

EC2 Instance Setup and Apache Web Server Deployment Project

Project Description

This project involves deploying a simple HTML web page on an AWS EC2 instance using the Apache Web Server. The main goal is to understand cloud infrastructure setup, Linux server management, and web server deployment.

The project includes the following steps:

1. Launching an EC2 instance on AWS and configuring its network and security settings.
2. Connecting to the instance via SSH to perform server-side operations.
3. Installing and configuring the Apache Web Server on the EC2 instance.
4. Deploying a custom HTML page in the web server's root directory.
5. Configuring firewall and security group rules to allow HTTP access from the internet.
6. Testing the web page by accessing the EC2 public IP in a browser.

Project Description

This project involves deploying a simple HTML web page on an AWS EC2 instance using the Apache Web Server. The main goal is to understand cloud infrastructure setup, Linux server management, and web server deployment.

The project includes the following steps:

1. Launching an EC2 instance on AWS and configuring its network and security settings.
2. Connecting to the instance via SSH to perform server-side operations.
3. Installing and configuring the Apache Web Server on the EC2 instance.
4. Deploying a custom HTML page in the web server's root directory.
5. Configuring firewall and security group rules to allow HTTP access from the internet.
6. Testing the web page by accessing the EC2 public IP in a browser.

Step 1:

1. Launch an EC2 Instance

1. Login to AWS Console → Navigate to EC2 → Click Launch Instance.
2. Choose an Amazon Machine Image (AMI): Select Amazon Linux / CentOS 9 / Ubuntu depending on your preference.

EC2 > Instances > Launch an instance

① It seems like you may be new to launching instances in EC2. Take a walkthrough to learn about EC2, how to launch instances and about best practices

Take a walkthrough

Do not show me this message again.

Launch an instance

Info

Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.

Name and tags

Info

Name

Ec2_Server1

Add additional tags

▼ Application and OS Images (Amazon Machine Image)

Info

An AMI contains the operating system, application server, and applications for your instance. If you don't see a suitable AMI below, use the search field or choose **Browse more AMIs**.

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

Recents

Quick Start

Amazon Linux

macOS

Ubuntu

Windows

Red Hat

SUSE Linux

Debian

Q

Browse more AMIs

3. Select Instance Type: For testing, t2.micro (free tier) is enough.
4. Configure Instance: Keep default network settings or customize if needed.
5. Add Storage: Default 8 GB is sufficient.
6. Configure Security Group:
 - o Allow SSH (port 22) for your IP.
 - o Allow HTTP (port 80) for public access.
 - o Optionally, allow HTTPS (port 443) if planning SSL.

Subnet | Info
No preference (Default subnet in any availability zone)

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

We'll create a new security group called 'launch-wizard-3' with the following rules:

- ☒ Allow SSH traffic from
Helps you connect to your instance
Anywhere
0.0.0.0/0
- ☒ Allow HTTPS traffic from the internet
To set up an endpoint, for example when creating a web server
- ☒ Allow HTTP traffic from the internet
To set up an endpoint, for example when creating a web server

⚠ 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.

▼ **Configure storage** | Info Advanced

1x | 8 GiB | gp3 | Root volume, 3000 IOPS, Not encrypted

7. Review and Launch → Select key pair (create new or use existing) → Launch Instance.

Step 2: Connect to EC2 via SSH

1. Open terminal (Linux/macOS) or use PuTTY (Windows).

Instances (1/1) | Info

Find Instance by attribute or tag (case-sensitive) | All states | Last updated 3 minutes ago | Connect | Instance state | Actions | Launch instances

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IP
EC2_Server1	i-0fbc7958efc3f7e12	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1d	ec2-44-223-65-48.com...	44.223.6...

i-0fbc7958efc3f7e12 (EC2_Server1)

Details | Status and alarms | Monitoring | Security | Networking | Storage | Tags

▼ **Instance summary** | Info

Instance ID i-0fbc7958efc3f7e12	Public IPv4 address 44.223.65.48 open address	Private IPv4 addresses 172.31.24.253
IPv6 address -	Instance state Running	Public DNS ec2-44-223-65-48.compute-1.amazonaws.com open address
Hostname type IP name: ip-172-31-24-253.ec2.internal	Private IP DNS name (IPv4 only) ip-172-31-24-253.ec2.internal	
Answer private resource DNS name	Instance type	Elastic IP addresses

2. Run command:

Install Apache (ex. Httpd Service)

```
sudo yum install httpd -y
```

```

[ec2-user@ip-172-31-24-253 ~]$ sudo -i
[root@ip-172-31-24-253 ~]# sudo yum install httpd -y
Amazon Linux 2023 Kernel Livepatch repository                234 kB/s | 26 kB    00:00
Dependencies resolved.
=====
Package                Arch      Version              Repository           Size
=====
Installing:
httpd                  x86_64    2.4.65-1.amzn2023.0.1  amazonlinux          47 k
Installing dependencies:
apr                    x86_64    1.7.5-1.amzn2023.0.4  amazonlinux          129 k
apr-util               x86_64    1.6.3-1.amzn2023.0.1  amazonlinux          98 k
generic-logos-httpd    noarch    18.0.0-12.amzn2023.0.3  amazonlinux          19 k
httpd-core              x86_64    2.4.65-1.amzn2023.0.1  amazonlinux          1.4 M
httpd-filesystem        noarch    2.4.65-1.amzn2023.0.1  amazonlinux          13 k
httpd-tools             x86_64    2.4.65-1.amzn2023.0.1  amazonlinux          81 k
libbrotli               x86_64    1.0.9-4.amzn2023.0.2  amazonlinux          315 k
mailcap                 noarch    2.1.49-3.amzn2023.0.3  amazonlinux          33 k
Installing weak dependencies:
apr-util-openssl        x86_64    1.6.3-1.amzn2023.0.1  amazonlinux          17 k
mod_http2                x86_64    2.0.27-1.amzn2023.0.3  amazonlinux          166 k
mod_lua                  x86_64    2.4.65-1.amzn2023.0.1  amazonlinux          60 k
=====
Transaction Summary
=====
Install 12 Packages

Total download size: 2.3 M
Installed size: 6.9 M
Downloading Packages:
(1/12): apr-util-openssl-1.6.3-1.amzn2023.0.1.x86_64.rpm 462 kB/s | 17 kB    00:00
(2/12): apr-1.7.5-1.amzn2023.0.4.x86_64.rpm             2.9 MB/s | 129 kB   00:00
(3/12): apr-util-1.6.3-1.amzn2023.0.1.x86_64.rpm         2.0 MB/s | 98 kB    00:00
(4/12): generic-logos-httpd-18.0.0-12.amzn2023.0.3.noarch 807 kB/s | 19 kB    00:00
(5/12): httpd-2.4.65-1.amzn2023.0.1.x86_64.rpm          1.7 MB/s | 47 kB    00:00

```

Start the Service

`sudo systemctl start httpd` # Start Apache

`sudo systemctl enable httpd` # Enable at boot

`sudo systemctl status httpd` # Check status

```

[root@ip-172-31-24-253 ~]# sudo systemctl start httpd
[root@ip-172-31-24-253 ~]# sudo systemctl enable httpd
[root@ip-172-31-24-253 ~]# sudo systemctl status httpd
● httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled; preset: disabled)
   Active: active (running) since Mon 2025-10-13 05:10:47 UTC; 2min 44s ago
     Docs: man:httpd.service(8).
    Main PID: 26653 (httpd)
      Status: "Total requests: 0; Idle/Busy workers 100/0; Requests/sec: 0; Bytes served/sec: 0 B/sec"
       Tasks: 177 (limit: 1106)
      Memory: 13.0M
         CPU: 156ms
    CGroup: /system.slice/httpd.service
            └─26653 /usr/sbin/httpd -DFOREGROUND
              └─26654 /usr/sbin/httpd -DFOREGROUND
                └─26655 /usr/sbin/httpd -DFOREGROUND
                  └─26656 /usr/sbin/httpd -DFOREGROUND
                    └─26657 /usr/sbin/httpd -DFOREGROUND

Oct 13 05:10:47 ip-172-31-24-253.ec2.internal systemd[1]: Starting httpd.service - The Apache HTTP Server...
Oct 13 05:10:47 ip-172-31-24-253.ec2.internal systemd[1]: Started httpd.service - The Apache HTTP Server.
Oct 13 05:10:47 ip-172-31-24-253.ec2.internal httpd[26653]: Server configured, listening on: port 80
[root@ip-172-31-24-253 ~]#

```

Test:

`echo "<h1>Welcome to My Apache Web Server on EC2</h1>" | sudo tee /var/www/html/index.html`

```
Oct 13 05:10:47 ip-172-31-24-253.ec2.internal httpd[26653]: Server configured, listening on: port 80
[root@ip-172-31-24-253 ~]# echo "<h1>Welcome to My Apache Web Server on EC2</h1>" | sudo tee /var/www/html/index.html
<h1>Welcome to My Apache Web Server on EC2</h1>
[root@ip-172-31-24-253 ~]#
```

Step 3: Install Firewall

sudo yum install firewalld -y

```
[root@ip-172-31-24-253 ~]# sudo yum install firewalld -y
Last metadata expiration check: 0:07:02 ago on Mon Oct 13 05:10:06 2025.
Dependencies resolved.
```

Package	Architecture	Version
Installing:		
firewalld	noarch	1.2.3-1.amzn2023
Installing dependencies:		
firewalld-filesystem	noarch	1.2.3-1.amzn2023
gobject-introspection	x86_64	1.82.0-1.amzn2023
ipset	x86_64	7.11-1.amzn2023.0.3
ipset-libs	x86_64	7.11-1.amzn2023.0.3
iptables-libs	x86_64	1.8.8-3.amzn2023.0.2
iptables-nft	x86_64	1.8.8-3.amzn2023.0.2
libnetfilter_conntrack	x86_64	1.0.8-2.amzn2023.0.2
libnftnl	x86_64	1.0.1-19.amzn2023.0.2
libnftnl	x86_64	1.2.2-2.amzn2023.0.2
nftables	x86_64	1:1.0.4-3.amzn2023.0.2
python3-firewall	noarch	1.2.3-1.amzn2023
python3-gobject-base	x86_64	3.48.2-3.amzn2023.0.2
python3-nftables	x86_64	1:1.0.4-3.amzn2023.0.2
Installing weak dependencies:		
libcap-ng-python3	x86_64	0.8.2-4.amzn2023.0.2

Transaction Summary

Install 15 Packages

Total download size: 2.5 M

Installed size: 11 M

Downloading Packages:

(1/15): firewalld-filesystem-1.2.3-1.amzn2023.noarch.rpm

(2/15): gobject-introspection-1.82.0-1.amzn2023.x86_64.rpm

sudo systemctl start firewalld

sudo firewall-cmd --permanent --add-service=http

sudo firewall-cmd --reload

```
[root@ip-172-31-24-253 ~]# sudo systemctl start firewalld
[root@ip-172-31-24-253 ~]# sudo firewall-cmd --permanent --add-service=http
success
[root@ip-172-31-24-253 ~]# sudo firewall-cmd --reload
success
[root@ip-172-31-24-253 ~]#
```

Step 4:

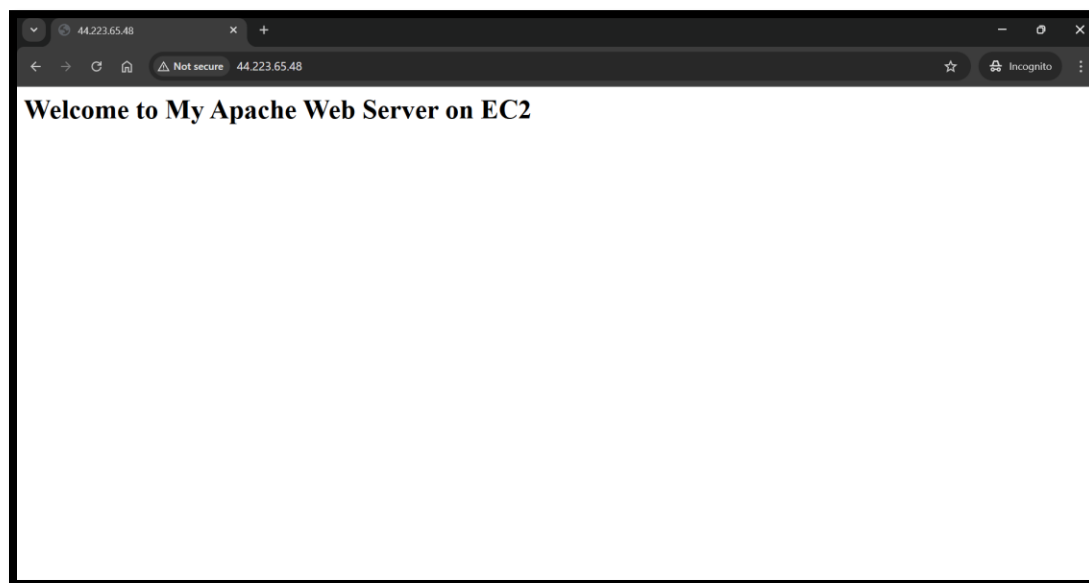
Use curl <your Public IP> to check webpage

```

services: dhcpv6-client http mdns ssh
ports:
protocols:
forward: yes
masquerade: no
forward-ports:
source-ports:
icmp-blocks:
rich rules:
[root@ip-172-31-24-253 ~]# ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data:
64 bytes from 8.8.8.8: icmp_seq=1 ttl=117 time=0.956 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=117 time=1.01 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=117 time=1.24 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=117 time=0.993 ms
64 bytes from 8.8.8.8: icmp_seq=5 ttl=117 time=1.72 ms
64 bytes from 8.8.8.8: icmp_seq=6 ttl=117 time=1.28 ms
64 bytes from 8.8.8.8: icmp_seq=7 ttl=117 time=1.33 ms
64 bytes from 8.8.8.8: icmp_seq=8 ttl=117 time=1.25 ms
64 bytes from 8.8.8.8: icmp_seq=9 ttl=117 time=1.00 ms
64 bytes from 8.8.8.8: icmp_seq=10 ttl=117 time=1.20 ms
64 bytes from 8.8.8.8: icmp_seq=11 ttl=117 time=1.46 ms
64 bytes from 8.8.8.8: icmp_seq=12 ttl=117 time=1.01 ms
64 bytes from 8.8.8.8: icmp_seq=13 ttl=117 time=1.24 ms
64 bytes from 8.8.8.8: icmp_seq=14 ttl=117 time=0.999 ms
64 bytes from 8.8.8.8: icmp_seq=15 ttl=117 time=1.02 ms
64 bytes from 8.8.8.8: icmp_seq=16 ttl=117 time=1.19 ms
64 bytes from 8.8.8.8: icmp_seq=17 ttl=117 time=1.38 ms
64 bytes from 8.8.8.8: icmp_seq=18 ttl=117 time=0.993 ms
^C
-- 8.8.8.8 ping statistics --
18 packets transmitted, 18 received, 0% packet loss, time 17024ms
rtt min/avg/max/mdev = 0.956/1.181/1.723/0.200 ms
[root@ip-172-31-24-253 ~]# curl 44.223.65.48
<h1>Welcome to My Apache Web Server on EC2</h1>
[root@ip-172-31-24-253 ~]#

```

Open public IP in New window you will get your webpage here.



Skills Learned

- Launching and configuring EC2 instances
- Accessing EC2 using SSH
- Installing Apache Web Server on Linux
- Configuring firewall and security groups
- Deploying a basic HTML web page
- Using Linux commands for server management

Project Summary

Deployed a simple HTML web page on an AWS EC2 instance using Apache Web Server. The project involved launching and configuring an EC2 instance, connecting via SSH, installing Apache, deploying a web page, and configuring firewall and security settings. Gained hands-on experience in cloud infrastructure, Linux server management, web server deployment, and network security.

Conclusion

This project successfully demonstrated the deployment of a web server on an AWS EC2 instance. It provided practical experience in cloud computing, Linux server management, and web service hosting. By completing this project, skills in configuring EC2 instances, installing Apache, managing security settings, and deploying web content were effectively enhanced, preparing for real-world cloud and server administration tasks.

Project Member

1. Sakshi Jain
2. Umesh Chimankar
3. Jay Soni
4. Nikita Binnar
5. Vaishavi Kumbhar
6. Kaveri Kanawade