

Building a Web Server on AWS EC2: A Journey from Manual Setup to Cloud Automation

Introduction: Building a Cloud Web Server the Smart Way

Imagine launching a web server in the cloud. For most beginners, it's a monumental task, but with the right tools, we can simplify the process while still learning the ropes of **AWS**, **Apache**, **SSL**, and **CloudFormation**.

In this journey, I'll guide you through setting up a **web server** on **AWS EC2**, installing **Apache**, securing the connection with **SSL certificates**, and automating the process using **CloudFormation**. Along the way, you'll see both the manual and automated approaches – the “manual” approach lets us learn every step thoroughly, while the “automated” approach saves us time and headaches in the future.

Let's start with a manual configuration, then use **AWS CloudFormation** to replicate the setup automatically. We'll finish by connecting our **custom domain** and securing the server with **Let's Encrypt SSL**.

Part 1: The Manual Setup - Understanding the Basics

Before diving into automation, we'll first manually configure the server. Think of this step as the “DIY” phase. While automation can save time, it's important to understand the fundamental steps that go into setting up the server.

Step 1: Launching the EC2 Instance

AWS EC2 (Elastic Compute Cloud) is the heart of cloud computing. It provides **scalable compute capacity**, and you can think of it as your virtual machine in the cloud. For our project, we're using a **t2.micro** instance – a great choice for learning and small applications, especially since it falls under the AWS Free Tier.

- Log into your **AWS Management Console** and navigate to **EC2**.
- Click **Launch Instance**, and select **Ubuntu** as the operating system.
- Choose **t2.micro** (because it's free within the AWS Free Tier).
- Set up a **Security Group** that allows traffic on:
 - **Port 22 (SSH)**: So we can access the server.
 - **Port 80 (HTTP)**: To serve our website.
 - **Port 443 (HTTPS)**: For encrypted traffic (more on this soon).
 - **Port 10000**: For Webmin (our server management tool).

Why EC2?

AWS EC2 is widely used for its flexibility and scalability. You can easily resize instances as your needs grow. Whether you're building a small app or a full enterprise infrastructure, EC2 has you covered.

EC2 Instance Launch Settings

EC2 > Instances > Launch an instance

⋮ ⌂ ⌃ ⌄

Name and tags [Info](#)

Name
Cloud-DevOps_Webserver [Add additional tags](#)

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

Amazon Linux macOS Ubuntu Windows Red Hat SUSE Linux Debian

[Browse more AMIs](#)
Including AMIs from AWS, Marketplace and the Community

Amazon Machine Image (AMI)

Ubuntu Server 24.04 LTS (HVM), SSD Volume Type
ami-084568db4383264d4 (64-bit (x86)) / ami-0c4e709339fa8521a (64-bit (Arm))
Virtualization: hvm ENA enabled: true Root device type: ebs [Free tier eligible](#)

Description
Ubuntu Server 24.04 LTS (HVM), EBS General Purpose (SSD) Volume Type. Support available from Canonical (<http://www.ubuntu.com/cloud/services>).

Canonical, Ubuntu, 24.04, amd64 noble image

Architecture: 64-bit (x86) AMI ID: ami-084568db4383264d4 Publish Date: 2025-03-05 Username: ubuntu [Verified provider](#)

Summary

Number of instances [Info](#)
1

Software Image (AMI)
Canonical, Ubuntu, 24.04, amd64... [read more](#)
ami-084568db4383264d4

Virtual server type (instance type)
t2.micro

Firewall (security group)
New security group

Storage (volumes)
1 volume(s) - 8 GiB

Free tier: In your first year of opening an AWS account, you get 750 hours per month of t2.micro instance usage (or t3.micro where t2.micro isn't available) when used with free tier AMIs, 750 hours per month of public IPv4 address usage, 30 GiB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GB of bandwidth to the internet.

[Cancel](#) [Launch instance](#) [Preview code](#)

EC2 > Instances > Launch an instance

⋮ ⌂ ⌃ ⌄

Network settings [Info](#) [Edit](#)

Network [Info](#)
vpc-0c2072053840da3a4

Subnet [Info](#)
No preference (Default subnet in any availability zone)

Auto-assign public IP [Info](#)
Enable
Additional charges apply when outside of free tier allowance

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-1' with the following rules:

Allow SSH traffic from Anywhere
Helps you connect to your instance
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

Summary

Number of instances [Info](#)
1

Software Image (AMI)
Canonical, Ubuntu, 24.04, amd64... [read more](#)
ami-084568db4383264d4

Virtual server type (instance type)
t2.micro

Firewall (security group)
New security group

Storage (volumes)
1 volume(s) - 8 GiB

[Cancel](#) [Launch instance](#) [Preview code](#)

Configure storage [Info](#) [Advanced](#)

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

[Add new volume](#)

Free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage

The selected AMI contains instance store volumes, however the instance does not allow any instance store volumes. None of the instance store volumes from the AMI will be accessible from the instance

Create key pair

Key pair name
Key pairs allow you to connect to your instance securely.

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](#)

Cancel **Create key pair**

EC2 > Instances

Instances (1/1) Info

Find Instance by attribute or tag (case-sensitive)		Last updated 1 minute ago	Connect	Instance state ▾	Actions ▾	Launch instances			
<input checked="" type="checkbox"/> Name	<input type="text" value="Cloud-DevOps..."/>	<input type="button" value="All states ▾"/>		<input checked="" type="checkbox"/> Running	<input checked="" type="checkbox"/> t2.micro	<input checked="" type="checkbox"/> Initializing	<input type="button" value="View alarms +"/>	us-east-1a	ec2-54-204-76-98

i-05958682dec50f2db (Cloud-DevOps_Webserver)

Details Status and alarms Monitoring Security Networking Storage Tags

Instance summary

Instance ID	i-05958682dec50f2db	Public IPv4 address	54.204.76.98 open address
IPv6 address	-	Instance state	Running
Hostname type	IP name: ip-172-31-92-245.ec2.internal	Private IP DNS name (IPv4 only)	ip-172-31-92-245.ec2.internal
Answer private resource DNS name	IPv4 (A)	Instance type	t2.micro
Auto-assigned IP address	54.204.76.98 [Public IP]	VPC ID	vpc-0c2072053840da3a4
IAM Role	-	Subnet ID	subnet-0c1c578a48279d865
		Private IPv4 addresses	172.31.92.245
		Public IPv4 DNS	ec2-54-204-76-98.compute-1.amazonaws.com open address
		Elastic IP addresses	-
		AWS Compute Optimizer finding	Opt-in to AWS Compute Optimizer for recommendations.
		Auto Scaling Group name	-

Step 2: Connecting via SSH

Once the instance is running, it's time to **SSH into** it. Think of SSH as your secure tunnel to your remote server. By connecting via SSH, you can issue commands and configure the server as if you were sitting in front of it.

```
ssh -i Test_key_Pair.pem ubuntu@your-ec2-ip
```

Here, the **Test_key_Pair.pem** is your SSH key that AWS provided. You'll need it to access your instance securely. Don't lose it!

```
PROBLEMS OUTPUT TERMINAL PORTS GITLENS AZURE DEBUG CONSOLE
ssh - .ssh + ... ×

drwxr-xr-x 7 encrypted staff 224 May 8 20:55 .
drwxr-xr-x@ 56 encrypted staff 1792 May 5 21:38 ..
-rw-r--r--@ 1 encrypted staff 2498 Apr 19 20:48 Hydrogen_key.pem
-rw-r--r--@ 1 encrypted staff 1678 May 8 20:45 Test_key_Pair.pem
-rw-r--r-- 1 root staff 60 Apr 18 01:27 config
-rw----- 1 encrypted staff 1960 May 8 20:55 known_hosts
-rw----- 1 encrypted staff 1220 May 8 20:54 known_hosts.old
○ (.venv) Encrypteds-MacBook-Pro: ssh encrypteds ssh -i Test_key_Pair.pem ubuntu@54.204.76.98
Welcome to Ubuntu 24.04.2 LTS (GNU/Linux 6.8.0-1024-aws x86_64)

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

System information as of Thu May 8 20:00:31 UTC 2025
System load: 0.0 Processes: 106
Usage of /: 25.3% of 6.71GB Users logged in: 0
Memory usage: 21% IPv4 address for enX0: 172.31.92.245
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-172-31-92-245:~$
```

Step 3: Installing Apache

Apache is one of the most popular **open-source web servers**. It's easy to configure, well-documented, and robust enough for most web applications.

Let's install Apache, enable it, and start it:

```
sudo apt update
```

```
sudo apt install -y apache2
```

```
sudo systemctl enable apache2
```

```
sudo systemctl start apache2
```

Now, Apache is up and running! Let's replace the default Apache landing page with something more personalized.

```
echo "<h1>🚀 CloudFormation Web Server</h1><p>This page was deployed automatically via CloudFormation.</p>" > /var/www/html/index.html
```

Why Apache?

Apache is a classic choice for web hosting. While there are newer web servers (like Nginx), Apache remains a solid, reliable option for hosting websites, making it a great tool for this lab.

Apache Installation and HTML Page

```
PROBLEMS OUTPUT TERMINAL PORTS GITLENS AZURE DEBUG CONSOLE
ubuntu@ip-172-31-92-245:~$ sudo apt update -y
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:4 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Packages [15.0 MB]
Get:5 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe Translation-en [5982 kB]
Get:7 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [820 kB]
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Components [3871 kB]
Get:9 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 c-n-f Metadata [301 kB]
Get:10 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Packages [269 kB]
Get:11 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse Translation-en [118 kB]
Get:12 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Components [35.0 kB]
Get:13 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 c-n-f Metadata [8328 B]
Get:14 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 Packages [1067 kB]
Get:15 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main Translation-en [229 kB]
Get:16 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 Components [161 kB]
Get:17 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 c-n-f Metadata [13.5 kB]
Get:18 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 Packages [1063 kB]
Get:19 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe Translation-en [269 kB]
Get:20 http://us-east-1.archive.ubuntu.com/ubuntu noble-updates/universe amd64 Components [376 kB]
Get:21 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 c-n-f Metadata [26.0 kB]
Get:22 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted amd64 Packages [1073 kB]
Get:23 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted Translation-en [221 kB]
Get:24 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted amd64 Components [212 kB]
Get:25 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted amd64 c-n-f Metadata [492 B]
Get:26 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 Packages [21.7 kB]
Get:27 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse Translation-en [4788 B]
Get:28 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 Components [940 B]
Get:29 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 c-n-f Metadata [592 B]
Get:30 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/main amd64 Packages [39.1 kB]
Get:31 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/main Translation-en [8676 B]
Get:32 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/main amd64 Components [7064 B]
Get:33 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/main amd64 c-n-f Metadata [272 B]
Get:34 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe amd64 Packages [27.0 kB]
Get:35 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe Translation-en [16.5 kB]
Get:36 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe amd64 Components [16.4 kB]
Get:37 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe amd64 c-n-f Metadata [1304 B]
Get:38 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/restricted amd64 Components [216 B]
Get:39 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/restricted amd64 c-n-f Metadata [116 B]
Get:40 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports/multiverse amd64 Components [212 B]
Get:41 http://us-east-1.archive.ubuntu.com/ubuntu noble-backports/multiverse amd64 c-n-f Metadata [116 B]
Get:42 http://security.ubuntu.com/ubuntu noble-security/main Translation-en [152 kB]
Get:43 http://security.ubuntu.com/ubuntu noble-security/main amd64 Components [21.6 kB]
Get:44 http://security.ubuntu.com/ubuntu noble-security/main amd64 c-n-f Metadata [7068 B]
Get:45 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Packages [836 kB]
Get:46 http://security.ubuntu.com/ubuntu noble-security/universe Translation-en [182 kB]
Get:47 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Components [52.3 kB]
Get:48 http://security.ubuntu.com/ubuntu noble-security/universe amd64 c-n-f Metadata [17.0 kB]
Get:49 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Packages [1041 kB]
Get:50 http://security.ubuntu.com/ubuntu noble-security/restricted Translation-en [215 kB]
Get:51 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Components [212 B]
Get:52 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 c-n-f Metadata [468 B]
Get:53 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Packages [17.7 kB]
Get:54 http://security.ubuntu.com/ubuntu noble-security/multiverse Translation-en [3792 B]
Get:55 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Components [208 B]
Get:56 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 c-n-f Metadata [380 B]
Fetched 34.0 MB in 9s (3821 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
82 packages can be upgraded. Run 'apt list --upgradable' to see them.
ubuntu@ip-172-31-92-245:~$
```

```
ubuntu@ip-172-31-92-245:~$ sudo apt install apache2 -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
apache2-bin apache2-data apache2-utils libaprutil64 libaprutil1-dbd-sqlite3 libaprutil1-ldap libaprutil1t64 liblLua5.4-0 ssl-cert
Suggested packages:
apache2-doc apache2-suexec-pristine | apache2-suexec-custom www-browser
The following NEW packages will be installed:
apache2 apache2-bin apache2-data apache2-utils libapr1t64 libaprutil1-dbd-sqlite3 libaprutil1-ldap libaprutil1t64 liblLua5.4-0 ssl-cert
0 upgraded, 10 newly installed, 0 to remove and 82 not upgraded.
Need to get 2084 kB of archives.
After this operation, 8094 kB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 libapr1t64 amd64 1.7.2-3.1ubuntu0.1 [108 kB]
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu/main amd64 libaprutil1t64 amd64 1.6.3-1.1ubuntu7 [91.9 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu/noble/main amd64 libaprutil1-dbd-sqlite3 amd64 1.6.3-1.1ubuntu7 [11.2 kB]
Get:4 http://us-east-1.ec2.archive.ubuntu.com/ubuntu/main amd64 libaprutil1-ldap amd64 1.6.3-1.1ubuntu7 [9116 B]
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu/noble/main amd64 liblLua5.4-0 amd64 5.4.6-3build2 [166 kB]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu/noble-updates/main amd64 apache2-bin amd64 2.4.58-1ubuntu8.6 [1330 kB]
Get:7 http://us-east-1.ec2.archive.ubuntu.com/ubuntu/noble-updates/main amd64 apache2-data all 2.4.58-1ubuntu8.6 [163 kB]
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu/noble-updates/main amd64 apache2-utils amd64 2.4.58-1ubuntu8.6 [97.2 kB]
Get:9 http://us-east-1.ec2.archive.ubuntu.com/ubuntu/noble-updates/main amd64 apache2 amd64 2.4.58-1ubuntu8.6 [90.2 kB]
Get:10 http://us-east-1.ec2.archive.ubuntu.com/ubuntu/noble/main amd64 ssl-cert all 1.1.2ubuntu1 [17.8 kB]
Fetched 2084 kB in 0s (38.2 MB/s)
Preconfiguring packages...
```

```
No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-92-245:~$ sudo systemctl enable apache2
Synchronizing state of apache2.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable apache2
ubuntu@ip-172-31-92-245:~$ sudo systemctl start apache2
ubuntu@ip-172-31-92-245:~$
```

Welcome to My EC2 Web Server!

This page is custom-made and manually configured on AWS.

Step 4: Setting Up SSL with Let's Encrypt

We need to ensure our website is secure. **SSL certificates** encrypt the data between the server and the client, which is crucial for user privacy and trust.

To get an SSL certificate for free, we'll use **Let's Encrypt**, a non-profit that provides free certificates.

First, we install **Certbot**, a tool for obtaining and managing Let's Encrypt certificates:

```
sudo apt install -y certbot python3-certbot-apache
```

Now, let's request an SSL certificate for our domain:

```
sudo certbot --apache -d yourdomain.com -d www.yourdomain.com
```

Certbot automatically configures Apache to use SSL. It's as simple as that!

Why Let's Encrypt?

Let's Encrypt is a game-changer for website security. It provides free SSL certificates and automates the installation and renewal process. This has become a best practice in the industry.

Certbot SSL Installation

PROBLEMS OUTPUT TERMINAL PORTS GITLENS AZURE DEBUG CONSOLE

ssh - ssh + ⚡ ⚡ ⚡ ⚡ ⚡ ⚡ ⚡ ⚡ ⚡

```
ubuntu@ip-172-31-92-245:~$ sudo apt install software-properties-common -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  python3-software-properties
The following packages will be upgraded:
  python3-software-properties software-properties-common
  2 upgraded, 0 newly installed, 0 to remove and 80 not upgraded.
Need to get 44.3 kB of archives.
After this operation, 1024 B of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 software-properties-common all 0.99.49.2 [14.4 kB]
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 python3-software-properties all 0.99.49.2 [29.8 kB]
Fetched 44.3 kB in 0s (2533 kB/s)
(Reading database ... 71283 files and directories currently installed.)
Preparing to unpack .../software-properties-common_0.99.49.2_all.deb ...
Unpacking software-properties-common (0.99.49.2) over (0.99.49.1) ...
Preparing to unpack .../python3-software-properties_0.99.49.2_all.deb ...
Unpacking python3-software-properties (0.99.49.2) over (0.99.49.1) ...
Setting up python3-software-properties (0.99.49.2) ...
Setting up software-properties-common (0.99.49.2) ...
Processing triggers for man-db (2.12.0-4build2) ...
Processing triggers for dbus (1.14.10-4ubuntu4.1) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-92-245:~$ sudo add-apt-repository universe -y
Adding components(s) 'universe' to all repositories.
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu noble-security InRelease
Reading package lists... Done
ubuntu@ip-172-31-92-245:~$ sudo apt update -y
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu noble-security InRelease
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
80 packages can be upgraded. Run 'apt list --upgradable' to see them.
ubuntu@ip-172-31-92-245:~$ sudo apt install certbot python3-certbot-apache -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  augeas-lenses libaugeas0 python3-acme python3-augeas python3-certbot python3-configargparse python3-icu python3-josepy python3-parsedatetime python3-rfc3339
Suggested packages:
  augeas-doc python-certbot-doc python3-certbot-nginx augeas-tools python-acme-doc python-certbot-apache-doc
The following NEW packages will be installed:
  augeas-lenses certbot libaugeas0 python3-acme python3-augeas python3-certbot python3-certbot-apache python3-configargparse python3-icu python3-josepy python3-parsedatetime
python3-rfc3339
0 upgraded, 12 newly installed, 0 to remove and 80 not upgraded.
```

```

ubuntu@ip-172-31-86-242:~$ sudo certbot --apache -d cloud-devops.xyz
Saving debug log to /var/log/letsencrypt/letsencrypt.log
Enter email address (used for urgent renewal and security notices)
(Enter 'c' to cancel):
Invalid email address: .

If you really want to skip this, you can run the client with
--register-unsafely-without-email but you will then be unable to receive notice
about impending expiration or revocation of your certificates or problems with
your Certbot installation that will lead to failure to renew.

Enter email address (used for urgent renewal and security notices)
(Enter 'c' to cancel): -----  

-----  

Please read the Terms of Service at
https://letsencrypt.org/documents/LE-SA-v1.5-February-24-2025.pdf. You must
agree in order to register with the ACME server. Do you agree?
-----  

(Y)es/(N)o: Y  

-----  

Would you be willing, once your first certificate is successfully issued, to
share your email address with the Electronic Frontier Foundation, a founding
partner of the Let's Encrypt project and the non-profit organization that
develops Certbot? We'd like to send you email about our work encrypting the web,
EFF news, campaigns, and ways to support digital freedom.
-----  

(Y)es/(N)o: N  

Account registered.  

Requesting a certificate for cloud-devops.xyz  

Successfully received certificate.  

Certificate is saved at: /etc/letsencrypt/live/cloud-devops.xyz/fullchain.pem  

Key is saved at: /etc/letsencrypt/live/cloud-devops.xyz/privkey.pem  

This certificate expires on 2025-08-06.  

These files will be updated when the certificate renews.  

Certbot has set up a scheduled task to automatically renew this certificate in the background.  

Deploying certificate  

Successfully deployed certificate for cloud-devops.xyz to /etc/apache2/sites-available/000-default-le-ssl.conf  

Congratulations! You have successfully enabled HTTPS on https://cloud-devops.xyz  

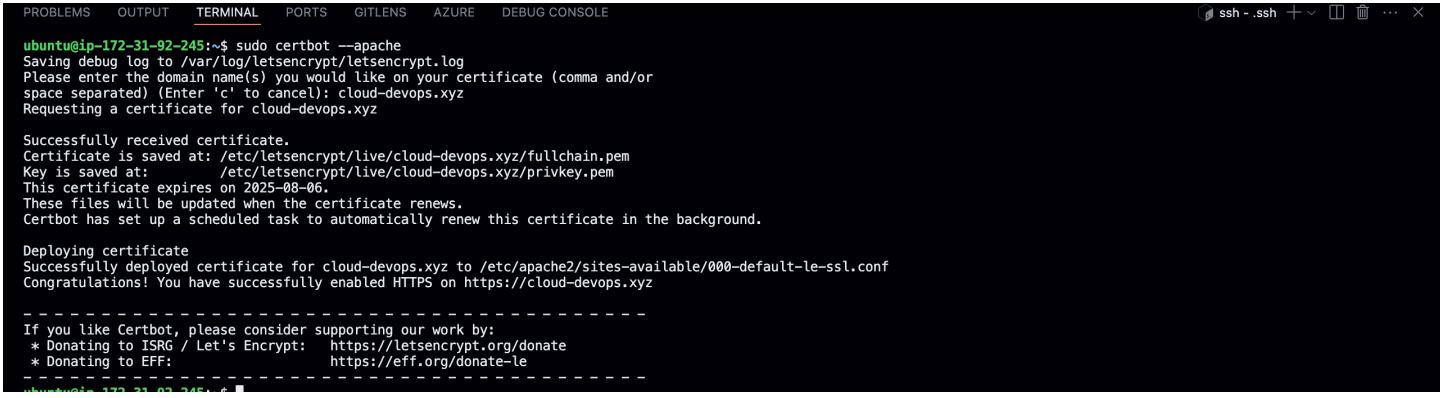
-----  

If you like Certbot, please consider supporting our work by:  

* Donating to ISRG / Let's Encrypt: https://letsencrypt.org/donate  

* Donating to EFF: https://eff.org/donate-le
-----
```

ubuntu@ip-172-31-86-242:~\$



```

PROBLEMS OUTPUT TERMINAL PORTS GITLENS AZURE DEBUG CONSOLE
ubuntu@ip-172-31-92-245:~$ sudo certbot --apache
Saving debug log to /var/log/letsencrypt/letsencrypt.log
Please enter the domain name(s) you would like on your certificate (comma and/or
space separated) (Enter 'c' to cancel): cloud-devops.xyz
Requesting a certificate for cloud-devops.xyz

Successfully received certificate.
Certificate is saved at: /etc/letsencrypt/live/cloud-devops.xyz/fullchain.pem
Key is saved at: /etc/letsencrypt/live/cloud-devops.xyz/privkey.pem
This certificate expires on 2025-08-06.
These files will be updated when the certificate renews.
Certbot has set up a scheduled task to automatically renew this certificate in the background.

Deploying certificate
Successfully deployed certificate for cloud-devops.xyz to /etc/apache2/sites-available/000-default-le-ssl.conf
Congratulations! You have successfully enabled HTTPS on https://cloud-devops.xyz

-----  

If you like Certbot, please consider supporting our work by:  

* Donating to ISRG / Let's Encrypt: https://letsencrypt.org/donate  

* Donating to EFF: https://eff.org/donate-le
-----
```

Step 5: Auto-Renewing SSL Certificates

We don't want to manually renew our certificates every 90 days. Instead, we set up a **cron job** to renew them automatically:

`sudo crontab -e`

Add the following line:

```
0 3 * * * certbot renew --quiet
```

Now, **Certbot** will automatically renew your SSL certificates every day at 3:00 AM.

Cron Job for SSL Renewal

```
ubuntu@ip-172-31-92-245:~$ sudo crontab -e
no crontab for root - using an empty one

Select an editor. To change later, run 'select-editor'.
1. /bin/nano           <---- easiest
2. /usr/bin/vim.basic
3. /usr/bin/vim.tiny
4. /bin/ed

Choose 1-4 [1]: 1
crontab: installing new crontab
ubuntu@ip-172-31-92-245:~$ wget http://prdownloads.sourceforge.net/webadmin/webmin_2.105_all.deb
--2025-05-08 20:31:22-- http://prdownloads.sourceforge.net/webadmin/webmin_2.105_all.deb
Resolving prdownloads.sourceforge.net (prdownloads.sourceforge.net)... 104.18.12.149, 104.18.13.149, 2606:4700::6812:d95, ...
Connecting to prdownloads.sourceforge.net (prdownloads.sourceforge.net)|104.18.12.149|:80... connected.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: http://downloads.sourceforge.net/project/webadmin/webmin/2.105/webmin_2.105_all.deb [following]
--2025-05-08 20:31:22-- http://downloads.sourceforge.net/project/webadmin/webmin/2.105/webmin_2.105_all.deb
Resolving downloads.sourceforge.net (downloads.sourceforge.net)... 104.18.12.149, 104.18.13.149, 2606:4700::6812:c95, ...
Reusing existing connection to prdownloads.sourceforge.net:80.
HTTP request sent, awaiting response... 302 Found
Location: http://netactuate.dl.sourceforge.net/project/webadmin/webmin/2.105/webmin_2.105_all.deb?viasf=1 [following]
--2025-05-08 20:31:22-- http://netactuate.dl.sourceforge.net/project/webadmin/webmin/2.105/webmin_2.105_all.deb?viasf=1
Resolving netactuate.dl.sourceforge.net (netactuate.dl.sourceforge.net)... 104.225.3.66
Connecting to netactuate.dl.sourceforge.net (netactuate.dl.sourceforge.net)|104.225.3.66|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 33671982 (32M) [application/octet-stream]
Saving to: 'webmin_2.105_all.deb'

webmin_2.105_all.deb          100%[=====] 32.11M 57.5MB/s   in 0.6s

2025-05-08 20:31:23 (57.5 MB/s) - 'webmin_2.105_all.deb' saved [33671982/33671982]

ubuntu@ip-172-31-92-245:~$ sudo apt install ./webmin_2.105_all.deb
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Note, selecting 'webmin' instead of './webmin_2.105_all.deb'
The following additional packages will be installed:
bz2 libalgorithm-c3-perl libauthen-pam-perl libb-hooks-endofscope-perl libb-hooks-op-check-perl libclass-c3-perl libclass-c3-xs-perl libclass-data-inheritable-perl
libclass-inspector-perl libclass-method-modifiers-perl libclass-singleton-perl libclass-xsaccessor-perl libdata-optlist-perl libdatetime-locale-perl libdatetime-perl
libdatetime-timezone-perl libdevcallchecker-perl libdevcaller-perl libdevlexalias-perl libdevstacktrace-perl libdynaloader-functions-perl libencode-detect-perl
libeval-closure-perl libexception-class-perl libfile-sharedir-perl libio-pty-perl libmodule-implementation-perl libmodule-runtime-perl libmro-compat-perl
libnamespace-autoclean-perl libnamespacetospace-clean-perl libnet-ssleay-perl libpackage-stash-perl libpackage-stash-xs-perl libpadwalker-perl libparams-classify-perl
libparams-util-perl libparams-validationcompiler-perl libreadonly-perl libref-util-perl libref-util-xs-perl librole-tiny-perl libspecio-perl libsub-exporter-perl
libsub-exporter-progressive-perl libsub-identify-perl libsub-install-perl libsub-name-perl libsub-quote-perl libtry-tiny-perl libvariable-magic-perl libxstring-perl lynx
lynx-common mailcap perl-openssl-defaults unzip
Suggested packages:
bz2-doc libscalar-number-perl libtest-fatal-perl debhelper zip
The following NEW packages will be installed:
bz2 libalgorithm-c3-perl libauthen-pam-perl libb-hooks-endofscope-perl libb-hooks-op-check-perl libclass-c3-perl libclass-c3-xs-perl libclass-data-inheritable-perl
libclass-inspector-perl libclass-method-modifiers-perl libclass-singleton-perl libclass-xsaccessor-perl libdata-optlist-perl libdatetime-locale-perl libdatetime-perl
libdatetime-timezone-perl libdevcallchecker-perl libdevcaller-perl libdevlexalias-perl libdevstacktrace-perl libdynaloader-functions-perl libencode-detect-perl
libeval-closure-perl libexception-class-perl libfile-sharedir-perl libio-pty-perl libmodule-implementation-perl libmodule-runtime-perl libmro-compat-perl
libnamespace-autoclean-perl libnamespacetospace-clean-perl libnet-ssleay-perl libpackage-stash-perl libpackage-stash-xs-perl libpadwalker-perl libparams-classify-perl
libparams-util-perl libparams-validationcompiler-perl libreadonly-perl libref-util-perl libref-util-xs-perl librole-tiny-perl libspecio-perl libsub-exporter-perl
libsub-exporter-progressive-perl libsub-identify-perl libsub-install-perl libsub-name-perl libsub-quote-perl libtry-tiny-perl libvariable-magic-perl libxstring-perl lynx
lynx-common mailcap perl-openssl-defaults unzip webmin
0 upgraded, 58 newly installed, 0 to remove and 80 not upgraded.
Need to get 6433 kB/40.1 MB of archives.
```

```
PROBLEMS OUTPUT TERMINAL PORTS GITLENS AZURE DEBUG CONSOLE

GNU nano 7.2
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h dom mon dow   command
0 0 * * * /usr/bin/certbot renew --quiet
```

Webmin Configuration

Before transitioning into automation, I took an extra step to enhance server management via a browser interface by configuring **Webmin**. This tool provides a rich GUI for system

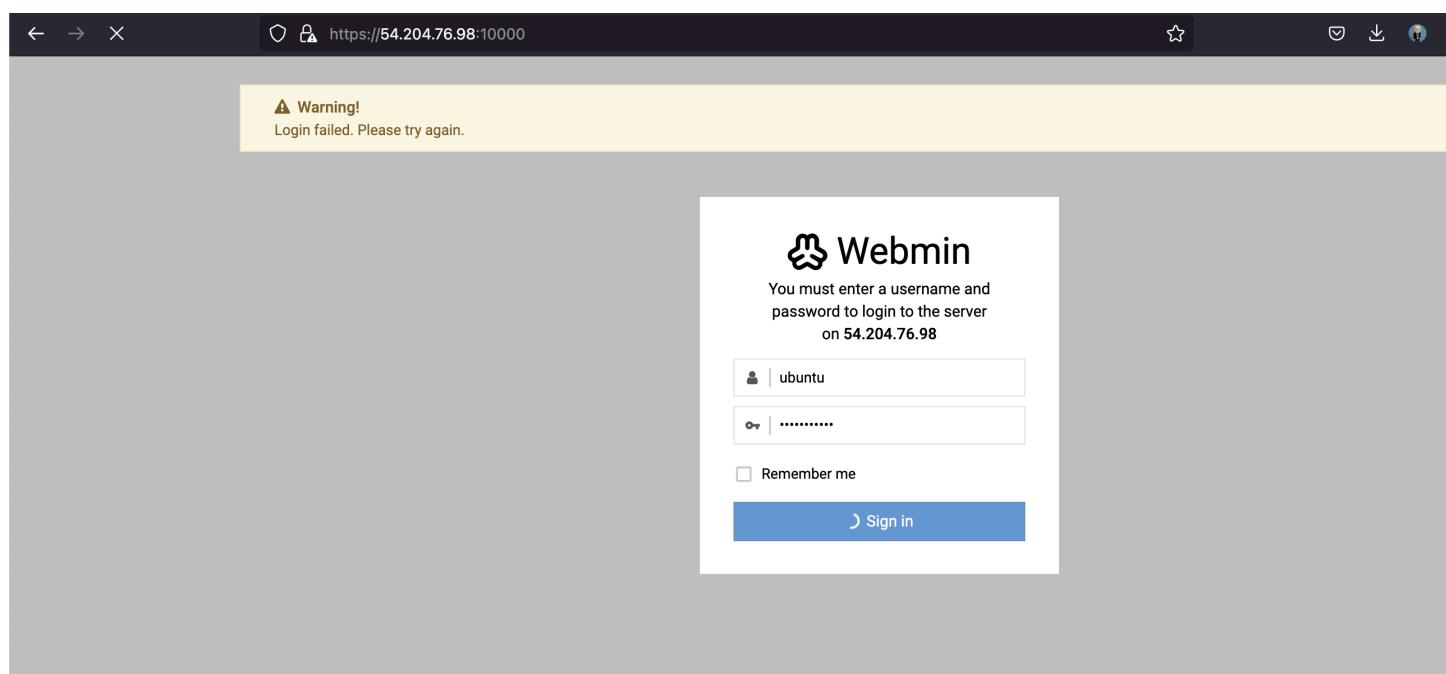
administration tasks like user management, package updates, and more.

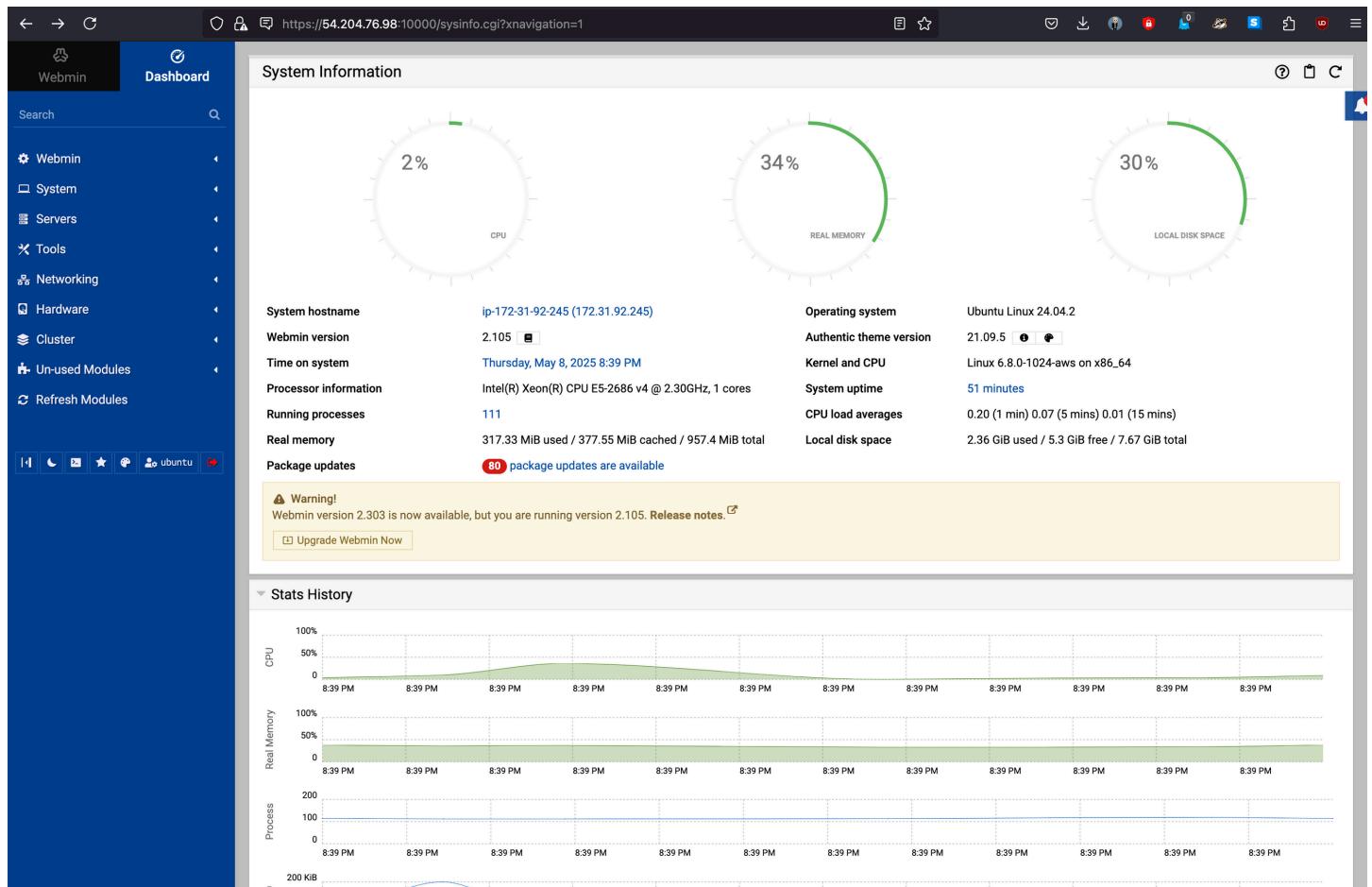
🛠 Steps Taken:

1. **Downloaded and installed Webmin** on the EC2 instance:
2. wget <https://www.webmin.com/download/deb/webmin-current.deb>
3. sudo dpkg -i webmin-current.deb
4. sudo apt --fix-broken install
5. **Resolved permission error** during the .deb install:
 - **Error encountered:**
 - N: Download is performed unsandboxed as root as file '/home/ubuntu/webmin_2.105_all.deb' couldn't be accessed by user '_apt'.
 - **Fix:** We manually moved the .deb file and reran the install with the proper privileges.
6. **Opened port 10000** for Webmin access:
7. **Accessed Webmin via browser** at:
8. https://your-server-ip:10000
9. **Logged in with:**
 - **Username:** ubuntu
 - **Password:** (set manually during the initial setup)

This gave me an alternative, secure way to manage the instance before proceeding with the CloudFormation setup in Part 2.

```
ubuntu@ip-172-31-92-245:~$ sudo passwd ubuntu
New password:
Retype new password:
passwd: password updated successfully
```





The screenshot shows the AWS CloudWatch Instances page with the following details:

- Instances (1/1) Info:** Last updated 1 minute ago.
- Filter:** Instance ID = i-05958682dec50f2db
- Actions:** Connect, Instance state, Actions, Launch instances.
- Table Headers:** Instance ID, Name, Instance state, Instance type, Status check, Alarm status, Availability Zone, Public IPv4 IP.
- Table Data:** Cloud-DevOps... (i-05958682dec50f2db), Running, t2.micro, 2/2 checks passed, us-east-1a, ec2-54-204-7...

Instance Summary (i-05958682dec50f2db):

- Details:** Instance ID: i-05958682dec50f2db, IPv6 address: -, Hostname type: IP name: ip-172-31-92-245.ec2...
- Status and alarms:** No status or alarms shown.
- Terminate (delete) instance dialog:**
 - Warning:** On an EBS-backed instance, the default action is for the root EBS volume to be deleted when the instance is terminated. Storage on any local drives will be lost.
 - Question:** Are you sure you want to terminate these instances?
 - Fields:** Instance ID: i-05958682dec50f2db (Cloud-DevOps_Webserver), Termination protection: Disabled.
 - Note:** To confirm that you want to delete the instances, choose the terminate button below. Instances with termination protection enabled will not be terminated. Terminating the instance cannot be undone.
 - Buttons:** Cancel, Terminate (delete).

Part 2: Automating the Process with CloudFormation

Now that we've manually set up our server, let's make it repeatable. **AWS CloudFormation** allows us to automate the entire infrastructure deployment.

Step 1: CloudFormation Template Creation

CloudFormation is like the **blueprint** of your AWS resources. It allows you to define and provision infrastructure using code. Here's a basic YAML template that launches an EC2 instance, sets up Apache, and configures security groups.

`AWSTemplateFormatVersion: '2010-09-09'`

`Description: > Launch an EC2 instance, install Apache web server, allow traffic on ports 80, 443, 22, and 10000 (Webmin), and display a custom HTML page.`

This template defines:

- An **EC2 instance** running **Ubuntu**.
- A **Security Group** allowing inbound traffic on the necessary ports.
- A **User Data** script that installs Apache and serves our custom page.

Why CloudFormation?

CloudFormation is perfect for **Infrastructure as Code (IaC)**. It allows you to define your infrastructure in YAML or JSON and deploy it with the click of a button, ensuring consistency across multiple environments.

CloudFormation Template Code

The screenshot shows the AWS CloudFormation console interface. On the left, there's a sidebar with navigation links for CloudFormation, Stacks, Drifts, StackSets, Exports, Infrastructure Composer, Hooks, Registry, Public extensions, Activated extensions, Publisher, Spotlight, and Feedback. The main area shows a list of stacks under 'Stacks (1)' with one stack named 'EC2-WebServer-Stack' shown in detail. The 'Template' tab is selected in the top navigation bar. The template content is as follows:

```
AWSTemplateFormatVersion: '2010-09-09'
Description: EC2 Web Server Setup (Manual Part 2 - CloudFormation)

Resources:
  WebServerSecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: Enable HTTP and HTTPS access
      SecurityGroupIngress:
        - IpProtocol: tcp
          FromPort: 80
          ToPort: 80
          CidrIp: 0.0.0.0/0
        - IpProtocol: tcp
          FromPort: 443
          ToPort: 443
          CidrIp: 0.0.0.0/0
        - IpProtocol: tcp
          FromPort: 22
          ToPort: 22
          CidrIp: 0.0.0.0/0
        - IpProtocol: tcp
          FromPort: 10000
          ToPort: 10000
          CidrIp: 0.0.0.0/0

  EC2Instance:
    Type: AWS::EC2::Instance
    Properties:
      InstanceType: t2.micro
      KeyName: Test_key_Pair
      ImageId: ami-084568d4383264d4 # Replace with region-specific Ubuntu 22.04 AMI
      SecurityGroups:
        - !Ref WebServerSecurityGroup
      UserData:
        Fn::Base64: !Sub |
          #!/bin/bash
          apt update -y
          apt install -y apache2
          systemctl start apache2
          systemctl enable apache2
          echo "<h1> CloudFormation Web Server</h1><p>This page was deployed automatically via CloudFormation.</p>" > /var/www/html/index.html

Outputs:
  InstancePublicIP:
    Description: Public IP of the newly created EC2 instance
    Value: !GetAtt EC2Instance.PublicIp
```

Step 2: Deploying the Stack

After writing the template, we deploy it as a **CloudFormation Stack**:

1. Go to **CloudFormation** and click "**Create Stack**".
2. Upload or paste your CloudFormation YAML template.
3. Enter the **KeyName** parameter (your EC2 key pair name) and launch the stack.

CloudFormation will then take care of everything, from creating the EC2 instance to setting up Apache.

CloudFormation Stack Deployment

The screenshot shows the AWS CloudFormation 'Create stack' wizard. On the left, there's a sidebar with navigation links like 'CloudFormation', 'Stacks', 'Exports', 'Infrastructure Composer', 'Hooks', 'Registry', 'Spotlight', and 'Feedback'. The main area is titled 'Create stack' and 'Prerequisite - Prepare template'. It shows four steps: 'Step 1 Create stack' (selected), 'Step 2 Specify stack details', 'Step 3 Configure stack options', and 'Step 4 Review and create'. Under 'Prerequisite - Prepare template', it says 'You can also create a template by scanning your existing resources in the IaC generator'. There are three options: 'Choose an existing template' (selected), 'Build from Infrastructure Composer', and 'Sync from Git'. Below that, there's a section for 'Specify template' with 'Template source' options: 'Amazon S3 URL' (disabled), 'Upload a template file' (selected), and 'Sync from Git'. A 'Choose file' button is shown, along with a note about JSON or YAML formatted files. At the bottom, it shows the S3 URL: https://s3.us-east-1.amazonaws.com/cf-templates-1aioewtvieuzv-us-east-1/2025-05-08T212901.171Zvc9-Ec2_Webserver.yml and a 'View in Infrastructure Composer' link. At the bottom right are 'Cancel' and 'Next' buttons.

CloudFormation > Stacks > Create stack

CloudFormation

- Stacks
- StackSets
- Exports

Infrastructure Composer
IaC generator

Hooks overview
Hooks

▼ Registry

- Public extensions
- Activated extensions
- Publisher

Spotlight

Feedback

Step 1 Create stack

Step 2 **Specify stack details**

Step 3 Configure stack options

Step 4 Review and create

Specify stack details

Provide a stack name

Stack name

EC2-WebServer-Stack

Stack name must contain only letters (a-z, A-Z), numbers (0-9), and hyphens (-) and start with a letter. Max 128 characters. Character count: 19/128.

Parameters

Parameters are defined in your template and allow you to input custom values when you create or update a stack.

No parameters

There are no parameters defined in your template

Cancel Previous Next

CloudFormation > Stacks > Create stack

CloudFormation

- Stacks
- StackSets
- Exports

Infrastructure Composer
IaC generator

Hooks overview
Hooks

▼ Registry

- Public extensions
- Activated extensions
- Publisher

Spotlight

Feedback

Step 1 Create stack

Step 2 Specify stack details

Step 3 Configure stack options

Step 4 **Review and create**

Review and create

Step 1: Specify template

Edit

Prerequisite - Prepare template

Template

Template is ready

Template

Template URL

https://s3.us-east-1.amazonaws.com/cf-templates-1aioewtvieqv-us-east-1/2025-05-08T212901.171Zvc9-Ec2_Webserver.yml

Stack description

EC2 Web Server Setup (Manual Part 2 - CloudFormation)

Step 2: Specify stack details

Edit

Provide a stack name

Stack name

EC2-WebServer-Stack

Parameters

Search

Key ▲ | Value ▼

No parameters

There are no parameters defined in your template

Outputs (2)

Key	Value	Description	Export name
InstancePublicIP	3.93.15.122	Public IP of the newly created EC2 instance	-
WebsiteURL	http://3.93.15.122	Access your site at	-

Resources (2)

Logical ID	Physical ID	Type	Status	Module
EC2Instance	i-0fc6e3f03593c752c	AWS::EC2::Instance	CREATE_COMPLETE	-
WebServerSecurityGroup	EC2-WebServer-Stack-WebServerSecurityGroup-Kcn0UDTowZTP	AWS::EC2::SecurityGroup	CREATE_COMPLETE	-

Part 3: Connect Domain and Install SSL

Once our server is up and running, we connect our **custom domain** and **install the SSL certificate**.

Step 1: Update DNS Settings

We use **Hostinger**, **GoDaddy**, or any other domain provider to create an **A record** that points to our EC2 instance's IP address.

Domain Registrar DNS Settings

[Home](#)

[Websites](#)

[Domains](#)

[Domain portfolio](#)

[Get a New Domain](#)

[Transfers](#)

[Emails](#)

[VPS](#)

[Dark web monitoring](#)

[Billing](#)

[Find new domain](#) [Generate domain using AI](#)

Type your desired domain name into the domain checker search bar and find out if it's available instantly!

[Search](#)

.com US\$ 4.99	.net US\$ 11.99	.io US\$ 31.99	.org US\$ 7.99	.online US\$ 1.99	.shop US\$ 0.99
-----------------------------------	------------------------------------	-----------------------------------	-----------------------------------	--------------------------------------	------------------------------------

cloud-devops.xyz is already taken.

Explore similar options below.

[Home](#)

[Websites](#)

[Domains](#)

[Domain portfolio](#)

[Get a New Domain](#)

[Transfers](#)

[Emails](#)

[VPS](#)

[Dark web monitoring](#)

Domain portfolio [- Domain portfolio](#) [+ Add new domain](#)

Protect your internet identity

[cloud-devops.org](#) or [See more options](#)

US\$ 15.99 **US\$ 7.99/1st yr** [Get now](#)

<input type="checkbox"/> Domain name ↓	Status ↓	Expiration date ↓	Auto-renewal
<input type="checkbox"/> cloud-devops.xyz		-	-

[Setup](#)

[Main menu](#)

[Domain Overview](#)

[DNS / Nameservers](#)

[Domain Ownership](#)

DNS / Nameservers [- Domain portfolio - cloud-devops.xyz - DNS / Nameservers](#)

[DNS records](#) [Child nameservers](#) [DNSSEC](#) [Forwarding](#) [DNS history](#)

Your domain will be online soon
It can take anywhere from 15 minutes to 24 hours for domains to start working after DNS changes. Once this happens, your website will be accessible via [cloud-devops.xyz](#). [More details](#)

Nameservers
Nameservers handle internet requests for your domain. You can use Hostinger nameservers or use custom nameservers to point to other hosting provider.

ns1.dns-parking.com
ns2.dns-parking.com

[Change Nameservers](#)

Manage DNS records
These records define how your domain behaves. Common uses include pointing your domain at web servers or configuring email delivery for your domain.

Type A	Name @	Points to 54.204.76.98	TTL 50	Add Record
---------------------------	-----------	---------------------------	-----------	----------------------------

```
encrypted@Encrypteds-MacBook-Pro GitHub % nslookup cloud-devops.xyz
Server:      103.86.96.100
Address:     103.86.96.100#53

Non-authoritative answer:
Name:   cloud-devops.xyz
Address: 54.204.76.98

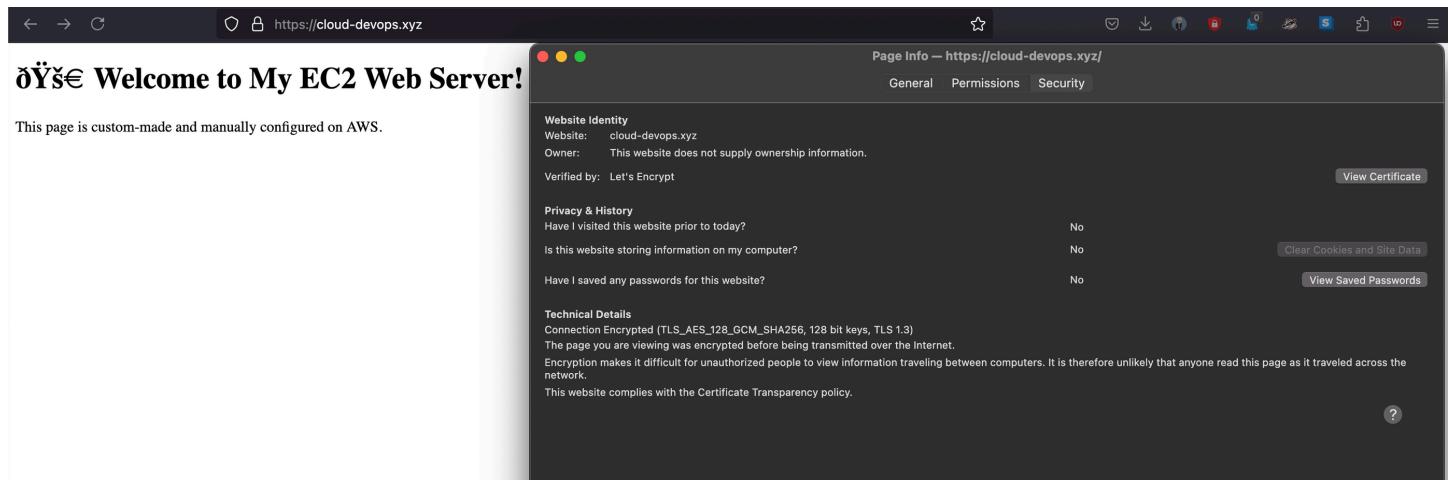
encrypted@Encrypteds-MacBook-Pro GitHub %
```

Step 2: Install SSL Using Certbot

We SSH into our EC2 instance and run the same **Certbot** commands as before to install SSL:

```
sudo certbot --apache -d yourdomain.com -d www.yourdomain.com
```

Now, our website is secure, and we have **HTTPS** running.



Conclusion: From Manual to Automated Deployment

With the manual setup, we learned how to configure and secure a web server step by step. Then, we took that knowledge and automated it using **AWS CloudFormation**, simplifying future deployments. By connecting a custom domain and adding SSL, we've built a **secure, scalable web server** hosted on AWS.

Next Steps

- Use this template as a base for scaling and adding more services.
- Explore integrating this setup with **AWS Elastic Load Balancers** or **Auto Scaling** for high availability.

You're absolutely right! Including the errors we encountered and how we resolved them adds a layer of authenticity to the tutorial. It provides real-world context and demonstrates problem-solving, which is invaluable for readers, especially those who might run into the same issues.

Troubleshooting Common Errors During Setup

No journey is without its challenges! During the manual setup, we encountered a few issues. Here's a breakdown of the errors, what caused them, and how we fixed them.

Error 1: "N: Download is performed unsandboxed as root as file

'/home/ubuntu/webmin_2.105_all.deb' couldn't be accessed by user '_apt'. - pkgAcquire::Run (13: Permission denied)"

What Happened?

While trying to install **Webmin** using the .deb package on our EC2 instance, we encountered the following error:

N: Download is performed unsandboxed as root as file '/home/ubuntu/webmin_2.105_all.deb'
couldn't be accessed by user '_apt'. - pkgAcquire::Run (13: Permission denied)

Cause of the Issue:

This error occurs when the **APT package manager** (which handles package installation on Ubuntu) is run as root and doesn't have proper permissions for downloading the .deb package. The system is essentially trying to download the file without proper access rights.

How We Resolved It:

To resolve the issue, we had to fix the permissions of the directory where the file was being downloaded. Here's the fix we applied:

1. Remove the partially downloaded file:

First, we cleaned up the directory and removed any partially downloaded files:

2. sudo rm -f /home/ubuntu/webmin_2.105_all.deb

3. Install the package again with proper permissions:

We tried running the installation again, but with elevated privileges to ensure the download happened correctly:

4. sudo dpkg -i webmin_2.105_all.deb

Lesson Learned:

When installing packages via dpkg, it's important to ensure that the user running the command has the right permissions to download and install files. Running as root (without the proper environment) can sometimes interfere with the process, so checking permissions and file locations is critical.

Terminal Output with the Error and Resolution

```

ubuntu@ip-172-31-92-245:~$ wget http://prdownloads.sourceforge.net/webadmin/webmin_2.105_all.deb
--2025-05-08 20:31:22-- http://prdownloads.sourceforge.net/webadmin/webmin_2.105_all.deb
Resolving prdownloads.sourceforge.net (prdownloads.sourceforge.net)... 104.18.12.149, 104.18.13.149, 2606:4700::6812:d95, ...
Connecting to prdownloads.sourceforge.net (prdownloads.sourceforge.net)|104.18.12.149|:80... connected.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: http://downloads.sourceforge.net/project/webadmin/webmin/2.105/webmin_2.105_all.deb [following]
--2025-05-08 20:31:22-- http://downloads.sourceforge.net/project/webadmin/webmin/2.105/webmin_2.105_all.deb
Resolving downloads.sourceforge.net (downloads.sourceforge.net)... 104.18.12.149, 104.18.13.149, 2606:4700::6812:c95, ...
Reusing existing connection to prdownloads.sourceforge.net:80.
HTTP request sent, awaiting response... 200 OK
Length: 33671982 (32M) [application/octet-stream]
Saving to: 'webmin_2.105_all.deb'

webmin_2.105_all.deb          100%[=====] 32.11M 57.5MB/s   in 0.6s

2025-05-08 20:31:23 (57.5 MB/s) - 'webmin_2.105_all.deb' saved [33671982/33671982]

ubuntu@ip-172-31-92-245:~$ sudo apt install ./webmin_2.105_all.deb
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Note, selecting 'webmin' instead of './webmin_2.105_all.deb'
The following additional packages will be installed:
bz2 libalgorithm-c3-perl libauthen-pam-perl libb-hooks-endofscope-perl libb-hooks-op-check-perl libclass-c3-perl libclass-c3-xs-perl libclass-data-inheritable-perl
libclass-inspector-perl libclass-method-modifiers-perl libclass-singleton-perl libclass-xsaccessor-perl libdata-optlist-perl libdatETIME-locale-perl libdatETIME-perl
libdatETIME-timezone-perl libdevel-callchecker-perl libdevel-caller-perl libdevel-lexalias-perl libdevel-stacktrace-perl libdynaloader-functions-perl libencode-detect-perl
libeval-closure-perl libexception-class-perl libfile-sharedir-perl libio-pty-perl libmodule-implementation-perl libmodule-runtime-perl libmro-compat-perl
libnamespace-autoclean-perl libnamespace-clean-perl libnet-ssleay-perl libpackage-stash-perl libpadwalker-perl libparams-classify-perl
libparams-util-perl libparams-validationcompiler-perl libreadonly-perl libref-util-perl libref-util-xs-perl librole-tiny-perl libspecio-perl libsub-exporter-perl
libsub-exporter-progressive-perl libsub-identify-perl libsub-install-perl libsub-name-perl libsub-quote-perl libtry-tiny-perl libvariable-magic-perl libxstring-perl lynx
lynx-common mailcap perl-openssl-defaults unzip
Suggested packages:
bz2-doc libscalar-number-perl libtest-fatal-perl debhelper zip
The following NEW packages will be installed:
bz2 libalgorithm-c3-perl libauthen-pam-perl libb-hooks-endofscope-perl libb-hooks-op-check-perl libclass-c3-perl libclass-c3-xs-perl libclass-data-inheritable-perl
libclass-inspector-perl libclass-method-modifiers-perl libclass-singleton-perl libclass-xsaccessor-perl libdata-optlist-perl libdatETIME-locale-perl libdatETIME-perl
libdevel-callchecker-perl libdevel-caller-perl libdevel-lexalias-perl libdevel-stacktrace-perl libdynaloader-functions-perl libencode-detect-perl
libeval-closure-perl libexception-class-perl libfile-sharedir-perl libio-pty-perl libmodule-implementation-perl libmodule-runtime-perl libmro-compat-perl
libnamespace-autoclean-perl libnamespace-clean-perl libnet-ssleay-perl libpackage-stash-perl libpadwalker-perl libparams-classify-perl
libparams-util-perl libparams-validationcompiler-perl libreadonly-perl libref-util-perl libref-util-xs-perl librole-tiny-perl libspecio-perl libsub-exporter-perl
libsub-exporter-progressive-perl libsub-identify-perl libsub-install-perl libsub-name-perl libsub-quote-perl libtry-tiny-perl libvariable-magic-perl libxstring-perl lynx
lynx-common mailcap perl-openssl-defaults unzip webmin
0 upgraded, 58 newly installed, 0 to remove and 80 not upgraded.
Need to get 6433 kB/40.1 MB of archives.

```

```

Scanning Linux images...
Running kernel seems to be up-to-date.
No services need to be restarted.
No containers need to be restarted.
No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
N: Download is performed unsandboxed as root as file '/home/ubuntu/webmin_2.105_all.deb' couldn't be accessed by user '_apt'. - pkgAcquire::Run (13: Permission denied)
ubuntu@ip-172-31-92-245:~$ sudo mv ~/webmin_2.105_all.deb /tmp/
ubuntu@ip-172-31-92-245:~$ sudo apt install /tmp/webmin_2.105_all.deb
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Note, selecting 'webmin' instead of '/tmp/webmin_2.105_all.deb'
webmin is already the newest version (2.105).
0 upgraded, 0 newly installed, 0 to remove and 80 not upgraded.
ubuntu@ip-172-31-92-245:~$ 

```

Error 2: "403 Forbidden" After Installing Apache

What Happened?

After setting up **Apache** and configuring the default page, we tried accessing the server via <http://your-ec2-ip>, but received a **403 Forbidden** error.

Cause of the Issue:

This error typically occurs when Apache is not configured to allow public access to the directory containing the web page, or if file permissions are not set correctly.

How We Resolved It:

Here's how we fixed the issue:

1. Check the directory permissions:

We ensured that the directory `/var/www/html` where Apache serves files had the correct permissions:

2. `sudo chown -R www-data:www-data /var/www/html`
3. `sudo chmod -R 755 /var/www/html`

4. Check the Apache configuration:

We confirmed that Apache's configuration allowed access to the public directory. We edited the `000-default.conf` file to make sure it allowed traffic:

5. sudo nano /etc/apache2/sites-available/000-default.conf
6. Then, we checked that the following lines were set up correctly:
 7. <Directory /var/www/html>
 8. Options Indexes FollowSymLinks
 9. AllowOverride All
 10. Require all granted
11. </Directory>

12. **Restart Apache:**

After modifying the configurations, we restarted Apache to apply the changes:

13. sudo systemctl restart apache2

Lesson Learned:

Permissions and Apache configurations are critical when setting up a web server. Always check both the filesystem permissions and the Apache configuration to ensure everything is set up correctly.

Error 3: "SSL Certificate Installation Failed"

What Happened?

After configuring **SSL with Let's Encrypt**, we encountered a **certificate installation failure**.

Cause of the Issue:

This happened because the **HTTP-01 challenge** used by Let's Encrypt requires the server to be accessible over **HTTP** (port 80). However, since we didn't have port 80 open in our security group at the time, the certificate installation failed.

How We Resolved It:

To fix this, we:

1. **Checked our Security Group:**

We confirmed that **port 80** was open in the EC2 security group.

2. **Re-ran the Certbot Command:**

After ensuring port 80 was open, we re-ran the Certbot command to obtain the SSL certificate:

3. sudo certbot --apache -d yourdomain.com

4. **Verified the SSL Certificate:**

After Certbot successfully issued the certificate, we verified it by visiting the site via <https://yourdomain.com>.

Lesson Learned:

When using Let's Encrypt, it's essential to have port 80 open temporarily for the HTTP-01 challenge. Once the certificate is installed, we can secure the server with HTTPS, and traffic on port 80 can be redirected to HTTPS.

Conclusion: Troubleshooting Made Easy

Every server setup comes with its fair share of challenges, but with persistence and careful troubleshooting, the solutions are always within reach. By addressing the errors above, we were able to learn valuable lessons about permissions, Apache configurations, and SSL setup.

In the next part of this tutorial, we'll see how **CloudFormation** can streamline the process and eliminate the need for manual intervention. However, having a solid understanding of the manual steps ensures that when something goes wrong, we know exactly how to troubleshoot and resolve it.

What We've Learned

1. **Apache setup:** Configuring Apache correctly is vital, from ensuring permissions to modifying the configuration files.
2. **SSL certificates with Let's Encrypt:** Let's Encrypt makes SSL accessible and free, but we must ensure the server is properly accessible during the certificate issuance process.
3. **Troubleshooting:** Every error provides a learning opportunity. Fixing issues with the right approach not only resolves problems but also enhances your understanding of the tools.

Final Thoughts:

I hope this blog has made the process of setting up a web server on **AWS EC2** more approachable. Whether you're just starting or have some experience under your belt, there's always something new to learn in the world of cloud computing. Stay tuned for more tutorials as we continue to automate and scale our cloud infrastructure!