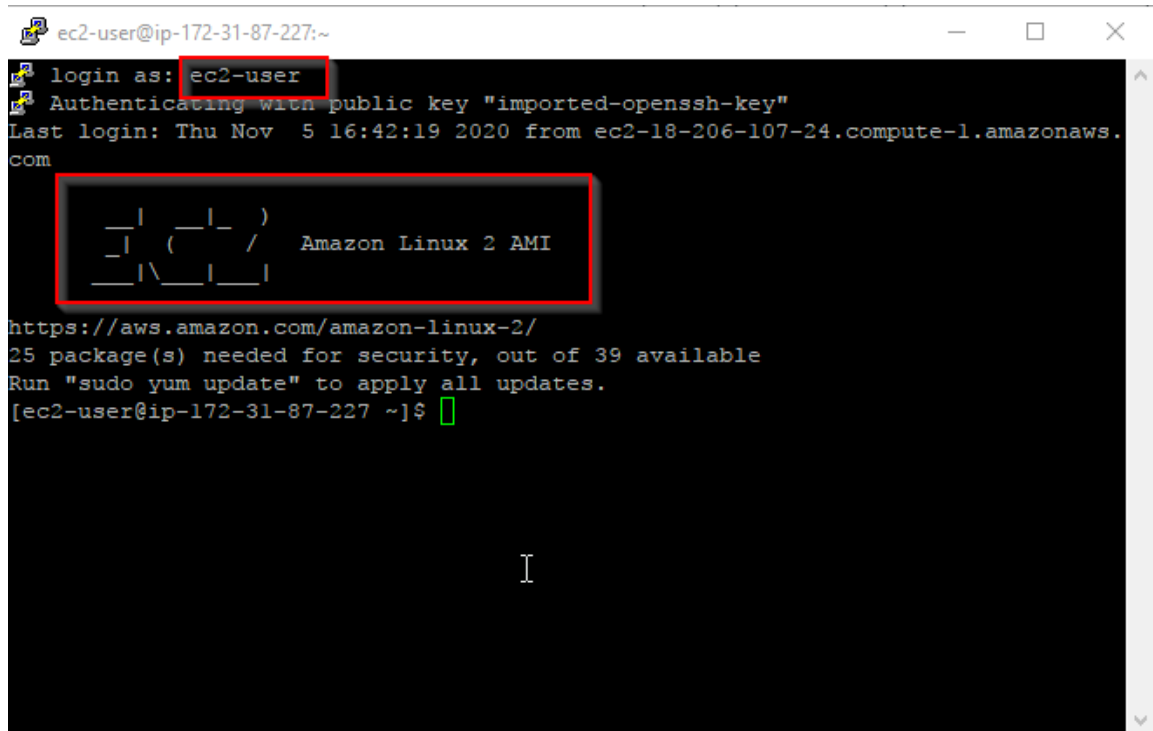


17. Now type **ec2-user** for login as and press Enter.



Note: Don't forget to type `ec2-user` to login as it is the default user by which you login into a Linux EC2 instance.

18. Now you have successfully connected to your Linux instance using putty.



```
ec2-user@ip-172-31-87-227:~  
login as: ec2-user  
Authenticating with public key "imported-openssh-key"  
Last login: Thu Nov  5 16:42:19 2020 from ec2-18-206-107-24.compute-1.amazonaws.com  
  
  _ | _ | _ )  
  _ | ( _ | /   Amazon Linux 2 AMI  
  _ | \ _ | _ |  
  
https://aws.amazon.com/amazon-linux-2/  
25 package(s) needed for security, out of 39 available  
Run "sudo yum update" to apply all updates.  
[ec2-user@ip-172-31-87-227 ~]$
```

5 INSTALL AND CONFIGURE JENKINS

In this step you will deploy Jenkins on your EC2 instance by completing the following tasks:

- Connect to Your Linux Instance
- Install JDK on the Linux Instance
- Download and Install Jenkins
- Configure Jenkins

5.1 Connect To Your Linux Instance

Note: Follow [Section 4.3](#) To connect to your Linux Instance using PUTTY.

5.2 Install JDK & Configure Jenkins

1. Now in new tab switch to the **root user**.

```
sudo -i
```

2. Update the software packages

```
sudo yum update -y
```

```
[ec2-user@jenkins ~]$ sudo -i
[root@jenkins ~]# sudo yum update -y
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
amzn2-core | 3.7 kB 00:00
No Match for argument: -y
No packages marked for update
[root@jenkins ~]#
```

3. Add the Jenkins repo using the following command:

```
sudo wget -O /etc/yum.repos.d/jenkins.repo \
https://pkg.jenkins.io/redhat-stable/jenkins.repo
```

4. Import a key file from Jenkins-CI to enable installation from the package

```
sudo rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io.key
```

```
[root@jenkins ~]# sudo wget -O /etc/yum.repos.d/jenkins.repo \
> https://pkg.jenkins.io/redhat-stable/jenkins.repo
--2022-09-20 07:38:02-- https://pkg.jenkins.io/redhat-stable/jenkins.repo
Resolving pkg.jenkins.io (pkg.jenkins.io)... 146.75.34.133, 2a04:4e42:78::645
Connecting to pkg.jenkins.io (pkg.jenkins.io)|146.75.34.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 85
Saving to: '/etc/yum.repos.d/jenkins.repo'
100%[=====>] 85 ---K/s in 0s
2022-09-20 07:38:02 (5.73 MB/s) - '/etc/yum.repos.d/jenkins.repo' saved [85/85]
[root@jenkins ~]# sudo rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io.key
[root@jenkins ~]#
```

5. Upgrade the Software Packages

```
sudo yum upgrade
```

```
[root@jenkins ~]# sudo yum upgrade
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
amzn2-core
jenkins
jenkins/primary_db
Resolving Dependencies
--> Running transaction check
--> Package amazon-ssm-agent.x86_64 0:3.1.1575.0-1.amzn2 will be updated
--> Package amazon-ssm-agent.x86_64 0:3.1.1732.0-1.amzn2 will be an update
--> Package chrony.x86_64 0:4.0-3.amzn2.0.2 will be updated
--> Package chrony.x86_64 0:4.2-5.amzn2.0.2 will be an update
--> Package dhclient.x86_64 12:4.2.5-77.amzn2.1.6 will be updated
--> Package dhclient.x86_64 12:4.2.5-79.amzn2.1.1 will be an update
--> Package dhcp-common.x86_64 12:4.2.5-77.amzn2.1.6 will be updated
--> Package dhcp-common.x86_64 12:4.2.5-79.amzn2.1.1 will be an update
--> Package dhcp-libs.x86_64 12:4.2.5-77.amzn2.1.6 will be updated
--> Package dhcp-libs.x86_64 12:4.2.5-79.amzn2.1.1 will be an update
--> Package ec2-net-utils.noarch 0:1.7.0-1.amzn2 will be updated
--> Package ec2-net-utils.noarch 0:1.7.1-1.amzn2 will be an update
--> Package gnupg2.x86_64 0:2.0.22-5.amzn2.0.4 will be updated
--> Package gnupg2.x86_64 0:2.0.22-5.amzn2.0.5 will be an update
--> Package kernel.x86_64 0:5.10.135-122.509.amzn2 will be installed
--> Package kernel.x86_64 0:5.10.135-122.509.amzn2 will be updated
```

6. Install **Jenkins** on which we are configuring as Jenkins server and start Jenkins service

```
sudo amazon-linux-extras install java-openjdk11 -y
```

```
Verifying : python-javapackages-3.4.1-11.amzn2.noarch
Verifying : libxi-1.7.9-1.amzn2.0.2.x86_64
Verifying : libICE-1.0.9-9.amzn2.0.2.x86_64
Verifying : libfontenc-1.1.3-3.amzn2.0.2.x86_64
Verifying : tzdata-java-2022c-1.amzn2.noarch
Verifying : 1:java-11-openjdk-11.0.16.0.8-1.amzn2.0.1.x86_64
Verifying : libXtst-1.2.3-1.amzn2.0.2.x86_64
Verifying : copy-jdk-configs-3.3-10.amzn2.noarch
Verifying : alsa-lib-1.1.4.1-2.amzn2.x86_64
Verifying : 1:java-11-openjdk-headless-11.0.16.0.8-1.amzn2.0.1.x86_64
Verifying : javapackages-tools-3.4.1-11.amzn2.noarch

Installed:
java-11-openjdk.x86_64 1:11.0.16.0.8-1.amzn2.0.1
```

7. Check for Java version

```
java -version
```

8. Now, install Jenkins.

```
sudo yum install jenkins -y
```

```
[root@jenkins ~]# sudo yum install jenkins -y
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd,
amzn2-core
Resolving Dependencies
--> Running transaction check
--> Package jenkins.noarch 0:2.361.1-1.1 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

=====
Package Arch Version Repository
=====
Installing:
jenkins noarch 2.361.1-1.1 jenkins
Transaction Summary
-----
Install 1 Package

Total download size: 89 M
Installed size: 89 M
Downloading packages:
jenkins-2.361.1-1.1.noarch.rpm
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : jenkins-2.361.1-1.1.noarch
  Verifying : jenkins-2.361.1-1.1.noarch

Installed:
jenkins.noarch 0:2.361.1-1.1

Complete!
[root@jenkins ~]#
```

9. Now, Enable & Start Jenkins

```
service jenkins start
service jenkins enable
```

```
[root@jenkins ~]# service jenkins start
Starting jenkins (via systemctl): [ OK ]
[root@jenkins ~]# service jenkins enable
Usage: /etc/init.d/jenkins {start|stop|status|try-restart|restart|force-reload|reload|probe}
[root@jenkins ~]#
```

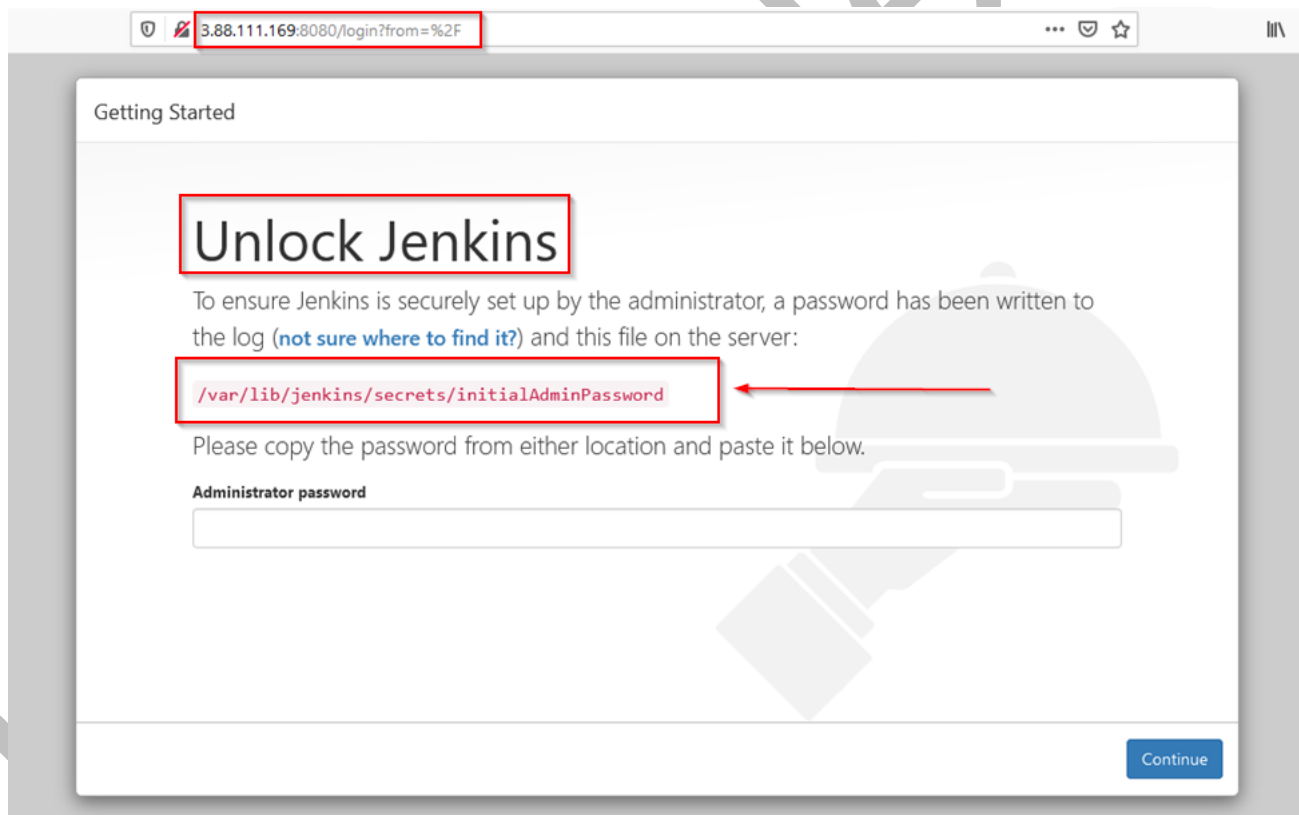
Thus, we have successfully installed Jenkins, in further steps we will configure Jenkins for username and password.

5.3 Configure Jenkins

1. is now installed and running on your EC2 instance.
2. To configure **Jenkins**:
3. Connect to http://<your_server_public_DNS>:8080 from your favorite browser. You will be able to access Jenkins through its management interface:
4. Open any Browser on your laptop/PC, In my case I will paste the below on the search bar <http://3.88.111.169:8080/> (you use the IP of your instance)
5. You will see, a web page will get displayed, it is Jenkins Interface.

Note: take public IP of the machine and paste in browser

<Public Ip of the machine>:8080



6. Get the **initialAdminPassword** using following command and **Copy** the **password** to start the Jenkins server setup.

```
sudo cat /var/lib/jenkins/secrets/initialAdminPassword
```

```
[root@jenkins ~]# sudo cat /var/lib/jenkins/secrets/initialAdminPassword  
dfa2b88a5dd3411f8ccbdb5d2c715bac  
[root@jenkins ~]#
```

7. **Paste** the **unlock key/password** under **Administrator password** and click on **Continue**.

Getting Started

Unlock Jenkins

To ensure Jenkins is securely set up by the administrator, a password has been written to the log ([not sure where to find it?](#)) and this file on the server:

```
/var/lib/jenkins/secrets/initialAdminPassword
```

Please copy the password from either location and paste it below.

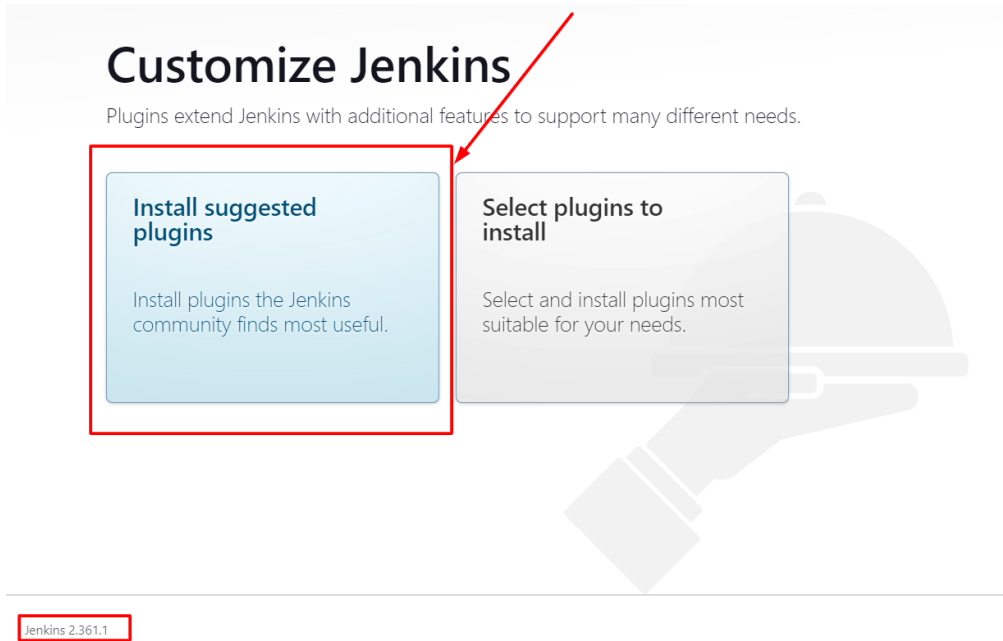
Administrator password

.....

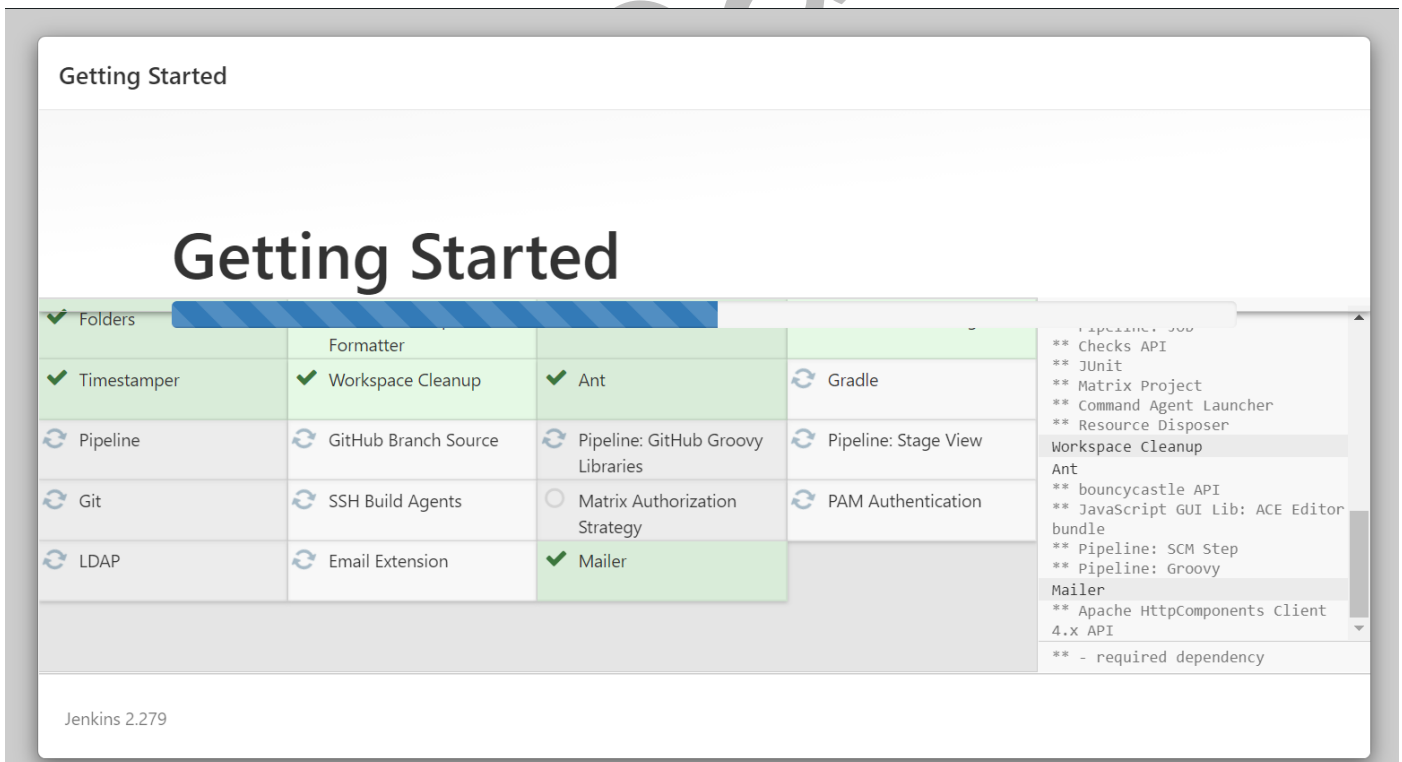
 This connection is not secure. Logins entered here could be compromised. [Learn More](#)

Continue

8. Now, click on **Install suggested plugins**.



8. Click Install suggested plugins.



9. Once the installation is complete, enter Administrator Credentials, click Save Credentials.

OR

You can click **Skip & Continue as Admin**

Getting Started

Create First Admin User

Username:

Password:

Confirm password:

Full name:

E-mail address:

Jenkins 2.279

1

Skip and continue as admin

2

Save and Continue

10. And then click Save and Finish.

K21A

Getting Started

Instance Configuration

Jenkins URL:

The Jenkins URL is used to provide the root URL for absolute links to various Jenkins resources. That means this value is required for proper operation of many Jenkins features including email notifications, PR status updates, and the BUILD_URL environment variable provided to build steps.

The proposed default value shown is **not saved yet** and is generated from the current request, if possible. The best practice is to set this value to the URL that users are expected to use. This will avoid confusion when sharing or viewing links.

Jenkins 2.279

Not now

Save and Finish

11. Click Start Using Jenkins

Getting Started

Jenkins is ready!

You have skipped the **setup of an admin user**.

To log in, use the username: "admin" and the administrator password you used to access the setup wizard.

Your Jenkins setup is complete.

Start using Jenkins

Jenkins search admin log out

Dashboard

- New Item
- People
- Build History
- Manage Jenkins
- My Views
- Lockable Resources
- New View

Build Queue ^
No builds in the queue.

Build Executor Status ^

Welcome to Jenkins!

This page is where your Jenkins jobs will be displayed. To get started, you can set up distributed builds or start building a software project.

Start building your software project

Create a job →

Set up a distributed build

Set up an agent →

Configure a cloud →

Learn more about distributed builds ↗

add description

Jenkins search admin log out

Dashboard

- New Item
- People
- Build History
- Manage Jenkins**
- My Views
- Lockable Resources
- New View

Build Queue ^
No builds in the queue.

Build Executor Status ^

Manage Jenkins

System Configuration

- Configure System**
Configure global settings and paths.
- Global Tool Configuration**
Configure tools, their locations and automatic installers.
- Manage Plugins**
Add, remove, disable or enable plugins that can extend the functionality of Jenkins.
- Manage Nodes and Clouds**
Add, remove, control and monitor the various nodes that Jenkins runs jobs on.

Security

- Configure Global Security**
Secure Jenkins; define who is allowed to access/use the system.
- Manage Credentials**
Configure credentials
- Configure Credential Providers**
Configure the credential providers

6 USEFUL COMMANDS (START/STOP, UPDATE, UNINSTALL)

- To stop Jenkins

```
sudo service jenkins stop
```

- To uninstall Jenkins

```
sudo service jenkins stop  
sudo yum remove jenkins  
sudo rm -r /var/lib/jenkins
```

- To start jenkins on a different port

```
Update port number in /etc/sysconfig/jenkins
```

- To fetch initial admin password

```
sudo su -  
cd /var/lib/jenkins/secrets/  
cat initialAdminPassword
```

7 DELETING/CLEANUP

Note: In this section we will be deleting all the resources we have used for this lab in order to be in free tier limit, as keeping these resources running will be chargeable.

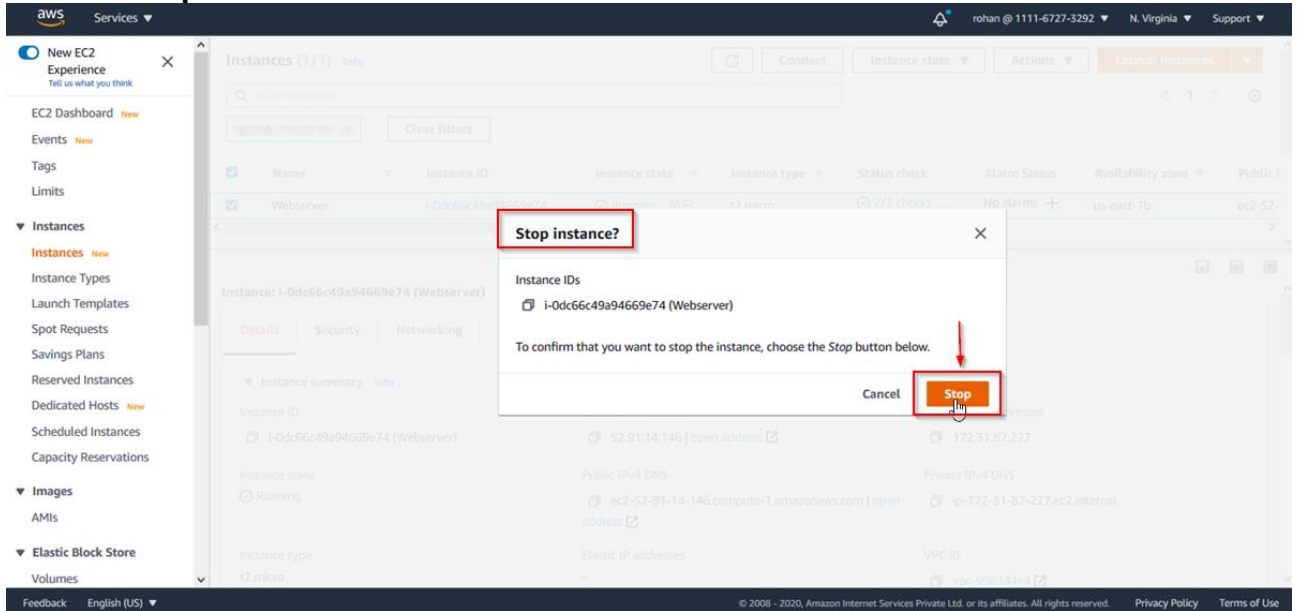
7.1 Stopping the Linux Instance

Note: We would always recommend to Terminate the instance if there is no requirement of it. But many times, it may happen that you shall require the instance after some time to continue with your work, then you shall stop the instance. Stopping the instance will not charge you but you shall be charged for the volume/storage attached to it. So, the charges shall be minimum as compared to running instance, but you will still get charged.

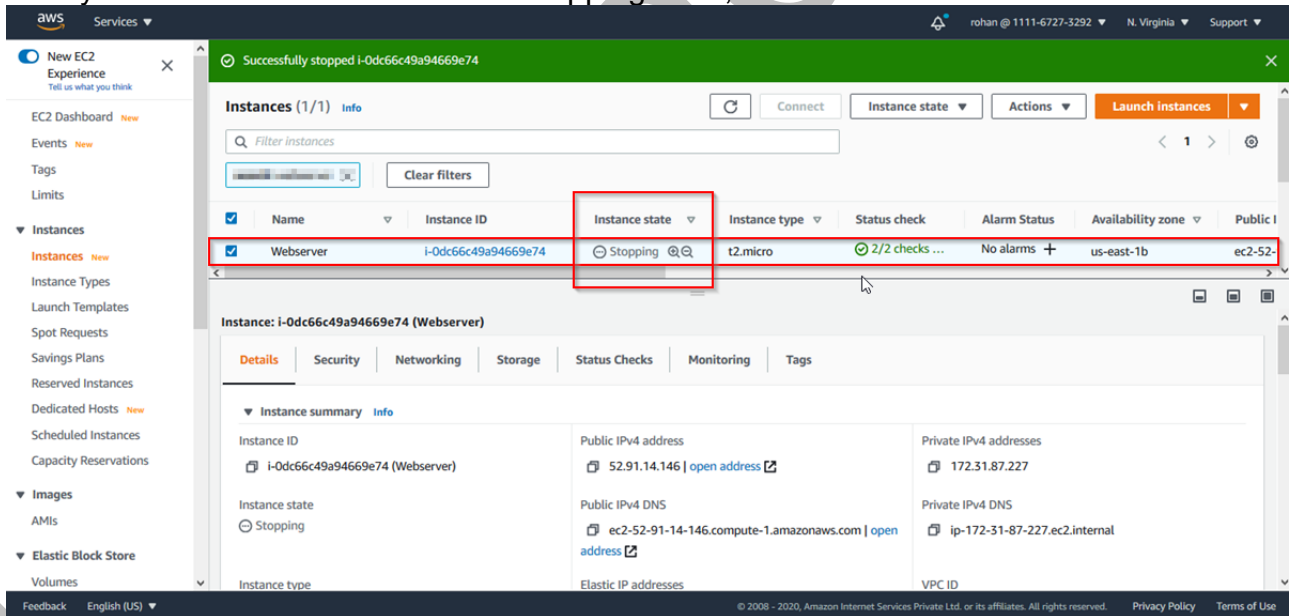
1. Back on EC2 Console, select our Instance, under **instance state** Click on **Stop instance**.

The screenshot shows the AWS Management Console interface for EC2 Instances. The 'Instances (1/1)' page is active, displaying a table with one instance: 'Webserver' (Instance ID: i-0dc66c49a94669e74) in a 'Running' state. A red box highlights the 'Instance state' dropdown menu, and a red arrow points to the 'Stop instance' option. The instance row is highlighted in blue. Below the table, the details for the selected instance are visible, including its Instance ID, Public IPv4 address (52.91.14.146), and Instance type (t2.micro).

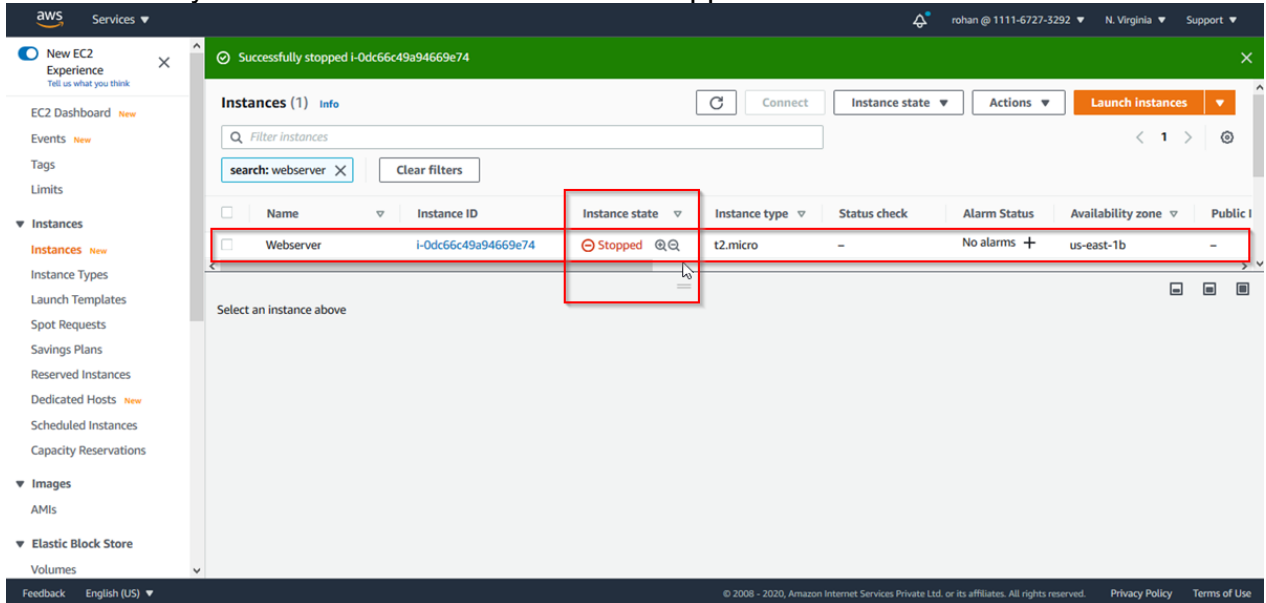
2. Click on **Stop** to Confirm.



3. Now you shall see our instance is in stopping state, wait for some time.



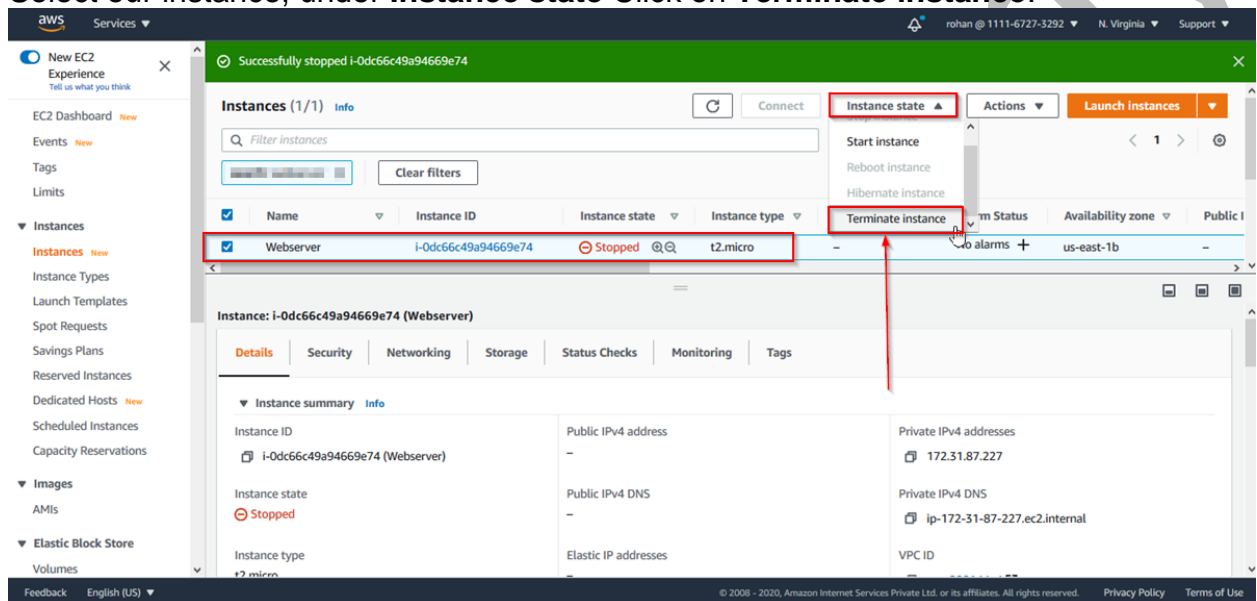
4. Refresh and you shall see our instance have stopped.



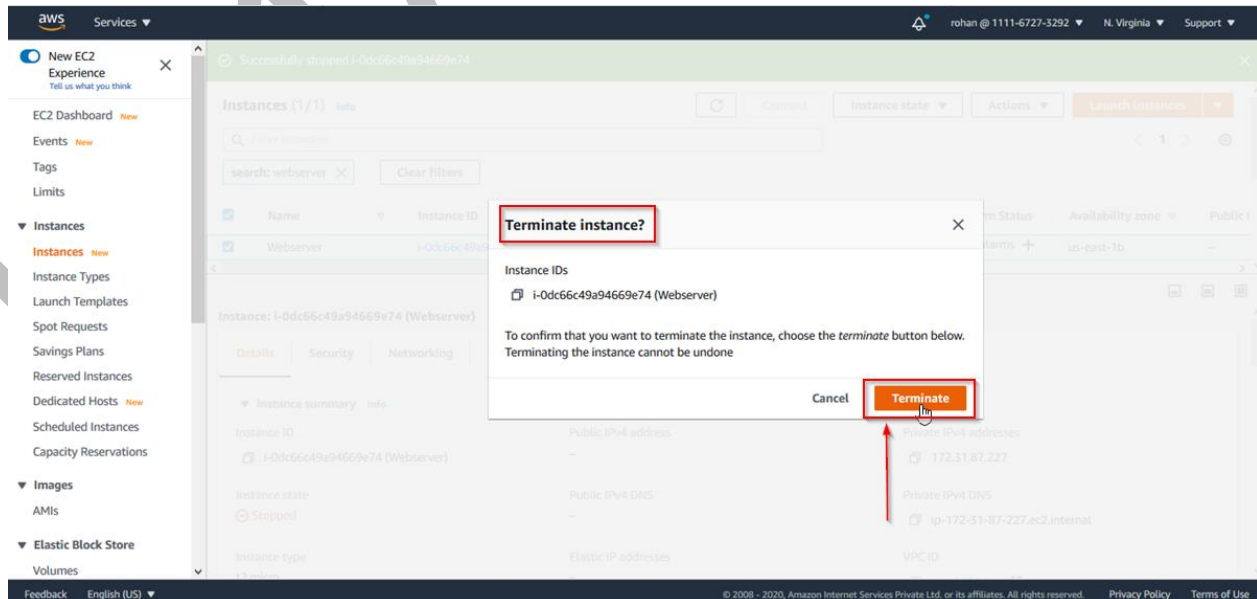
7.2 Terminating the Linux Instance

Note: In this section, we shall see the steps to Terminate the EC2 instance, as if the instances are not terminated once there is no requirement of it then you shall be charged for using those resources. So, to be in free tier and complete all the labs please terminate the resources here the EC2 instance when in no use.

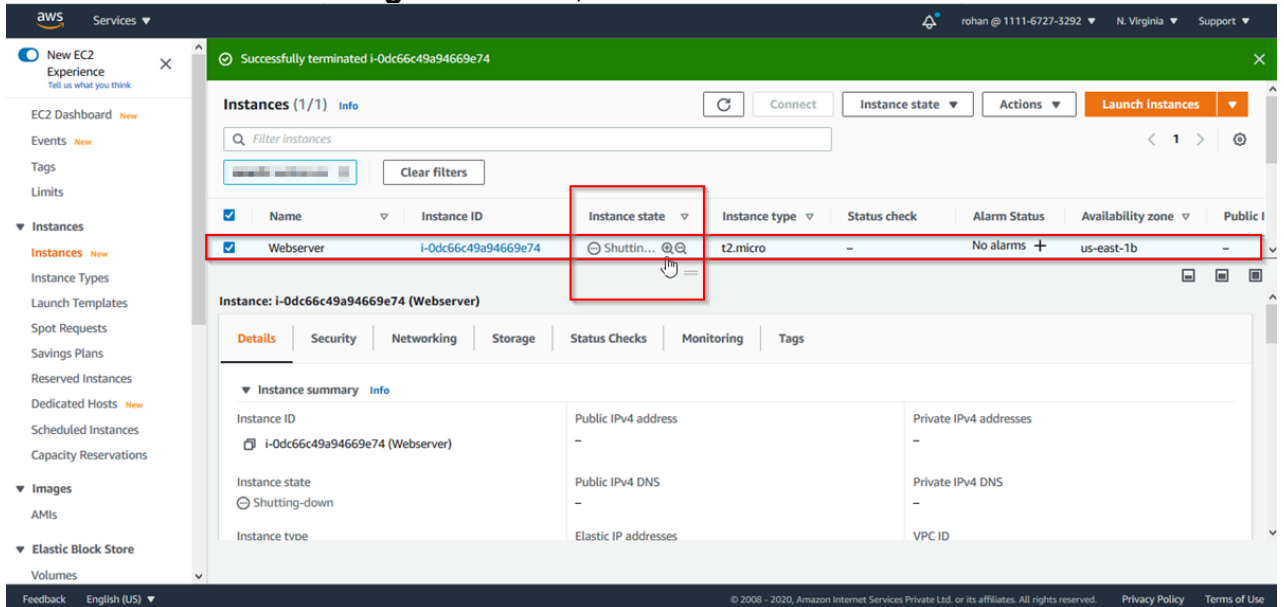
1. Select our instance, under Instance state Click on **Terminate instance**.



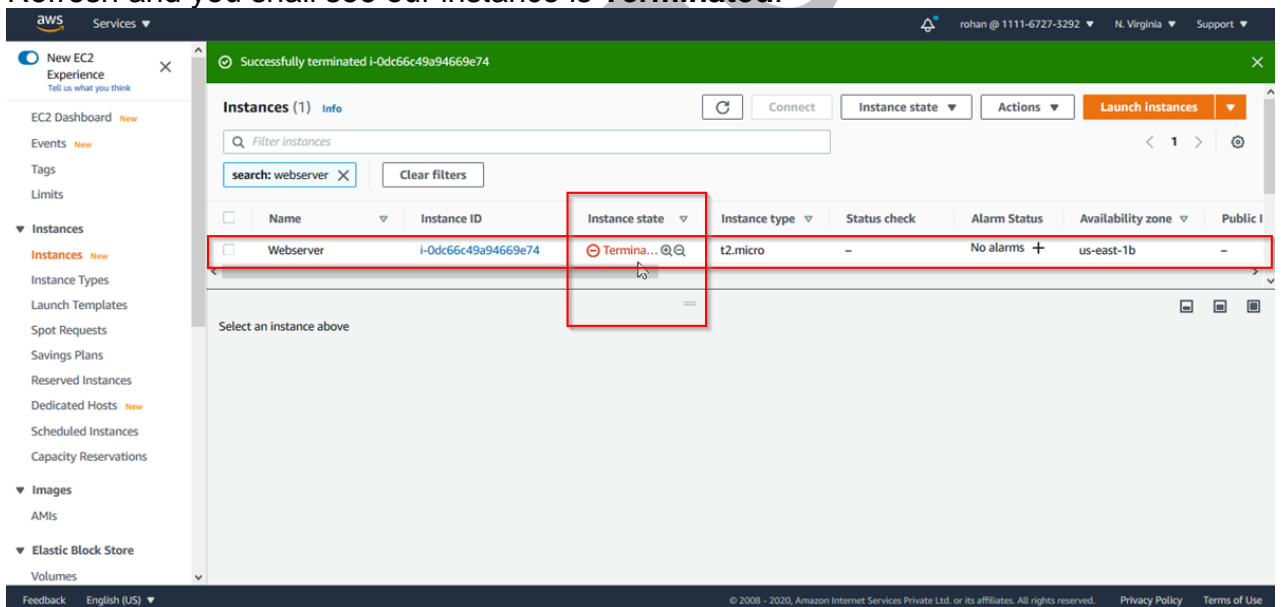
2. Now Click on **Terminate** to Confirm.



3. Your instance is in **shutting down** state, wait for some time.



4. Refresh and you shall see our instance is **Terminated**.



Thus, we have successfully Terminated the Linux EC2 instance.

8 TROUBLESHOOTING

Issue:

```
-bash: sudo: command not found
[ec2-user@ip-172-31-14-202 ~]$ sudo yum update -y
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
No packages marked for update
[ec2-user@ip-172-31-14-202 ~]$ sudo wget -O /etc/yum.repos.d/jenkins.repo http://
/pkg.jenkinsci.org/redhat/jenkins.repo
--2021-02-15 22:37:28-- http://pkg.jenkinsci.org/redhat/jenkins.repo
Resolving pkg.jenkinsci.org (pkg.jenkinsci.org)... failed: Name or service not k
nown.
wget: unable to resolve host address 'pkg.jenkinsci.org'
[ec2-user@ip-172-31-14-202 ~]$
```

```
[ec2-user@ip-172-31-14-202 ~]$ sudo service jenkins start
```

Starting jenkins (via systemctl): Job for jenkins.service failed because the control process exited with error code. See "systemctl status jenkins.service" and "journalctl -xe" for details.

[FAILED]

```
[ec2-user@ip-172-31-14-202 ~]$
```

```
ec2-user@ip-172-31-14-202:~
[ec2-user@ip-172-31-14-202 ~]$ sudo systemctl status jenkins.service
● jenkins.service - LSB: Jenkins Automation Server
   Loaded: loaded (/etc/rc.d/init.d/jenkins; bad; vendor preset: disabled)
   Active: failed (Result: exit-code) since Mon 2021-02-15 22:40:49 UTC; 1min 5s
   ago
     Docs: man:systemd-sysv-generator(8)
    Process: 3377 ExecStart=/etc/rc.d/init.d/jenkins start (code=exited, status=1/
FAILURE)

Feb 15 22:40:49 ip-172-31-14-202.us-east-2.compute.internal systemd[1]: Start...
Feb 15 22:40:49 ip-172-31-14-202.us-east-2.compute.internal runuser[3382]: pa...
Feb 15 22:40:49 ip-172-31-14-202.us-east-2.compute.internal jenkins[3377]: St...
Feb 15 22:40:49 ip-172-31-14-202.us-east-2.compute.internal jenkins[3377]: [F...
Feb 15 22:40:49 ip-172-31-14-202.us-east-2.compute.internal systemd[1]: jenk...
Feb 15 22:40:49 ip-172-31-14-202.us-east-2.compute.internal systemd[1]: Faile...
Feb 15 22:40:49 ip-172-31-14-202.us-east-2.compute.internal systemd[1]: Unit ...
Feb 15 22:40:49 ip-172-31-14-202.us-east-2.compute.internal systemd[1]: jengi...
Hint: Some lines were ellipsized, use -l to show in full.
[ec2-user@ip-172-31-14-202 ~]$ journalctl -xe
-- Defined-By: systemd
-- Support: http://lists.freedesktop.org/mailman/listinfo/systemd-devel
--
-- Unit jenkins.service has begun starting up.
Feb 15 22:40:49 ip-172-31-14-202.us-east-2.compute.internal runuser[3382]: pam_u
```

Solution:

<https://stackoverflow.com/questions/39621263/jenkins-fails-when-running-service-start-jenkins>

Search usable Java as /usr/bin/java might not point to minimal version required by Jenkins.

see <http://www.nabble.com/guinea-pigs-wanted-----Hudson-RPM-for-RedHat-Linux-td25673707.html>

candidates="

/etc/alternatives/java

/usr/lib/jvm/java-1.8.0/bin/java

/usr/lib/jvm/jre-1.8.0/bin/java

/usr/lib/jvm/java-11.0/bin/java

/usr/lib/jvm/jre-11.0/bin/java

/usr/lib/jvm/java-11-openjdk-amd64

/usr/bin/java

"

We need to install Java so make sure you have installed java and then you install all the Jenkins packages and keys.

9 SUMMARY

This activity guide covered steps to:

1. Launching a Linux Instance Using EC2
 - a. Creating an EC2 Instance
 - b. Accessing EC2 Instance Using Browser SSH Connection
 - c. Accessing EC2 Instance Using Putty
2. Install and Configure Jenkins
3. Deleting/Cleanup
 - a. Stopping the Linux Instance
 - b. Terminating the Linux Instance