Diego Gonzalez Fernandez

ÍNDICE

Lo primero que he hecho ha sido preparar mi máquina local para gestionar la seguridad. He creado el directorio .ssh en mi sistema Ubuntu local para almacenar las claves de forma organizada.

```
alumno17@A6Alumno17:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$ mkdir -p ~/.ssh
```

He ejecutado el comando chmod 700 ~/.ssh para establecer los permisos de seguridad correctos en mi carpeta de claves. Con esta acción, he restringido el acceso al directorio .ssh para que únicamente mi usuario tenga permisos de lectura, escritura y ejecución, garantizando así que el lugar donde guardaré mis credenciales esté protegido contra accesos externos.

```
alumno17@A6Alumno17:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$ chmod 700 ~/.ssh
```

Aquí he copiado mi llave privada (labsuser.pem), que tenía descargada en Windows, a este directorio seguro dentro de mi WSL para poder utilizarla.

```
alumno17@A6Alumno17:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$ cp /mnt/c/Users/Alumno.Desktop-DI5KTUG/Downlo
ads/labsuser.pem ~/.ssh/
```

Una vez tenía la llave lista, he lanzado el comando SSH apuntando a la IP pública de mi instancia (34.204.186.237). Al hacerlo, he logrado establecer la conexión cifrada y mi terminal ha cambiado: he dejado de estar en mi ordenador local para pasar a tener el control total dentro del servidor de AWS.

```
alumno17@A6Alumno17:/mnt/c/Users/Alumno.DESKTOP-DI5KTUG$ ssh -i ~/.ssh/labsuser.pem ubuntu@34.204.186.
237
The authenticity of host '34.204.186.237 (34.204.186.237)' can't be established.
ED25519 key fingerprint is SHA256:pSK2AFo5KekqtNAWFvR1IvFeQ2a7OOP6cmDbW726USg.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '34.204.186.237' (ED25519) to the list of known hosts.
Welcome to Ubuntu 24.04.3 LTS (GNU/Linux 6.14.0-1015-aws x86_64)

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

 System information as of Fri Nov 28 08:30:58 UTC 2025

  System load:  0.08              Temperature:           -273.1 C
  Usage of /:   26.2% of 6.71GB   Processes:             112
  Memory usage: 24%               Users logged in:       1
  Swap usage:   0%                IPv4 address for ens5: 172.31.65.230


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: Fri Nov 28 08:15:39 2025 from 18.206.107.28
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.
```

Ya dentro de la instancia EC2. Primero he actualizado los repositorios del sistema (apt update) para asegurarme de bajar las últimas versiones.

```
ubuntu@ip-172-31-65-230:~$ sudo apt update && sudo apt upgrade -y
```

Después, he instalado el servidor web Apache con sudo apt install apache2.

```
ubuntu@ip-172-31-65-230:~$ sudo apt install apache2 -y
```

Para alinear el servidor con las reglas de seguridad definidas en AWS (Security Groups), edité el archivo de configuración global /etc/apache2/ports.conf.

```
ubuntu@ip-172-31-65-230:~$ sudo nano /etc/apache2/ports.conf
```

Modifiqué la directiva Listen para establecer el puerto 8080 , sustituyendo el puerto 80 por defecto.

```
  GNU nano 7.2                                 /etc/apache2/ports.conf *
# If you just change the port or add more ports here, you will likely also
# have to change the VirtualHost statement in
# /etc/apache2/sites-enabled/000-default.conf

Listen 8080

<IfModule ssl_module>
        Listen 443
</IfModule>

<IfModule mod_gnutls.c>
        Listen 443
</IfModule>
```

Luego edité el archivo de configuración del sitio predeterminado /etc/apache2/sites-available/000-default.conf. Allí actualicé la definición del <VirtualHost> para que también apunte al puerto 8080, asegurando así que el tráfico entrante por ese puerto sea manejado correctamente por este sitio.

```
ubuntu@ip-172-31-65-230:~$ sudo nano /etc/apache2/sites-available/000-default.conf
```

```
  GNU nano 7.2                  /etc/apache2/sites-available/000-default.conf *
<VirtualHost *:8080>
        # The ServerName directive sets the request scheme, hostname and port that
        # the server uses to identify itself. This is used when creating
        # redirection URLs. In the context of virtual hosts, the ServerName
        # specifies what hostname must appear in the request's Host: header to
        # match this virtual host. For the default virtual host (this file) this
        # value is not decisive as it is used as a last resort host regardless.
        # However, you must set it for any further virtual host explicitly.
        #ServerName www.example.com
```

Para dotar al servidor de capacidad de procesamiento dinámico, instalé PHP y su módulo de integración con Apache ejecutando el comando sudo apt install php libapache2-mod-php -y .

```
ubuntu@ip-172-31-65-230:~$ sudo apt install php libapache2-mod-php -y
```

Después, ejecuté sudo systemctl restart apache2 para reiniciar el servicio.

```
ubuntu@ip-172-31-65-230:~$ sudo systemctl restart apache2
```

Comprobé que el servicio se había levantado correctamente con sudo systemctl status apache2, confirmando que estaba active (running).

```
ubuntu@ip-172-31-65-230:~$ sudo systemctl status apache2
● apache2.service - The Apache HTTP Server
     Loaded: loaded (/usr/lib/systemd/system/apache2.service; enabled; preset: enabled)
     Active: active (running) since Fri 2025-11-28 08:47:15 UTC; 21s ago
       Docs: https://httpd.apache.org/docs/2.4/
    Process: 18226 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SUCCESS)
   Main PID: 18229 (apache2)
      Tasks: 6 (limit: 1008)
     Memory: 10.7M (peak: 11.0M)
        CPU: 53ms
     CGroup: /system.slice/apache2.service
             ├─18229 /usr/sbin/apache2 -k start
             ├─18231 /usr/sbin/apache2 -k start
             ├─18232 /usr/sbin/apache2 -k start
             ├─18233 /usr/sbin/apache2 -k start
             ├─18234 /usr/sbin/apache2 -k start
             └─18235 /usr/sbin/apache2 -k start

Nov 28 08:47:15 ip-172-31-65-230 systemd[1]: Starting apache2.service - The Apache HTTP Server...
Nov 28 08:47:15 ip-172-31-65-230 systemd[1]: Started apache2.service - The Apache HTTP Server.
```

Al intentar verificar los puertos de escucha con netstat, el sistema me indicó que el comando no existía.

```
ubuntu@ip-172-31-65-230:~$ sudo netstat -tulpn | grep apache2
sudo: netstat: command not found
```

Para solucionar esto y poder realizar las comprobaciones de red requeridas, instalé el paquete de herramientas necesario mediante sudo apt install net-tools.

```
ubuntu@ip-172-31-65-230:~$ sudo apt install net-tools
```

Con las herramientas instaladas, ejecuté nuevamente sudo netstat -tulpn | grep apache2. Gracias a esto, pude verificar visualmente que el proceso apache2 está escuchando en el puerto 8080 , validando así que la configuración interna coincide con las reglas de entrada que definí en el Security Group de AWS.

```
ubuntu@ip-172-31-65-230:~$ sudo netstat -tulpn | grep apache2
tcp6       0      0 :::8080                 :::*                    LISTEN      18229/apache2
```

Finalmente, creé un archivo de prueba llamado info.php en el directorio raíz /var/www/html/ utilizando un echo redirigido con tee. Este archivo contiene la función phpinfo() y me servirá para confirmar desde mi navegador local que el servidor procesa correctamente el código PHP.

```
ubuntu@ip-172-31-65-230:~$ echo "<?php phpinfo(); ?>" | sudo tee /var/www/html/info.php
<?php phpinfo(); ?>
```

Una vez configurado Apache, quise verificar que realmente estaba sirviendo contenido antes de pasar al siguiente servidor. Ejecuté el comando curl http://localhost:8080/info.php desde la propia terminal. La salida, que mostró el código HTML/CSS generado por la función phpinfo(), me confirmó que el servidor no solo escucha en el puerto 8080, sino que también procesa correctamente los scripts PHP.

```
ubuntu@ip-172-31-65-230:~$ curl http://localhost:8080/info.php
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"><head>
<style type="text/css">
body {background-color: #fff; color: #222; font-family: sans-serif;}
pre {margin: 0; font-family: monospace;}
a:link {color: #009; text-decoration: none; background-color: #fff;}
a:hover {text-decoration: underline;}
table {border-collapse: collapse; border: 0; width: 934px; box-shadow: 1px 2px 3px rgba(0, 0, 0, 0.2);
}
.center {text-align: center;}
.center table {margin: 1em auto; text-align: left;}
.center th {text-align: center !important;}
td, th {border: 1px solid #666; font-size: 75%; vertical-align: baseline; padding: 4px 5px;}
th {position: sticky; top: 0; background: inherit;}
h1 {font-size: 150%;}
h2 {font-size: 125%;}
h2 a:link, h2 a:visited{color: inherit; background: inherit;}
.p {text-align: left;}
.e {background-color: #ccf; width: 300px; font-weight: bold;}
.h {background-color: #99c; font-weight: bold;}
.v {background-color: #ddd; max-width: 300px; overflow-x: auto; word-wrap: break-word;}
.v i {color: #999;}
img {float: right; border: 0;}
```

Siguiendo los requisitos de la práctica para implementar múltiples servidores web, procedí a instalar Nginx ejecutando sudo apt install nginx -y.

```
ubuntu@ip-172-31-65-230:~$ sudo apt install nginx -y
```

Después, edité su archivo de configuración por defecto con sudo nano /etc/nginx/sites-available/default

```
ubuntu@ip-172-31-65-230:~$ sudo nano /etc/nginx/sites-available/default
```

Modifiqué la directiva listen para que el servidor escuche en el puerto 8081, tanto en IPv4 como en IPv6. Este cambio es fundamental para alinear el servicio con la regla del Security Group que creé en AWS y evitar conflictos con otros servicios.

```
  GNU nano 7.2                        /etc/nginx/sites-available/default
##
# You should look at the following URL's in order to grasp a solid understanding
# of Nginx configuration files in order to fully unleash the power of Nginx.
# https://www.nginx.com/resources/wiki/start/
# https://www.nginx.com/resources/wiki/start/topics/tutorials/config_pitfalls/
# https://wiki.debian.org/Nginx/DirectoryStructure
#
# In most cases, administrators will remove this file from sites-enabled/ and
# leave it as reference inside of sites-available where it will continue to be
# updated by the nginx packaging team.
#
# This file will automatically load configuration files provided by other
# applications, such as Drupal or Wordpress. These applications will be made
# available underneath a path with that package name, such as /drupal8.
#
# Please see /usr/share/doc/nginx-doc/examples/ for more detailed examples.
##

# Default server configuration
#
server {
        listen 8081 default_server;
        listen [::]:8081 default_server;
```

Generé una página de inicio personalizada mediante el comando echo redirigido con tee. En ella escribí un código HTML simple "Servidor Nginx Funcionando en puerto 8081" que me servirá para distinguir claramente cuándo estoy accediendo a este servidor específico desde mi navegador.

```
ubuntu@ip-172-31-65-230:~$ echo "<h1>Servidor Nginx</h1><p>Funcionando en puerto 8081</p>" | sudo tee
<h1>Servidor Nginx</h1><p>Funcionando en puerto 8081</p>
```

Tras guardar la configuración, reinicié el servicio con sudo systemctl restart caddy y verifiqué su estado, que se mostró activo.

```
ubuntu@ip-172-31-65-230:~$ sudo systemctl restart nginx
ubuntu@ip-172-31-65-230:~$ sudo systemctl status nginx
● nginx.service - A high performance web server and a reverse proxy server
     Loaded: loaded (/usr/lib/systemd/system/nginx.service; enabled; preset: enabled)
     Active: active (running) since Fri 2025-11-28 08:52:16 UTC; 3s ago
       Docs: man:nginx(8)
    Process: 18690 ExecStartPre=/usr/sbin/nginx -t -q -g daemon on; master_process on; (code=exited, >
    Process: 18691 ExecStart=/usr/sbin/nginx -g daemon on; master_process on; (code=exited, status=0/>
   Main PID: 18693 (nginx)
      Tasks: 3 (limit: 1008)
     Memory: 2.4M (peak: 2.6M)
        CPU: 19ms
     CGroup: /system.slice/nginx.service
             ├─18693 "nginx: master process /usr/sbin/nginx -g daemon on; master_process on;"
             ├─18694 "nginx: worker process"
             └─18695 "nginx: worker process"

Nov 28 08:52:16 ip-172-31-65-230 systemd[1]: Starting nginx.service - A high performance web server a>
Nov 28 08:52:16 ip-172-31-65-230 systemd[1]: Started nginx.service - A high performance web server an>
lines 1-17/17 (END)
```

Para confirmar técnicamente el cambio de puerto, utilicé sudo netstat -tulpn | grep caddy. El resultado en la terminal confirmó que Caddy está escuchando correctamente en el puerto 8082 (tanto en IPv4 como en IPv6), completando así la configuración del tercer servidor web.

```
ubuntu@ip-172-31-65-230:~$ sudo netstat -tulpn | grep nginx
tcp        0      0 0.0.0.0:8081            0.0.0.0:*               LISTEN      18693/nginx: master
tcp6       0      0 :::8081                 :::*                    LISTEN      18693/nginx: master
```

Hago la prueba con curl http://localhost:8081.

```
ubuntu@ip-172-31-65-230:~$ curl http://localhost:8080/info.php
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"><head>
<style type="text/css">
body {background-color: #fff; color: #222; font-family: sans-serif;}
pre {margin: 0; font-family: monospace;}
a:link {color: #009; text-decoration: none; background-color: #fff;}
a:hover {text-decoration: underline;}
table {border-collapse: collapse; border: 0; width: 934px; box-shadow: 1px 2px 3px rgba(0, 0, 0, 0.2);
}
.center {text-align: center;}
.center table {margin: 1em auto; text-align: left;}
.center th {text-align: center !important;}
td, th {border: 1px solid #666; font-size: 75%; vertical-align: baseline; padding: 4px 5px;}
th {position: sticky; top: 0; background: inherit;}
h1 {font-size: 150%;}
h2 {font-size: 125%;}
h2 a:link, h2 a:visited{color: inherit; background: inherit;}
.p {text-align: left;}
```

Lo primero que hice fue instalar los paquetes necesarios debian-keyring, apt-transport-https, curl para poder gestionar repositorios externos y claves de seguridad correctamente.

```
ubuntu@ip-172-31-65-230:~$ sudo apt install -y debian-keyring debian-archive-keyring apt-transport-http
s curl
```

A continuación, descargué la clave GPG oficial de Caddy y la añadí al llavero del sistema. Esto es un paso de seguridad vital para garantizar que los paquetes que descargue son auténticos y no han sido manipulados.

```
ubuntu@ip-172-31-65-230:~$ curl -1sLf 'https://dl.cloudsmith.io/public/caddy/stable/gpg.key' | sudo gpg
 --dearmor -o /usr/share/keyrings/caddy-stable-archive-keyring.gpg
```

Con la clave lista, agregué el repositorio oficial estable de Caddy a mi lista de fuentes /etc/apt/sources.list.d/caddy-stable.list. Esto le dice a mi sistema Ubuntu dónde buscar el software para instalarlo.

```
ubuntu@ip-172-31-65-230:~$ curl -1sLf 'https://dl.cloudsmith.io/public/caddy/stable/debian.deb.txt' | s
udo tee /etc/apt/sources.list.d/caddy-stable.list
# Source: Caddy
# Site: https://github.com/caddyserver/caddy
# Repository: Caddy / stable
# Description: Fast, multi-platform web server with automatic HTTPS


deb [signed-by=/usr/share/keyrings/caddy-stable-archive-keyring.gpg] https://dl.cloudsmith.io/public/ca
ddy/stable/deb/debian any-version main

deb-src [signed-by=/usr/share/keyrings/caddy-stable-archive-keyring.gpg] https://dl.cloudsmith.io/publi
c/caddy/stable/deb/debian any-version main
```

Una vez configurado el repositorio, actualicé la lista de paquetes sudo apt update y procedí a instalar el servidor web con el comando sudo apt install caddy -y.

```
ubuntu@ip-172-31-65-230:~$ sudo apt update && sudo apt install caddy -y
```

Creé un directorio específico para este servicio con sudo mkdir -p /var/www/caddy, donde alojar la página de prueba exclusiva para Caddy.

```
ubuntu@ip-172-31-65-230:~$ sudo mkdir -p /var/www/caddy
```

Cree un archivo Markdown llamado README.md. Este tipo de archivo se creó línea por línea utilizando el comando echo y canalizando la salida a sudo tee -a. El uso de tee -a fue esencial.
El título principal: # Bienvenido a Caddy.

Un mensaje de confirmación: Este servidor está funcionando correctamente..

Un encabezado secundario: ## Características.

Una lista de ítems: - Servidor moderno, - HTTPS automático, y - Fácil configuración.

```
ubuntu@ip-172-31-65-230:~$ echo "# Bienvenido a Caddy" | sudo tee /var/www/caddy/README.md
# Bienvenido a Caddy
ubuntu@ip-172-31-65-230:~$ echo "" | sudo tee -a /var/www/caddy/README.md

ubuntu@ip-172-31-65-230:~$ echo "Este servidor está funcionando correctamente." | sudo tee -a /var/www/
caddy/README.md
Este servidor está funcionando correctamente.
ubuntu@ip-172-31-65-230:~$ echo "" | sudo tee -a /var/www/caddy/README.md

ubuntu@ip-172-31-65-230:~$ echo "## Características" | sudo tee -a /var/www/caddy/README.md
## Características
ubuntu@ip-172-31-65-230:~$ echo "- Servidor moderno" | sudo tee -a /var/www/caddy/README.md
- Servidor moderno
ubuntu@ip-172-31-65-230:~$ echo "- HTTPS automático" | sudo tee -a /var/www/caddy/README.md
- HTTPS automático
ubuntu@ip-172-31-65-230:~$ echo "- Fácil configuración" | sudo tee -a /var/www/caddy/README.md
- Fácil configuración
```

Ejecuté `curl -o /tmp/test-image.jpg` para descargar una imagen PNG y la guardé temporalmente en la carpeta /tmp. La salida del comando confirma la transferencia completa del archivo.

```
ubuntu@ip-172-31-65-230:~$ curl -o /tmp/test-image.jpg "https://www.python.org/static/apple-touch-icon-
144x144-precomposed.png"
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  7382  100  7382    0     0   176k      0 --:--:-- --:--:-- --:--:--  180k
```

Utilicé `sudo mv /tmp/test-image.jpg /var/www/caddy/test.jpg` para mover la imagen de la carpeta temporal a la raíz web de Caddy, asegurando así que el servidor pueda acceder a ella y servirla bajo el nombre `test.jpg`.

```
ubuntu@ip-172-31-65-230:~$ sudo mv /tmp/test-image.jpg /var/www/caddy/test.jpg
```

Con todos los activos listos, el paso final de esta fase fue editar la configuración. Abrí el archivo principal de configuración de Caddy, el Caddyfile, utilizando `sudo nano /etc/caddy/Caddyfile` para definir el puerto de escucha y las directivas de servicio de archivos.

```
ubuntu@ip-172-31-65-230:~$ sudo nano /etc/caddy/Caddyfile
```

Modifiqué la configuración para que el servidor escuche en el puerto 8082.

```
  GNU nano 7.2                                  /etc/caddy/Caddyfile *
# The Caddyfile is an easy way to configure your Caddy web server.
#
# Unless the file starts with a global options block, the first
# uncommented line is always the address of your site.
#
# To use your own domain name (with automatic HTTPS), first make
# sure your domain's A/AAAA DNS records are properly pointed to
# this machine's public IP, then replace ":80" below with your
# domain name.

:8082 {
        # Set this path to your site's directory.
        root * /var/www/caddy

        # Enable the static file server.
        file_server browse

        @markdown path *.md

        header @markdown Content-Type text/plain

        # Another common task is to set up a reverse proxy:
        # reverse_proxy localhost:8080

        # Or serve a PHP site through php-fpm:
        # php_fastcgi localhost:9000
}
```

Ejecuté sudo systemctl restart caddy seguido de sudo systemctl status caddy. La salida confirmó que Caddy se cargó con la nueva configuración y se encontraba active (running).

```
ubuntu@ip-172-31-65-230:~$ sudo systemctl restart caddy
ubuntu@ip-172-31-65-230:~$ sudo systemctl status caddy
● caddy.service - Caddy
     Loaded: loaded (/usr/lib/systemd/system/caddy.service; enabled; preset: enabled)
     Active: active (running) since Fri 2025-11-28 08:59:35 UTC; 4s ago
       Docs: https://caddyserver.com/docs/
   Main PID: 19554 (caddy)
      Tasks: 7 (limit: 1008)
     Memory: 18.4M (peak: 18.9M)
        CPU: 91ms
     CGroup: /system.slice/caddy.service
             └─19554 /usr/bin/caddy run --environ --config /etc/caddy/Caddyfile

Nov 28 08:59:35 ip-172-31-65-230 caddy[19554]: {"level":"info","ts":1764320375.3634114,"logger":"admin>
Nov 28 08:59:35 ip-172-31-65-230 caddy[19554]: {"level":"info","ts":1764320375.3637493,"logger":"tls.c>
Nov 28 08:59:35 ip-172-31-65-230 caddy[19554]: {"level":"warn","ts":1764320375.364572,"logger":"http",>
Nov 28 08:59:35 ip-172-31-65-230 caddy[19554]: {"level":"warn","ts":1764320375.3645875,"logger":"http">
Nov 28 08:59:35 ip-172-31-65-230 caddy[19554]: {"level":"info","ts":1764320375.3645911,"logger":"http.>
Nov 28 08:59:35 ip-172-31-65-230 caddy[19554]: {"level":"info","ts":1764320375.3647187,"msg":"autosave>
Nov 28 08:59:35 ip-172-31-65-230 caddy[19554]: {"level":"info","ts":1764320375.3647616,"msg":"serving >
Nov 28 08:59:35 ip-172-31-65-230 systemd[1]: Started caddy.service - Caddy.
Nov 28 08:59:35 ip-172-31-65-230 caddy[19554]: {"level":"info","ts":1764320375.3690698,"logger":"tls",>
Nov 28 08:59:35 ip-172-31-65-230 caddy[19554]: {"level":"info","ts":1764320375.3691418,"logger":"tls",>
lines 1-21/21 (END)
```

Con sudo netstat -tulpn | grep caddy, confirmé visualmente que el proceso de Caddy estaba enlazado al puerto 8082, validando que el servidor estaba listo para responder al tráfico entrante en ese puerto.

```
ubuntu@ip-172-31-65-230:~$ sudo netstat -tulpn | grep caddy
tcp        0      0 127.0.0.1:2019          0.0.0.0:*               LISTEN      19554/caddy
tcp6       0      0 :::8082                 :::*                    LISTEN      19554/caddy
```

Ejecuté curl http://localhost:8082/. La respuesta fue el código HTML que genera Caddy para mostrar el índice del directorio <title>/</title>, confirmando que el servidor escucha en el puerto 8082 y que la directiva file_server browse funciona correctamente.

```
ubuntu@ip-172-31-65-230:~$ curl http://localhost:8082/

<!DOCTYPE html>
<html>
        <head>
                <title>/</title>
                <link rel="canonical" href="//"  />
                <meta charset="utf-8">
                <meta name="color-scheme" content="light dark">
                <meta name="viewport" content="width=device-width, initial-scale=1.0"
<style nonce="15dd08b7-1d1a-4143-9d48-a64967373c2c">
* { padding: 0; margin: 0; box-sizing: border-box; }

body {
        font-family: Inter, system-ui, sans-serif;
        font-size: 16px;
        text-rendering: optimizespeed;
        background-color: #f3f6f7;
        min-height: 100vh;
}

img,
svg {
        vertical-align: middle;
        z-index: 1;
}

img {
        max-width: 100%;
        max-height: 100%;
        border-radius: 5px;
```

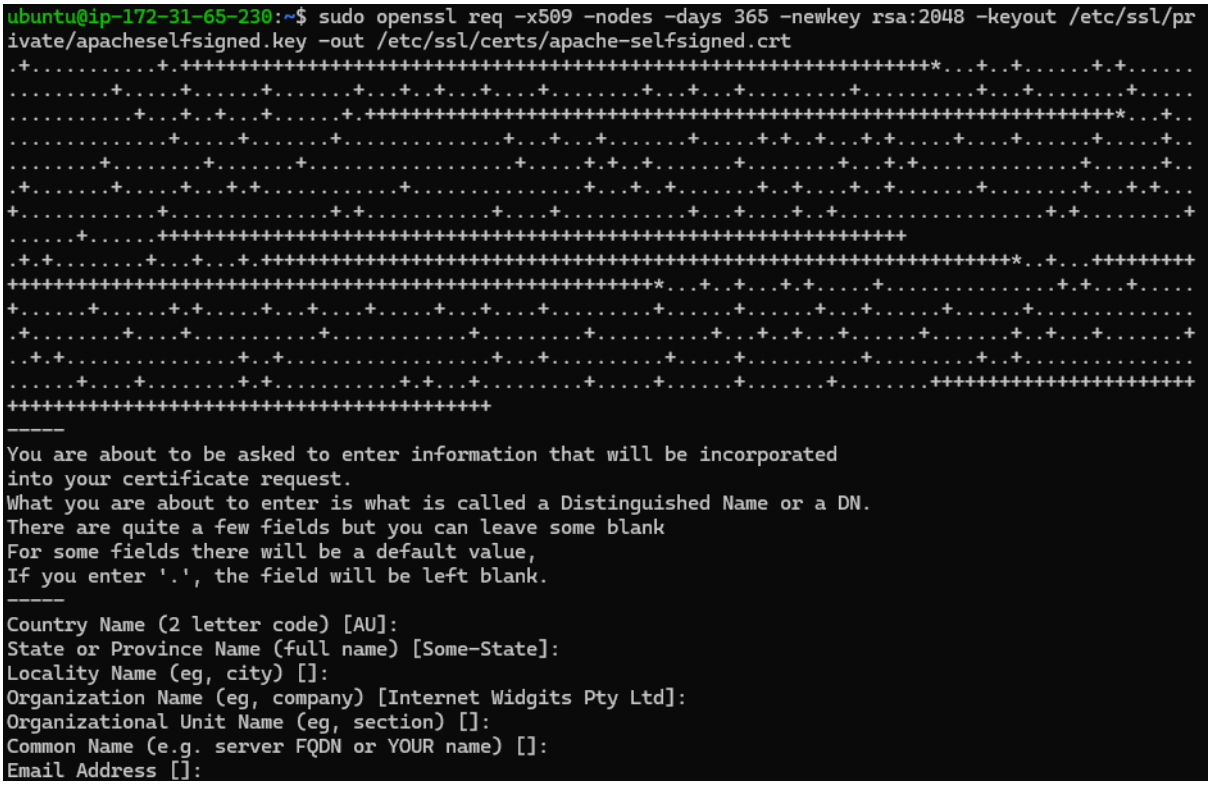
Al ejecutar curl http://localhost:8082/README.md, obtuve el texto plano del archivo Markdown que había creado. Esto verificó que Caddy estaba sirviendo el archivo y, crucialmente, que la configuración en el Caddyfile para forzar el Content-Type text/plain en los archivos .md estaba activa, mostrando el contenido sin renderizar.

```
ubuntu@ip-172-31-65-230:~$ curl http://localhost:8082/README.md
# Bienvenido a Caddy

Este servidor está funcionando correctamente.

## Características
- Servidor moderno
- HTTPS automático
- Fácil configuración
```

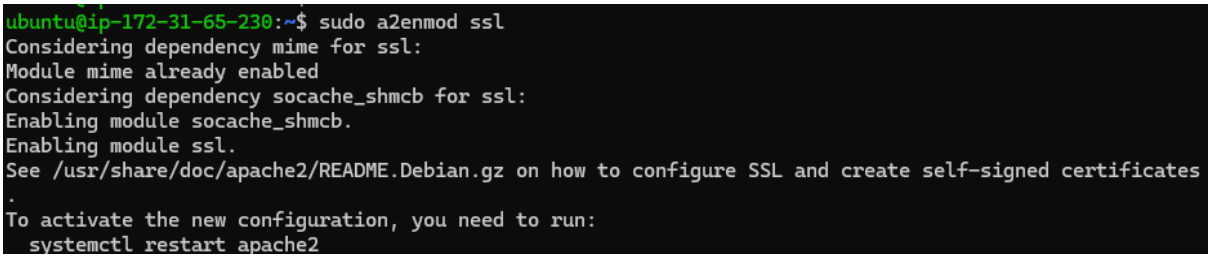
Ejecuté sudo apt install certbot python3-certbot-apache -y. Con este comando, instalé Certbot y sus dependencias para Apache.

```
ubuntu@ip-172-31-65-230:~$ sudo apt install certbot python3-certbot-apache -y
```

Ejecuté el comando sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048... Este comando genera la clave privada (apache-selfsigned.key) y el certificado público (apache-selfsigned.crt), solicitando varios datos para el Distinguished Name (DN) del certificado.



Una vez creados los archivos, habilité el módulo SSL de Apache con sudo a2enmod ssl. La salida confirmó que los módulos dependientes (mime y socache_shmcb) se activaron o ya estaban activos, y que el módulo ssl se habilitó correctamente.



Abrí el archivo de configuración de puertos con sudo nano /etc/apache2/ports.conf.

Dentro del archivo, modifiqué la directiva Listen para establecer el puerto 8443.

```
 GNU nano 7.2                                  /etc/apache2/ports.conf *
# If you just change the port or add more ports here, you will likely also
# have to change the VirtualHost statement in
# /etc/apache2/sites-enabled/000-default.conf

Listen 8443

<IfModule ssl_module>
        Listen 443
</IfModule>

<IfModule mod_gnutls.c>
        Listen 443
</IfModule>
```

Abrí el archivo de configuración con sudo nano /etc/apache2/sites-available/default-ssl.conf. Dentro de este archivo, definí el bloque <VirtualHost> para que escuche en el puerto 8443, el mismo que configuré previamente en ports.conf.

```
ubuntu@ip-172-31-65-230:~$ sudo nano /etc/apache2/sites-available/default-ssl.conf
```

```
 GNU nano 7.2                  /etc/apache2/sites-available/default-ssl.conf *
<VirtualHost *:8443>
        ServerAdmin webmaster@localhost
```

Ejecuté sudo a2ensite default-ssl.conf para crear el enlace simbólico del sitio en la carpeta sites-enabled, habilitando la nueva configuración segura.

```
ubuntu@ip-172-31-65-230:~$ sudo a2ensite default-ssl.conf
Enabling site default-ssl.
To activate the new configuration, you need to run:
  systemctl reload apache2
```

El sistema me indicó que debía ejecutar systemctl reload apache2. Lancé este comando y, por seguridad, también ejecuté sudo systemctl restart apache2 para asegurar que todos los cambios en la configuración SSL.

```
ubuntu@ip-172-31-65-230:~$ sudo systemctl reload apache2
```

```
ubuntu@ip-172-31-65-230:~$ sudo systemctl restart apache2
```

Ejecuté curl -i -k https://localhost:8443. El flag -k fue necesario para ignorar la advertencia de seguridad del certificado, ya que es autofirmado y no emitido por una autoridad reconocida.

```
ubuntu@ip-172-31-65-230:~$ curl -i -k https://localhost:8443
HTTP/1.1 200 OK
Date: Fri, 28 Nov 2025 09:08:36 GMT
Server: Apache/2.4.58 (Ubuntu)
Last-Modified: Fri, 28 Nov 2025 08:43:53 GMT
ETag: "29af-644a3a11f0242"
Accept-Ranges: bytes
Content-Length: 10671
Vary: Accept-Encoding
Content-Type: text/html

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <!--
    Modified from the Debian original for Ubuntu
    Last updated: 2022-03-22
    See: https://launchpad.net/bugs/1966004
  -->
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    <title>Apache2 Ubuntu Default Page: It works</title>
    <style type="text/css" media="screen">
  * {
    margin: 0px 0px 0px 0px;
    padding: 0px 0px 0px 0px;
  }

  body, html {
    padding: 3px 3px 3px 3px;
```

Ejecuté sudo systemctl status apache2. La salida confirmó que el servicio estaba active (running) y había iniciado correctamente el proceso principal (Main PID).

```
ubuntu@ip-172-31-65-230:~$ sudo systemctl status apache2
● apache2.service - The Apache HTTP Server
     Loaded: loaded (/usr/lib/systemd/system/apache2.service; enabled; preset: enabled)
     Active: active (running) since Fri 2025-11-28 09:08:26 UTC; 40s ago
       Docs: https://httpd.apache.org/docs/2.4/
    Process: 20204 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SUCCESS)
   Main PID: 20207 (apache2)
      Tasks: 6 (limit: 1008)
     Memory: 11.6M (peak: 12.1M)
        CPU: 67ms
     CGroup: /system.slice/apache2.service
             ├─20207 /usr/sbin/apache2 -k start
             ├─20209 /usr/sbin/apache2 -k start
             ├─20210 /usr/sbin/apache2 -k start
             ├─20211 /usr/sbin/apache2 -k start
             ├─20212 /usr/sbin/apache2 -k start
             └─20213 /usr/sbin/apache2 -k start

Nov 28 09:08:26 ip-172-31-65-230 systemd[1]: Starting apache2.service - The Apache HTTP Server...
Nov 28 09:08:26 ip-172-31-65-230 systemd[1]: Started apache2.service - The Apache HTTP Server.
```

Utilicé sudo netstat -tulpn | grep apache2. La salida confirmó visualmente que Apache estaba escuchando en los puertos 443 (HTTPS estándar, posiblemente para IPv6) y, más importante, en el puerto 8443 (HTTPS personalizado), lo que validaba la configuración completa del SSL.

```
ubuntu@ip-172-31-65-230:~$ sudo netstat -tulpn | grep apache2
tcp6       0      0 :::443                  :::*                    LISTEN      20207/apache2
tcp6       0      0 :::8443                 :::*                    LISTEN      20207/apache2
```

Ejecuté sudo systemctl status nginx y verifiqué que el servicio seguía active (running) y escuchando en el puerto 8081, confirmando que no fue afectado por los cambios SSL en Apache.

```
ubuntu@ip-172-31-65-230:~$ sudo systemctl status nginx
● nginx.service - A high performance web server and a reverse proxy server
     Loaded: loaded (/usr/lib/systemd/system/nginx.service; enabled; preset: enabled)
     Active: active (running) since Fri 2025-11-28 08:52:16 UTC; 17min ago
       Docs: man:nginx(8)
   Main PID: 18693 (nginx)
      Tasks: 3 (limit: 1008)
     Memory: 2.4M (peak: 2.6M)
        CPU: 19ms
     CGroup: /system.slice/nginx.service
             ├─18693 "nginx: master process /usr/sbin/nginx -g daemon on; master_process on;"
             ├─18694 "nginx: worker process"
             └─18695 "nginx: worker process"

Nov 28 08:52:16 ip-172-31-65-230 systemd[1]: Starting nginx.service - A high performance web server an>
Nov 28 08:52:16 ip-172-31-65-230 systemd[1]: Started nginx.service - A high performance web server and>
lines 1-15/15 (END)
```

Utilicé sudo netstat -tulpn | grep nginx y para verificar que cada uno estaba enlazado a su puerto respectivo

```
ubuntu@ip-172-31-65-230:~$ sudo netstat -tulpn | grep nginx
tcp        0      0 0.0.0.0:8081            0.0.0.0:*               LISTEN      18693/nginx: master
tcp6       0      0 :::8081                 :::*                    LISTEN      18693/nginx: master
```

Comprobé el estado con sudo systemctl status caddy y confirmé que también estaba active (running), asegurando que la última configuración en el Caddyfile se había cargado correctamente.

```
ubuntu@ip-172-31-65-230:~$ sudo systemctl status caddy
● caddy.service - Caddy
     Loaded: loaded (/usr/lib/systemd/system/caddy.service; enabled; preset: enabled)
     Active: active (running) since Fri 2025-11-28 08:59:35 UTC; 10min ago
       Docs: https://caddyserver.com/docs/
   Main PID: 19554 (caddy)
      Tasks: 7 (limit: 1008)
     Memory: 17.5M (peak: 18.9M)
        CPU: 111ms
     CGroup: /system.slice/caddy.service
             └─19554 /usr/bin/caddy run --environ --config /etc/caddy/Caddyfile

Nov 28 08:59:35 ip-172-31-65-230 caddy[19554]: {"level":"info","ts":1764320375.3634114,"logger":"admin>
Nov 28 08:59:35 ip-172-31-65-230 caddy[19554]: {"level":"info","ts":1764320375.3637493,"logger":"tls.c>
Nov 28 08:59:35 ip-172-31-65-230 caddy[19554]: {"level":"warn","ts":1764320375.364572,"logger":"http",>
Nov 28 08:59:35 ip-172-31-65-230 caddy[19554]: {"level":"warn","ts":1764320375.3645875,"logger":"http">
Nov 28 08:59:35 ip-172-31-65-230 caddy[19554]: {"level":"info","ts":1764320375.3645911,"logger":"http.>
Nov 28 08:59:35 ip-172-31-65-230 caddy[19554]: {"level":"info","ts":1764320375.3647187,"msg":"autosave>
Nov 28 08:59:35 ip-172-31-65-230 caddy[19554]: {"level":"info","ts":1764320375.3647616,"msg":"serving >
Nov 28 08:59:35 ip-172-31-65-230 systemd[1]: Started caddy.service - Caddy.
Nov 28 08:59:35 ip-172-31-65-230 caddy[19554]: {"level":"info","ts":1764320375.3690698,"logger":"tls",>
Nov 28 08:59:35 ip-172-31-65-230 caddy[19554]: {"level":"info","ts":1764320375.3691418,"logger":"tls",>
lines 1-21/21 (END)
```

Utilicé sudo netstat -tulpn | grep caddy para verificar que cada uno estaba enlazado a su puerto respectivo

```
ubuntu@ip-172-31-65-230:~$ sudo netstat -tulpn | grep caddy
tcp        0      0 127.0.0.1:2019          0.0.0.0:*               LISTEN      19554/caddy
tcp6       0      0 :::8082                 :::*                    LISTEN      19554/caddy
```

Lancé un sudo netstat -tulpn | grep -E '8080|8081|8082|8443' para tener una visión consolidada.

```
ubuntu@ip-172-31-65-230:~$ sudo netstat -tulpn | grep -E '8080|8081|8082|8443'
tcp        0      0 0.0.0.0:8081            0.0.0.0:*               LISTEN      18693/nginx: master
tcp6       0      0 :::8082                 :::*                    LISTEN      19554/caddy
tcp6       0      0 :::8081                 :::*                    LISTEN      18693/nginx: master
tcp6       0      0 :::8443                 :::*                    LISTEN      20207/apache2
```

Ejecuté curl http://localhost:8080. La respuesta obtenida fue el código HTML de la 'Apache2 Ubuntu Default Page: It works', confirmando la funcionalidad de Apache en su puerto HTTP personalizado.

```
ubuntu@ip-172-31-65-230:~$ curl http://localhost:8080
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <!--
    Modified from the Debian original for Ubuntu
    Last updated: 2022-03-22
    See: https://launchpad.net/bugs/1966004
  -->
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    <title>Apache2 Ubuntu Default Page: It works</title>
    <style type="text/css" media="screen">
  * {
    margin: 0px 0px 0px 0px;
    padding: 0px 0px 0px 0px;
  }

  body, html {
    padding: 3px 3px 3px 3px;

    background-color: #D8DBE2;

    font-family: Ubuntu, Verdana, sans-serif;
    font-size: 11pt;
    text-align: center;
  }

  div.main_page {
    position: relative;
    display: table;

    width: 800px;
```

Probé el servidor Nginx con curl http://localhost:8081. La respuesta, aunque en este caso también mostró la página por defecto de Apache, confirma que el servidor respondió a la petición en su puerto asignado.

```
ubuntu@ip-172-31-65-230:~$ curl http://localhost:8081
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xht
transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <!--
    Modified from the Debian original for Ubuntu
    Last updated: 2022-03-22
    See: https://launchpad.net/bugs/1966004
  -->
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    <title>Apache2 Ubuntu Default Page: It works</title>
    <style type="text/css" media="screen">
  * {
    margin: 0px 0px 0px 0px;
    padding: 0px 0px 0px 0px;
```

El comando curl http://localhost:8082 mostró la página de navegación de directorios generada por Caddy. Esto confirma que Caddy sirve contenido estático en el puerto 8082.

```
ubuntu@ip-172-31-65-230:~$ curl http://localhost:8082

<!DOCTYPE html>
<html>
        <head>
                <title>/</title>
                <link rel="canonical" href="//"  />
                <meta charset="utf-8">
                <meta name="color-scheme" content="light dark">
                <meta name="viewport" content="width=device-width, initial-scale=1.0">
<style nonce="b3b93fb8-e274-4caa-aab2-40e4b86de348">
* { padding: 0; margin: 0; box-sizing: border-box; }

body {
        font-family: Inter, system-ui, sans-serif;
        font-size: 16px;
        text-rendering: optimizespeed;
        background-color: #f3f6f7;
        min-height: 100vh;
}

img,
svg {
        vertical-align: middle;
        z-index: 1;
}

img {
```

Finalmente, ejecuté curl -i -k https://localhost:8443. La cabecera de respuesta mostró HTTP/1.1 200 OK y el campo Server: Apache/2.4.58 (Ubuntu)

```
ubuntu@ip-172-31-65-230:~$ curl -i -k https://localhost:8443
HTTP/1.1 200 OK
Date: Fri, 28 Nov 2025 09:12:12 GMT
Server: Apache/2.4.58 (Ubuntu)
Last-Modified: Fri, 28 Nov 2025 08:43:53 GMT
ETag: "29af-644a3a11f0242"
Accept-Ranges: bytes
Content-Length: 10671
Vary: Accept-Encoding
Content-Type: text/html

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <!--
    Modified from the Debian original for Ubuntu
    Last updated: 2022-03-22
    See: https://launchpad.net/bugs/1966004
  -->
  <head>
```