



ELASTIC CLOUD COMPUTE (EC2)

Amazon Elastic Compute Cloud (Amazon EC2) is a web service provided by Amazon Web Services (AWS) that allows users to rent virtual servers in the cloud. EC2 instances provide scalable computing capacity, allowing users to quickly scale up or down based on their application requirements.

Key features of Amazon EC2 include:

1. **Scalability:** Users can easily scale their compute capacity by launching additional instances or stopping unused instances. This flexibility is particularly useful for applications with varying workloads.
2. **Variety of Instance Types:** EC2 offers a wide range of instance types optimized for different use cases, such as compute-optimized, memory-optimized, storage-optimized, and GPU instances.
3. **Customization:** Users can choose the operating system, instance type, storage, and networking configurations for their EC2 instances, providing a high degree of customization to meet specific application requirements.
4. **Security:** EC2 instances can be launched within a Virtual Private Cloud (VPC), allowing users to control network settings, security groups, and access control lists. Amazon EC2 also supports key pairs for secure access to instances.
5. **Pay-as-You-Go Pricing:** Users pay only for the compute capacity they consume, with different pricing options based on instance type, region, and usage duration. This pay-as-you-go model is cost-effective and allows for efficient resource utilization.
6. **Elastic Load Balancing:** EC2 instances can be easily integrated with Elastic Load Balancers to distribute incoming traffic across multiple instances, improving fault tolerance and availability.
7. **AMI (Amazon Machine Image):** Users can create custom machine images containing their applications, configurations, and operating systems, making it easy to launch instances with pre-configured settings.



TO BEGIN WITH LAB:



STEP 1: LAUNCH A LINUX EC2 INSTANCE

1. Log in to AWS Console.
2. There you need to search EC2. Choose this service accordingly.

The screenshot shows the AWS EC2 service page. At the top left, there is a small orange square icon with a white server symbol, followed by the text "EC2" and a yellow star icon. Below this, the text "Virtual Servers in the Cloud" is displayed. The background of the dashboard is dark blue.

3. This is the dashboard for EC2. You can find different resources here.
4. But, to create an instance you need to click on **Launch Instance**.

The screenshot shows the AWS EC2 Dashboard. On the left, a sidebar lists various EC2 services: Instances, Images, Elastic Block Store, Network & Security, and more. The main area is titled 'Resources' and displays a summary of Amazon EC2 resources in the Europe (London) Region. It includes tables for Instances (running), Auto Scaling Groups, Dedicated Hosts, Elastic IPs, Instances, Key pairs, Load balancers, Placement groups, Security groups, Snapshots, and Volumes. Below this, there's a 'Launch instance' section with a prominent orange 'Launch instance' button, a 'Migrate a server' link, and a note about launching instances in the Europe (London) Region. To the right, there's a 'Service health' section showing the status of various EC2 services across different regions, with some metrics like Linux EC2 Instances at 28% and Windows EC2 Instances at 2%. A separate box on the right details 'EC2 Free Tier' offers, noting 9 offers in use, 6 forecasted to exceed free tier, and 5 exceeded pay-as-you-go pricing.

5. So, first and foremost, you must name your instance.

The screenshot shows the 'Name and tags' step of the 'Launch an instance' wizard. It has a title bar 'Launch an instance' with an 'Info' link. Below it, a sub-section titled 'Name and tags' also has an 'Info' link. The 'Name' field contains the value 'LinuxVM'. To the right of the name field is a blue 'Add additional tags' link.

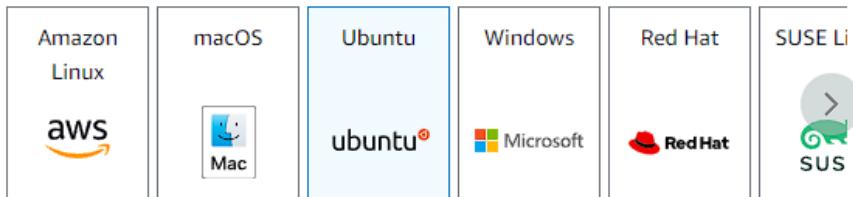
6. Then in the **Application and OS Images**, you have to select an Operating system.
7. Here, you are going to select **Ubuntu** as your Operating system.
8. For **Amazon Machine Image (AMI)**, select the latest AMI that is available. In this case it is ubuntu server 22.02 LTS.
9. Keep architecture to default as 64-bit (x86)

▼ Application and OS Images (Amazon Machine Image) [Info](#)

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below.

 *Search our full catalog including 1000s of application and OS images*

Quick Start



[Browse more AMIs](#)

Including AMIs from AWS, Marketplace and the Community

Amazon Machine Image (AMI)

Ubuntu Server 22.04 LTS (HVM), SSD Volume Type
ami-0e5f882be1900e43b (64-bit (x86)) / ami-00efc25778562c229 (64-bit (Arm))
Virtualization: hvm ENA enabled: true Root device type: ebs

Free tier eligible

Description

Canonical, Ubuntu, 22.04 LTS, amd64 jammy image build on 2023-12-07

Architecture

AMI ID

64-bit (x86) ▾

ami-0e5f882be1900e43b

Verified provider

10. Now you must choose an instance type. There are more instance types with higher specifications available, but for the time being, you must select **t2.micro**. This instance type is qualified for the free tier.

▼ Instance type [Info](#) | [Get advice](#)

Instance type

t2.micro
Family: t2 1 vCPU 1 GiB Memory Current generation: true
On-Demand Windows base pricing: 0.0178 USD per Hour
On-Demand RHEL base pricing: 0.0732 USD per Hour
On-Demand SUSE base pricing: 0.0132 USD per Hour
On-Demand Linux base pricing: 0.0132 USD per Hour

Free tier eligible

All generations

[Compare instance types](#)

Additional costs apply for AMIs with pre-installed software

11. In the key pair, you have to create a key pair, so that you can log in to your virtual machine. These key pairs are used as keys to log into your VMs.
12. As of now, you might not have any key pairs, so you have to create a key pair.

▼ 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

13. So, click on create new key pair. First you have to name your keypair.
14. Then select the key pair type as RSA.
15. For the private key file format, select **.pem**
16. Now click on create key pair.

Create key pair X

Key pair name

Key pairs allow you to connect to your instance securely.

linux-key

The name can include up to 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

Private key file format

.pem

For use with OpenSSH

.ppk

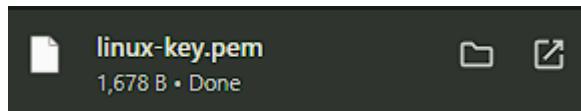
For use with PuTTY

⚠ 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 !\[\]\(ccd39a0dc6d5afcc151e1371f9462f58_img.jpg\)](#)

Cancel

Create key pair

17. After clicking on the Create Key Pair button, a file should have been downloaded; this file will be utilized throughout this experiment.



18. Here you can see your key pair.

A screenshot of the AWS Lambda console under the "Configuration" tab. It shows a section titled "Key pair (login)" with a "Info" link. Below it, a note says: "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." A dropdown menu shows "linux-key" selected, and a "Create new key pair" button is visible.

19. If you will take a look at network settings, you will see a VPC which is required and it is a default VPC.
20. The subnet has no preference because you haven't given it any.
21. The **auto assign public IP** feature is enabled.

A screenshot of the AWS Lambda console under the "Configuration" tab. It shows a section titled "Network settings" with an "Info" link. Under "VPC - required", a dropdown shows "vpc-037cc333342ffff6f0 (default) 172.31.0.0/16". Below it, a "Subnet" dropdown shows "No preference" and a "Create new subnet" button. At the bottom, an "Auto-assign public IP" dropdown is set to "Enable".

22. In the firewall option, you will see that it is creating a new security group, you can change its name and description, but for the time being keep it to default.

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*

launch-wizard-1

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](#)

launch-wizard-1 created 2024-01-10T16:27:27.542Z

23. And the security group you will see this Inbound security group rules. These Inbound rules are used for virtual machine to log in to different areas of interest. Let it be default too.

Inbound Security Group Rules

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

Type Info	Protocol Info	Port range Info
ssh	TCP	22
Source type Info	Source Info	Description - <i>optional</i> Info
Anywhere	<input style="width: 150px; height: 20px; border: 1px solid #ccc; padding: 2px; margin-bottom: 5px;" type="text" value="Add CIDR, prefix list or security"/> <input style="width: 150px; height: 20px; border: 1px solid #ccc; padding: 2px; margin-bottom: 5px;" type="text" value="0.0.0.0/0"/>	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. X

[Add security group rule](#)

24. Now the setup for EC2 instance is completed, you can click on launch instance. This will launch your instance and you will be able to see your instance.

[Cancel](#)

[Launch instance](#)

[Review commands](#)

25. You can see that your instance is launched successfully.

26. Now if you will click on this instance id, you will directed towards your instance.

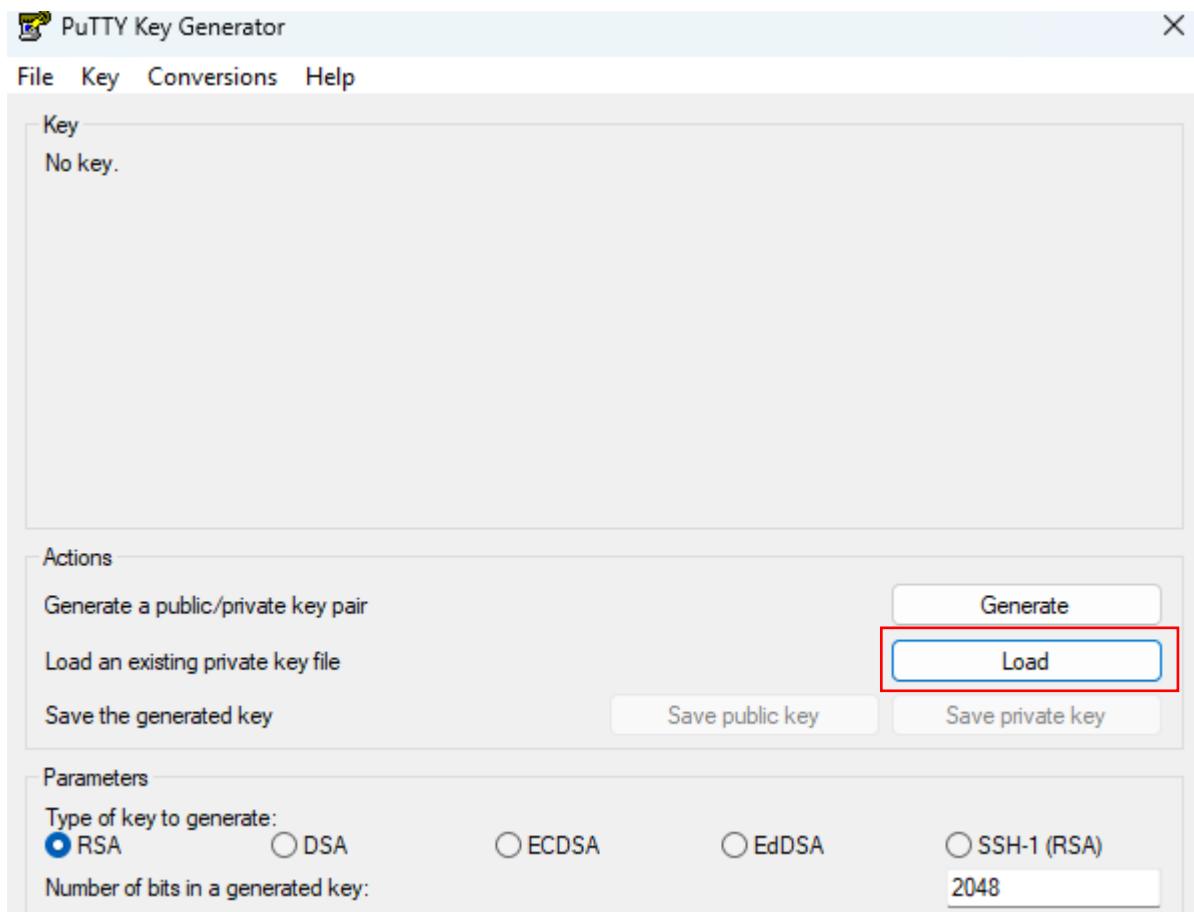


27. Here is your instance, it is up and running.

A screenshot of the AWS EC2 Instances list. It shows one instance named "LinuxVM" with the ID "i-0bc7f2f0b89682297". The instance is listed as "Running" with a status check of "Initializing". It is located in the "eu-west-2b" availability zone and has a public IPv4 address of "13.41.56.8". The "Actions" dropdown menu is open, showing options like "Stop", "Start", "Reboot", "Termination Protection", and "Launch instances". The entire screenshot is framed by a red border.

STEP 2: CONNECTING THE INSTANCE

1. So, if you want to connect your instance, you have to download **Putty**.
2. **Putty** is a tool which helps users to connect their instance and work on it.
3. To download **Putty**, you can use this website. <https://www.putty.org/>
4. With **Putty** you also need to download a tool called **PuttyGen**.
5. **PuttyGen** can also be downloaded from this website too.
6. Now once you have downloaded Putty and puttygen tools. First you have to open PuttyGen because this tool will help you to convert your .pem file to .ppk because Putty does not support .pem files it only supports .ppk files.
7. Now open puttygen. Click on load. This option opens your system files to browse from them.



8. Once you have loaded the .pem file here, now you have to click on **Save Private key**.
This tool will save your file as a private key.

Key

Public key for pasting into OpenSSH authorized_keys file:

```
ssh-rsa AAAAB3NzaC1yc2EAAAQABAAQCPvDJleMVgNKiloR2RV8ga05rd6+LHtNm
+civcvs9bYxq2BggSzN4rJGT4vXxowubBooL/7w/RIB6OA3eBKE8YqZHSkrN9XmzD0hQCi22sf78PtTCUOCCnm
AYpTSkVG1VNp6q1jE6WM2lZR+jJL+aG7RgGHUscVTjd3Utt8okpXrKPXX2JaK3oZ
+iOJ6FV3i6QRsQPRM0EAvk0/RohTPRWcN5wVDc5jG2ReTTAMg5Bpicu1/2F7NFh1xKOeLGHavrAjhVwxDOV
or7RHlcdONHN+uf3A7xU/9yqh83kmHDM3SMJJ9VvbChkpFaEHoMiOMIDfwSs1rvsNZGF8mP imported-
```

Key fingerprint: ssh-rsa 2048 SHA256:jmXWiY+S6T5BnOUiQ6duxmm06ZxP5AlgSGvjb7TRG0

Key comment: imported-openssh-key

Key passphrase:

Confirm passphrase:

Actions

Generate a public/private key pair

Load an existing private key file

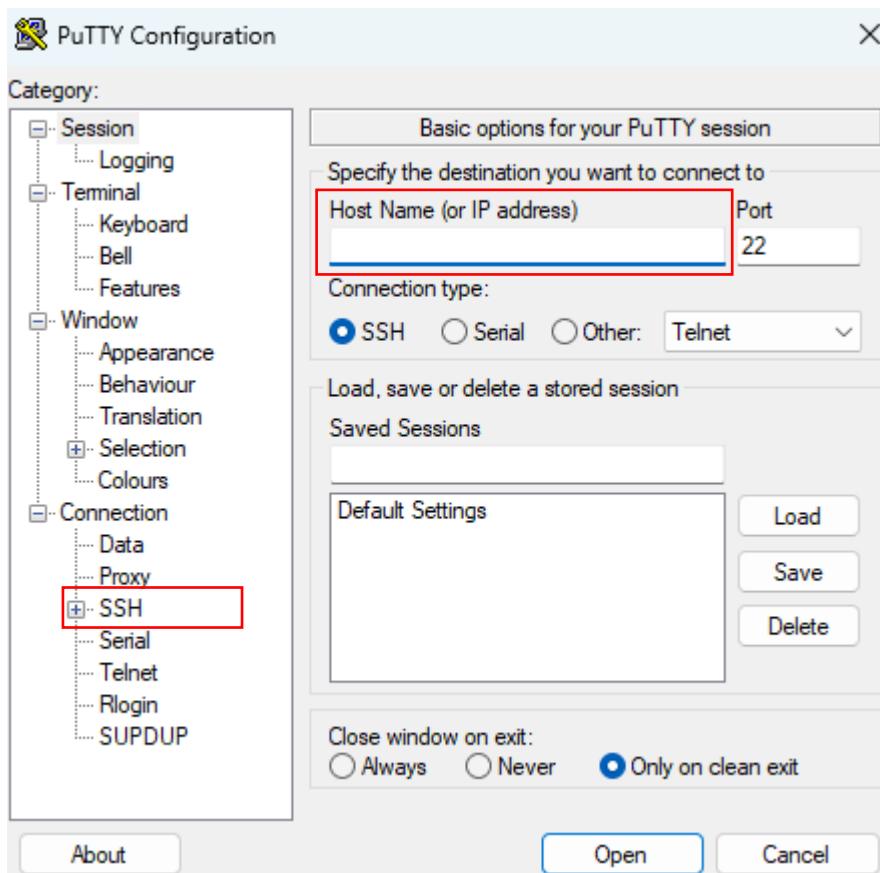
Save the generated key

Parameters

Type of key to generate: RSA DSA ECDSA EdDSA SSH-1 (RSA)

Number of bits in a generated key: 2048

9. Now open **Putty** and in the host name you have to paste the public IP address of the instance. So, copy that and paste it in the host name.
10. Once you've pasted the IP, now you have to click on SSH.

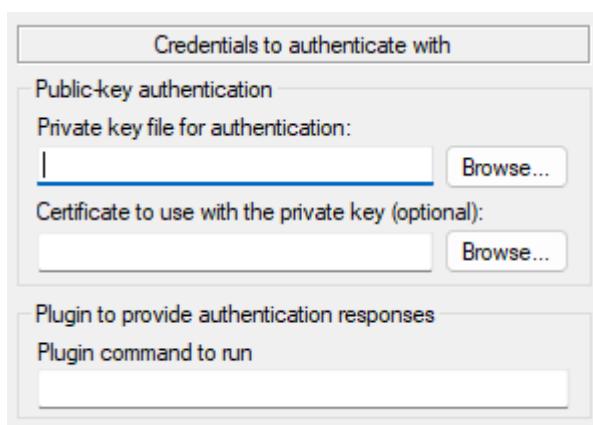


11. After clicking on SSH, you will see some more options, so you have to choose Auth, then after click on credentials.



12. In the credentials you have to browse to the place where you saved your .ppk file after converting it.

13. Then once the location is saved here. You can click on Open.



14. You will see that you have logged into the server.

15. Here in login as you have to **Write ubuntu**, because this will be your username to login.

```
ubuntu@ip-172-31-35-74: ~
login as: ubuntu
Authenticating with public key "imported-openssh-key"
Welcome to Ubuntu 22.04.3 LTS (GNU/Linux 6.2.0-1017-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

System information as of Wed Jan 10 17:14:38 UTC 2024

System load:  0.0          Processes:            95
Usage of /:   20.5% of 7.57GB  Users logged in:    0
Memory usage: 21%          IPv4 address for eth0: 172.31.35.74
Swap usage:   0%

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/*copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-35-74:~$
```

😊 STEP 3: CONNECTING THE INSTANCE THOUGH SSH CLIENT

1. To connect you instance via SSH client you have click on connect on the console.
2. Then you have to navigate to SSH client.
3. Now open command prompt on your local machine.

Connect to instance Info

Connect to your instance i-0bc7f2f0b89682297 (LinuxVM) using any of these options

EC2 Instance Connect

Session Manager

SSH client

EC2 serial console

Instance ID

i-0bc7f2f0b89682297 (LinuxVM)

1. Open an SSH client.
2. Locate your private key file. The key used to launch this instance is linux-key.pem
3. Run this command, if necessary, to ensure your key is not publicly viewable.
 chmod 400 "linux-key.pem"
4. Connect to your instance using its Public DNS:
 ec2-13-41-56-8.eu-west-2.compute.amazonaws.com

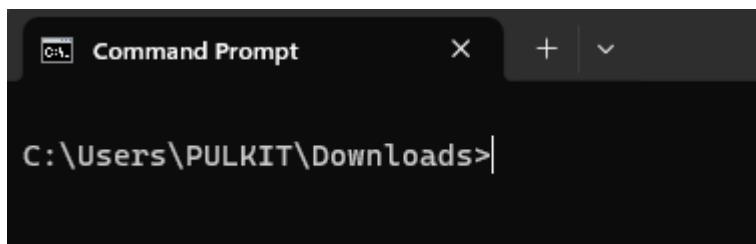
Example:

ssh -i "linux-key.pem" ubuntu@ec2-13-41-56-8.eu-west-2.compute.amazonaws.com

Note: In most cases, the guessed username is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI username.

Cancel

4. After opening command prompt navigate cmd to the location where your .ppk file is installed.



5. Once you are on the location where you have stored your .ppk file now you have to paste that example link that you copied earlier.
6. If you encounter this error then you can run these two commands which will give it permission to log in your instance.
icacls .\linuxkey.pem /inheritance:r
icacls .\linuxkey.pem /grant:"Administrator:(r)"

```
Bad permissions. Try removing permissions for user: BUILTIN\\Users (S-1-5-32-545) on file C:/tmp/linuxkey.pem.
@@@@@@@WARNING: UNPROTECTED PRIVATE KEY FILE!@@@@@@@
Permissions for 'linuxkey.pem' are too open.
It is required that your private key files are NOT accessible by others.
This private key will be ignored.
Load key "linuxkey.pem": bad permissions
ubuntu@ec2-13-40-184-170.eu-west-2.compute.amazonaws.com: Permission denied (publickey).
```

7. Once you have those two commands. Now try again to log in.

8. You will see that you have successfully logged into your Linux instance successfully.

```
C:\Users\PULKIT\Downloads>ssh -i "linux-key.pem" ubuntu@ec2-13-41-56-8.eu-west-2.compute.amazonaws.com
Welcome to Ubuntu 22.04.3 LTS (GNU/Linux 6.2.0-1017-aws x86_64)

 * Documentation: https://help.ubuntu.com
 * Management: https://landscape.canonical.com
 * Support: https://ubuntu.com/advantage

System information as of Wed Jan 10 17:42:23 UTC 2024

System load: 0.0          Processes:         97
Usage of /: 20.8% of 7.57GB  Users logged in:      1
Memory usage: 21%          IPv4 address for eth0: 172.31.35.74
Swap usage:  0%

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

Last login: Wed Jan 10 17:14:39 2024 from 192.140.153.216
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-35-74:~$ |
```

💡 STEP 4: DEPLOYING A WEB SERVER ON THE INSTANCE

1. Now you are going to install NGINX web server on your Linux instance.
2. For that you have to run a set of commands on your instance. You can choose to run these either in Putty or on Command Prompt.
3. The first command is for updating your Linux instance server.
sudo apt-get update

```
ubuntu@ip-172-31-35-74:~$ sudo apt-get update
```

4. The second command is for installing your nginx server.

```
sudo apt-get update install nginx
```

```
ubuntu@ip-172-31-35-74:~$ sudo apt-get install nginx
```

5. So, now you have to go back to your console and add port 80 to your security group.
6. Now select your instance and go to security, there you will a link to navigate you to the security group. Click on it.

Instance: i-0bc7f2f0b89682297 (LinuxVM)

Details | **Security** | Networking | Storage | Status checks | Monitoring | Tags

▼ Security details

IAM Role

Owner ID

463646775279

Launch time

Wed Jan 10 2024 22:22:18 GMT+0530 (India Standard Time)

Security groups

sg-0a32c29b69fd4424f (launch-wizard-1)

7. In the security group you will see some options present over there.
8. If you will just click on edit inbound rules under the inbound rules and add port 80 to your instance.

The screenshot shows the AWS Security Groups console for the security group `sg-0a32c29b69fd4424f - launch-wizard-1`. The **Details** tab is selected, displaying information such as the security group name, owner, and VPC ID. The **Inbound rules** tab is active, showing one rule: `sgr-03bf3cb4e17e33d91` (SSH, TCP port 22, source 0.0.0.0/0). The **Edit inbound rules** button is visible. Below this, the **Inbound rules info** section shows the detailed configuration of the two rules, including protocol, port range, and source. A warning message at the bottom advises against using 0.0.0.0/0 as a source. The **Save rules** button is highlighted in orange.

Security group rule ID	Type	Protocol	Port range	Source	Description
<code>sgr-03bf3cb4e17e33d91</code>	SSH	TCP	22	Custom 0.0.0.0/0	-
-	HTTP	TCP	80	Anywhere... 0.0.0.0/0	-

9. Now go back to your instance and copy your public IP address and paste it in a new tab.

10. You will be able to see the nginx server up and running.

The screenshot shows a web browser window with the URL `13.41.56.8`. The page displays the standard "Welcome to nginx!" message, indicating that the nginx web server is installed and working. The browser status bar shows "Not secure" and the IP address `13.41.56.8`.

