ARQUITECTURA EN LA NUBE - ADMINISTRACIÓN REMOTA DE SERVIDORES WEB EN AWS A TRAVÉS DE SSH

EVELYN ARCENTALES

24 DE NOVIEMBRE 2025

PARTE 0: PREPARACIÓN DEL ENTORNO LOCAL

Objetivo: Establecer un entorno de trabajo seguro en WSL2 para la gestión de conexiones SSH hacia instancias de AWS.

Comentarios: En esta fase inicial se verifica la correcta instalación de WSL2 y se prepara el sistema para trabajar con claves SSH. Se crea el directorio `.ssh` con permisos restrictivos (700) para garantizar la seguridad de las credenciales. Este paso es fundamental ya que establece las bases para todas las conexiones remotas posteriores.

COMANDOS Y PROCEDIMIENTOS PARTE 0:

PREPARACIÓN DEL ENTORNO LOCAL (WSL/VM) 1.

Verificar WSL2

```
nivecs@A6Alumno05:~$ wsl --version
Unknown command: --version
WSL
Wsman Shell commandLine, version 0.2.1

USAGE: wsl COMMAND [PARAMS...]

COMMANDS:
identify   - WS-Identify
enum       - WS-Enumerate
get        - WS-Get
put        - WS-Put
invoke     - WS-Invoke
xclean     - Delete all files generated by this tool set
xcred      - Create or display credential file
xcert      - Get server certificate (saved to <IPADDRESS>.crt)

PARAMS specification is specific to a COMMAND.

Output will be saved to ./response.xml. If you want to run parallel
executions in the same directory, define RTFILEPREFIX in the environment.
Doing so may significantly increase files generated.

Requires: curl, xmllint, GNU core utilities.
Optional: xsltproc for output formatting, gpg for encrypted credential.
Optional: wget as alternate for curl when not available.
```

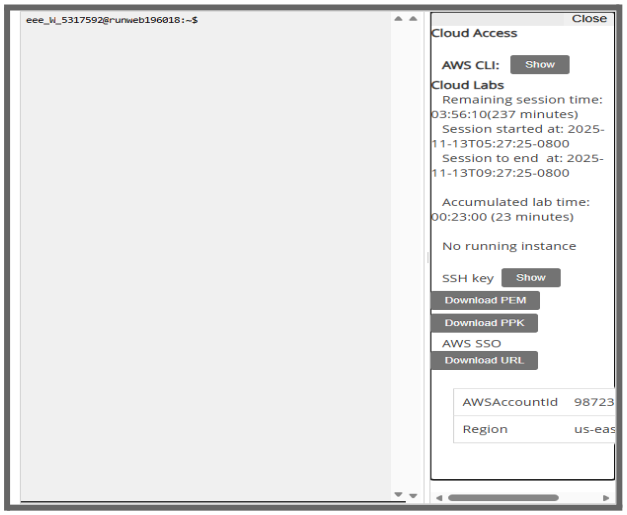
Descripción: Verifica que WSL2 esté instalado y funcionando correctamente.

2. Crear directorio para las claves SSH

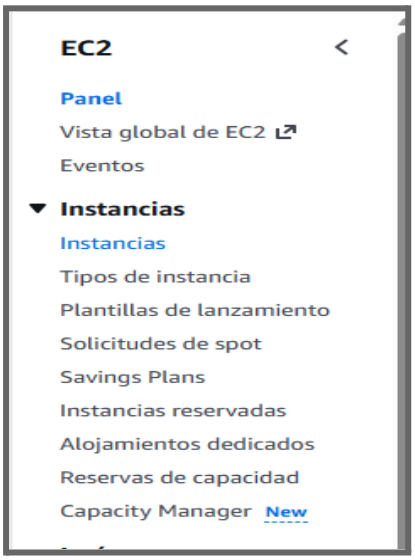
```
nivecs@A6Alumno05:~$ mkdir -p ~/.ssh
nivecs@A6Alumno05:~$ chmod 700 ~/.ssh
```

2. Crear directorio para las claves SSH
mkdir -p ~/.ssh
chmod 700 ~/.ssh

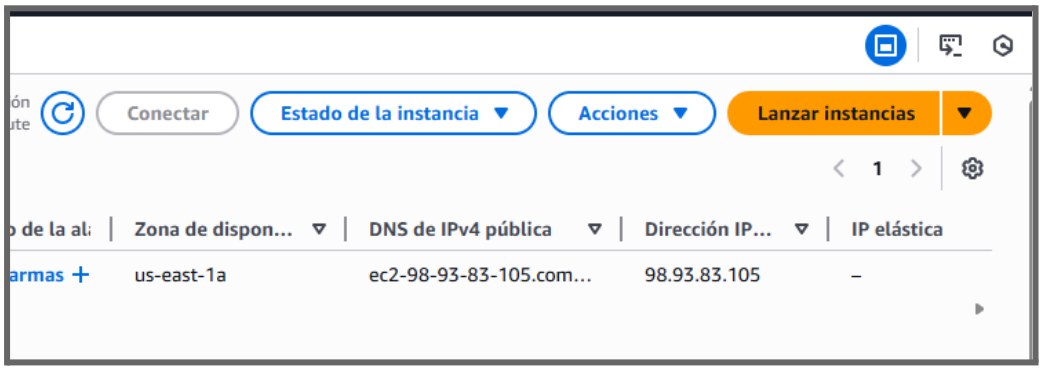
2. Configurar permisos de la clave PEM



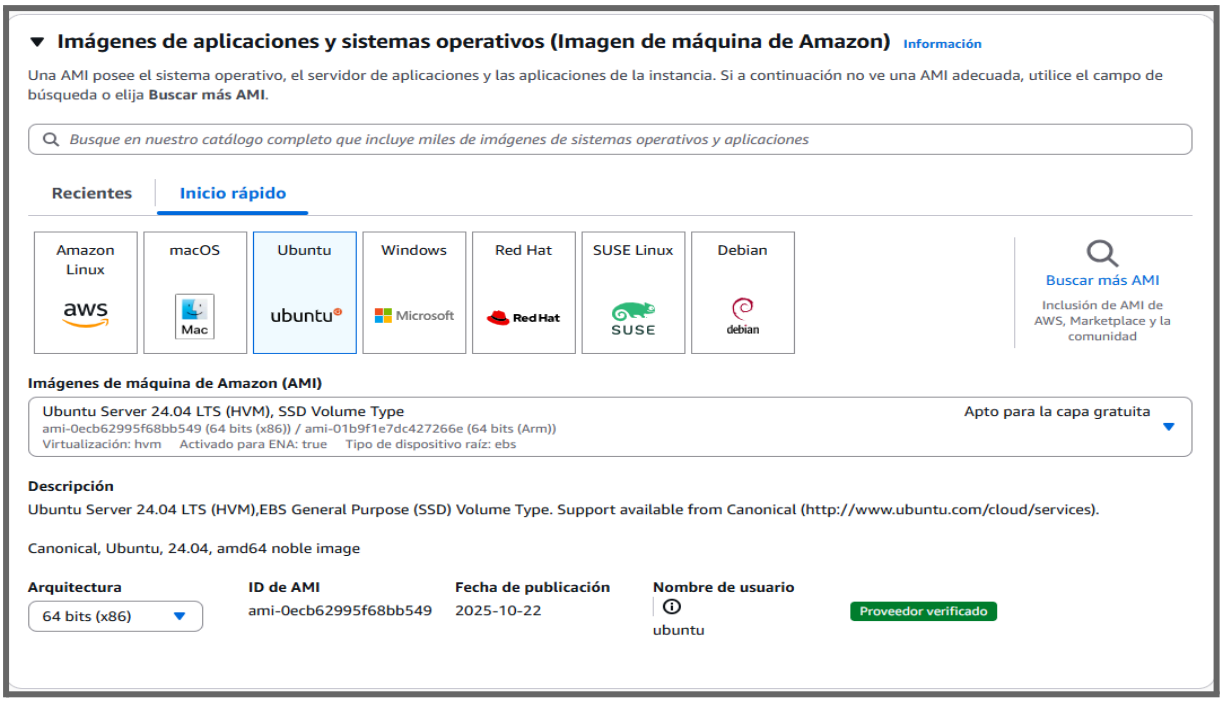
3. Crear instancia EC2



1.3 se da la opción de lanzar estancia

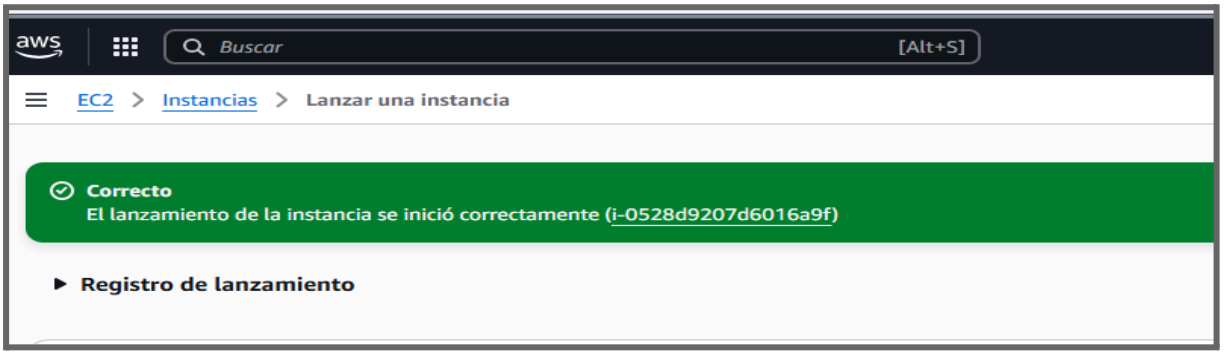


1.4 se recoge con que se va a trabajar



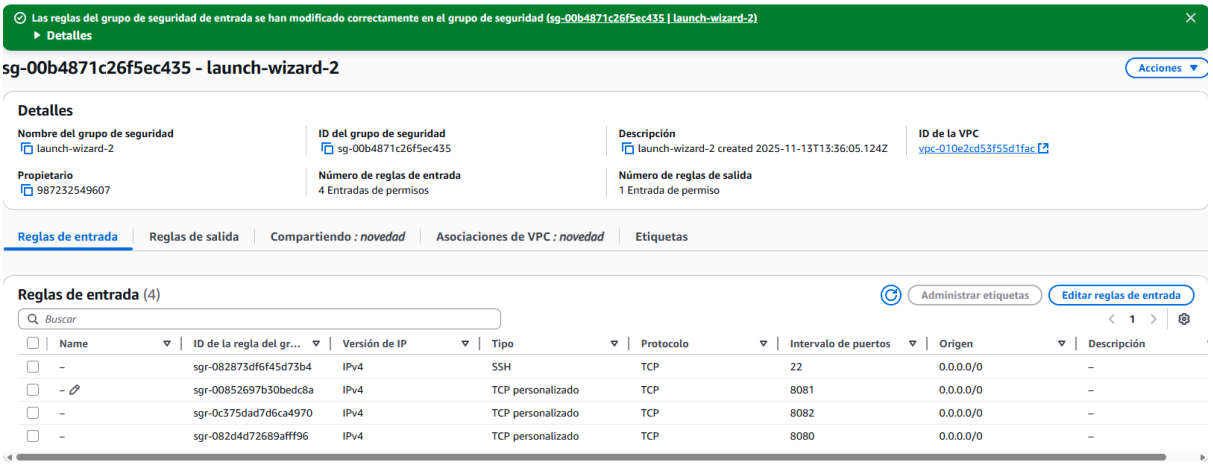
1.6 resumen de estancias

1.7



1.8 crear reglas de seguridad

1.10 Configurar Security Group (Reglas de entrada)



PARTE 1: CONFIGURACIÓN EN AWS (Interfaz Visual)

Objetivo: Iniciar el proceso de aprovisionamiento de una máquina virtual en la nube de AWS.

Comentarios: Esta interfaz permite seleccionar el tipo de instancia EC2 que se utilizará. Es el punto de partida para desplegar nuestra infraestructura en la nube, donde se definen los recursos computacionales necesarios.

Pasos para mover la clave a WSL:

Configuración de Clave PEM

Objetivo: Generar y descargar el par de claves criptográficas para autenticación SSH.

Comentarios: La clave PEM es el elemento de seguridad esencial para acceder remotamente a la instancia. Debe protegerse adecuadamente ya que es la única forma de autenticación en el servidor.

cp /mnt/c/Users/TU-USUARIO/Downloads/labsuser.pem ~/.ssh/

```
nivecs@A6Alumno05:~$ cp /mnt/c/Users/Alumno.DESKTOP-DI5KTUG/Desktop/iso/nube/labsuser.pem ~/.ssh/
```

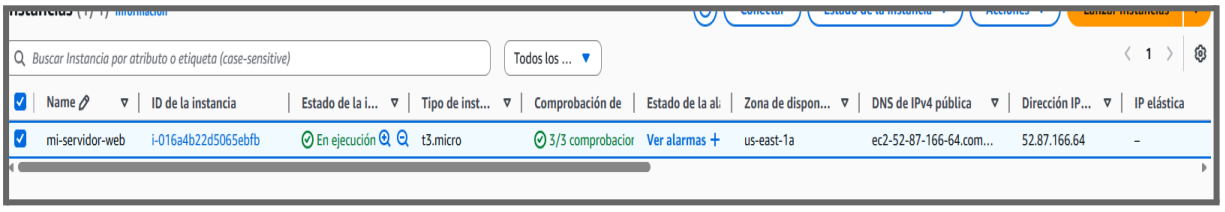
Creación y descarga del archivo .pem para autenticación SSH

2. Configurar permisos de la clave PEM

```
nivecs@A6Alumno05:~/.ssh$ chmod 400 labsuser.pem
```

```
nivecs@A6Alumno05:~$ ssh -i ~/.ssh/labsuser.pem ubuntu@52.87.166.64
```

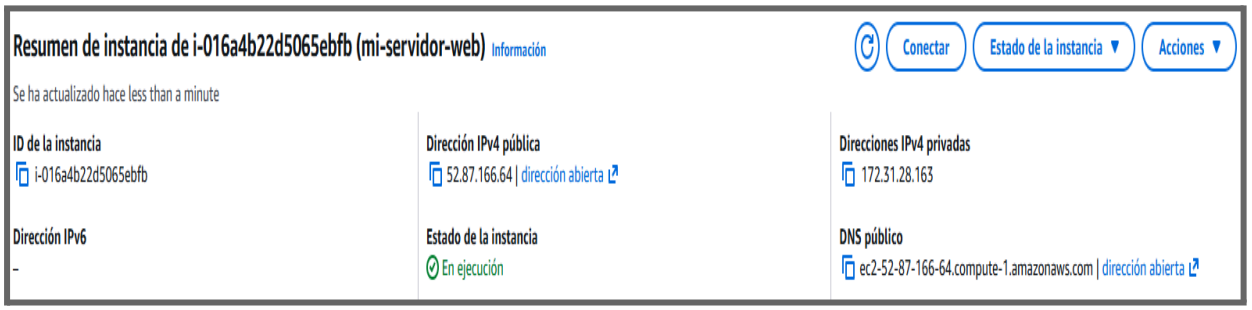
3. Crear instancia EC2



PARTE 2: CONEXIÓN SSH DESDE WSL A AWS

Objetivo: Establecer una conexión segura entre el entorno local y la instancia remota en AWS.

1. Obtener la IP pública de la instancia



2. Conectar por SSH

```
nivecs@A6Alumno05:~$ sudo ssh -i ~/.ssh/labsuser.pem ubuntu@52.87.166.64
The authenticity of host '52.87.166.64 (52.87.166.64)' can't be established.
ED25519 key fingerprint is SHA256:zoc3Q9TLKCtWIfTCvwv/ouRxHhDjwrjWKTBgt/mHgkU.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '52.87.166.64' (ED25519) to the list of known hosts.
Welcome to Ubuntu 24.04.3 LTS (GNU/Linux 6.14.0-1016-aws x86_64)

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

System information as of Tue Nov 18 07:59:02 UTC 2025

  System load:  0.0               Temperature:           -273.1 C
  Usage of /:   34.0% of 6.71GB   Processes:             111
  Memory usage: 22%               Users logged in:       0
  Swap usage:   0%                IPv4 address for ens5: 172.31.28.163
```

imagen 1.1

Se utiliza la IP pública de la instancia y la clave PEM previamente configurada para establecer una sesión SSH. Los permisos de la clave deben ser 400 o 600 para que SSH acepta su uso. Esta conexión permite administrar el servidor remoto como si estuviéramos trabajando localmente.

PARTE 3: EJECUTAR LA SEGUNDA PRÁCTICA ORIGINAL

1: INSTALACIÓN Y CONFIGURACIÓN DE APACHE:8080

Objetivo: Instalar y configurar el servidor web Apache para escuchar en un puerto no estándar.

```
ubuntu@ip-172-31-28-163:~$ 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 libapr1t64 libaprutil1-dbd-sqlite3 libaprutil1-ldap libaprutil1t64
  liblua5.4-0 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
  liblua5.4-0 ssl-cert
0 upgraded, 10 newly installed, 0 to remove and 5 not upgraded.
Need to get 2086 kB of archives.
```

imagen 1.0

1.1 Configurar Apache en puerto 8080 Comando:

Apache se configura en el puerto 8080 en lugar del puerto 80 predeterminado. Se modifica el archivo `ports.conf` y los VirtualHosts correspondientes. La instalación de PHP permite servir contenido dinámico, ampliando las capacidades del servidor.

```
  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>
```

imagen 1.1

Instalación de Apache - Comandos para instalar Apache2, PHP y configuración del puerto 8080 en archivos de configuración

1.2 Modificar el VirtualHost Comando:

```
 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

        ServerAdmin webmaster@localhost
        DocumentRoot /var/www/html

        # Available loglevels: trace8, ..., trace1, debug, info, notice, warn,
        # error, crit, alert, emerg.
        # It is also possible to configure the loglevel for particular
        # modules, e.g.
        #LogLevel info ssl:warn
```

imagen 1.2

1.3 comprobación

Verificación de Apache - Comprobación de puerto en escucha y prueba de funcionamiento mediante curl a info.php

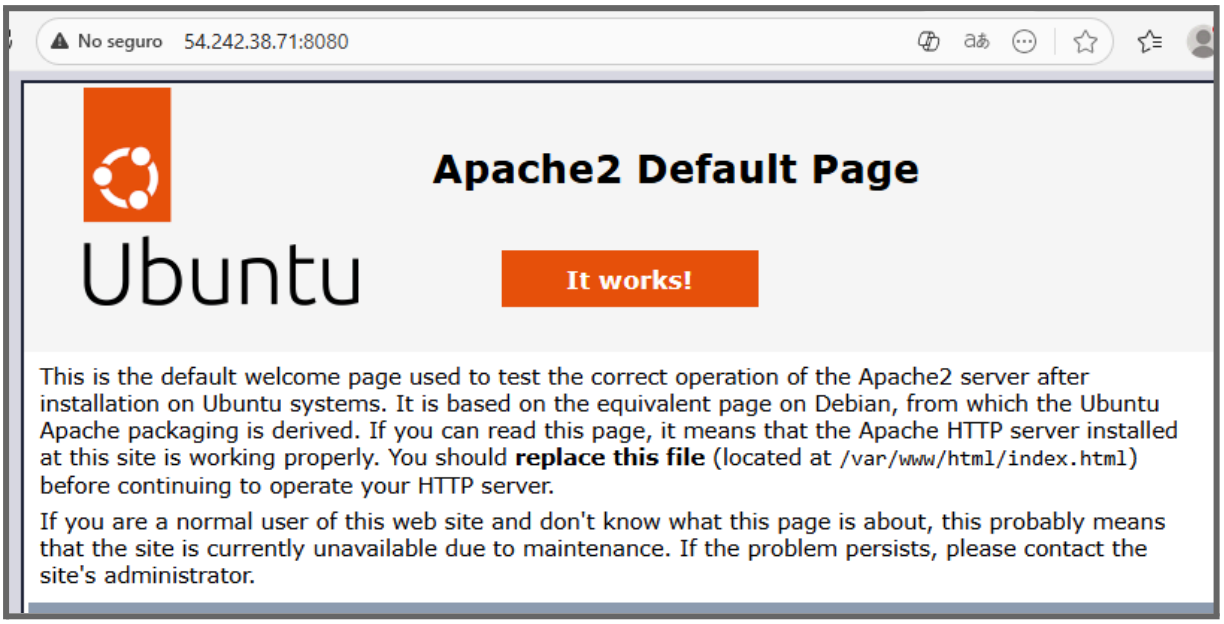


imagen 1.3

Se verifica mediante el comando `ss -tuln` que Apache está escuchando en el puerto 8080 y se prueba con `curl` para confirmar que el archivo PHP de prueba responde adecuadamente.

5. Instalar PHP Comando:

```
ubuntu@ip-172-31-28-163:~$ sudo  apt install php libapache2-mod-php -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libapache2-mod-php8.3 php-common php8.3 php8.3-cli php8.3-common php8.3-opcache php8.3-readline
Suggested packages:
  php-pear
The following NEW packages will be installed:
  libapache2-mod-php libapache2-mod-php8.3 php php-common php8.3 php8.3-cli php8.3-common php8.3-opcache
  php8.3-readline
0 upgraded, 9 newly installed, 0 to remove and 5 not upgraded.
Need to get 4922 kB of archives.
```

imagen 1.4

confirma el éxito en el despliegue y configuración de los tres servidores web, cada uno en un puerto diferente, y verifica la configuración HTTPS experimental en Apache, aunque con la limitación natural de un certificado autofirmado.

6. Reiniciar Apache Comando:

```
ubuntu@ip-172-31-28-163:~$ sudo systemctl restart apache2
ubuntu@ip-172-31-28-163:~$
```

imagen 1.5

El comando se ejecuta sin mostrar una salida (silenciosamente), indicando un reinicio exitoso del servicio.

7. Verificar estado de Apache Comando:

```
ubuntu@ip-172-31-28-163:~$ 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 Tue 2025-11-18 13:30:13 UTC; 1min 35s ago
       Docs: https://httpd.apache.org/docs/2.4/
    Process: 7782 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SUCCESS)
   Main PID: 7785 (apache2)
      Tasks: 6 (limit: 1017)
     Memory: 10.8M (peak: 11.1M)
        CPU: 55ms
     CGroup: /system.slice/apache2.service
             ├─7785 /usr/sbin/apache2 -k start
             ├─7787 /usr/sbin/apache2 -k start
             ├─7788 /usr/sbin/apache2 -k start
             ├─7789 /usr/sbin/apache2 -k start
             ├─7790 /usr/sbin/apache2 -k start
             └─7791 /usr/sbin/apache2 -k start

Nov 18 13:30:13 ip-172-31-28-163 systemd[1]: Starting apache2.service - The Apache HTTP Server...
Nov 18 13:30:13 ip-172-31-28-163 systemd[1]: Started apache2.service - The Apache HTTP Server.
```

imagen 1.6

Esta imagen muestra el resultado del comando utilizado para **verificar el estado del servicio Apache2** en la instancia EC2 remota.

comando: ss -tuln | grep 8080

```
ubuntu@ip-172-31-28-163:~$ ss -tuln | grep 8080
tcp   LISTEN 0      511                *:8080            *:*
ubuntu@ip-172-31-28-163:~$
```

imagen 1.7

El resultado verifica que la modificación del archivo `ports.conf` de Apache, cambiando el puerto de escucha predeterminado (80) al puerto **8080** , fue exitosa

8. Crear archivo PHP de prueba Comando:

```
ubuntu@ip-172-31-28-163:~$ echo "<?php  phpinfo(); ?>" | sudo tee /var/www/html/info.php
<?php  phpinfo(); ?>
```

imagen 1.8

Esta imagen muestra la ejecución del comando para **crear un archivo de prueba en PHP** dentro del directorio raíz de Apache.

9. Probar Apache desde terminal Comando:

curl http://localhost:8080/info.php

```
ubuntu@ip-172-31-28-163:~$ curl http://54.242.38.71:8080/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;}
hr {width: 934px; background-color: #ccc; border: 0; height: 1px;}
:root {--php-dark-grey: #333; --php-dark-blue: #4F5B93; --php-medium-blue: #8892BF; --php-light-blue: #E2E4EF; --php-
accent-purple: #793862}@media (prefers-color-scheme: dark) {
```

imagen 1.9

Esta imagen muestra la ejecución del comando para **probar el archivo `info.php`** en el servidor Apache desde la terminal.

Uso de IP Pública: Aunque el paso se titula "Probar Apache desde terminal", el comando utiliza la IP pública de la instancia (`54.242.38.71`) en lugar de `localhost`. Esto sugiere una prueba de conectividad de red interna/externa utilizando la interfaz pública.

PARTE 2: INSTALACIÓN Y CONFIGURACIÓN DE NGINX

1. Instalar Nginx Comando:

sudo apt install nginx -y

```
ubuntu@ip-172-31-28-163:~$ sudo apt install nginx -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  nginx-common
Suggested packages:
  fcgiwrap nginx-doc
The following NEW packages will be installed:
  nginx nginx-common
0 upgraded, 2 newly installed, 0 to remove and 5 not upgraded.
Need to get 564 kB of archives.
After this operation, 1596 kB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 nginx-common all 1.24.0-2ubuntu7.5 [43.
4 kB]
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 nginx amd64 1.24.0-2ubuntu7.5 [520 kB]
Fetched 564 kB in 0s (17.2 MB/s)
```

imagen 2.1

Preparación para el Puerto 8081: Una vez instalado, el siguiente paso será modificar el archivo de configuración de Nginx para que escuche en el puerto 8081, evitando conflictos con Apache (puerto 8080).

2. Configurar Nginx en puerto 8081

```
ubuntu@ip-172-31-28-163: ~
  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;

        # SSL configuration
        #
        # listen 443 ssl default_server;
        # listen [::]:443 ssl default_server;
        #
```

imagen 2.2

Configurar Nginx para que escuche en el puerto **8081**, permitiendo que coexista con Apache (que usa el puerto 8080) en la misma instancia EC2, y evitando el puerto 80 estándar.

3. Crear página HTML personalizada

```
ubuntu@ip-172-31-28-163:~$ echo "<h1>Servidor Nginx de Evelyn</h1><p>Funcionando en puerto 8081</p>" | sudo tee /usr/
share/nginx/html/index.html
<h1>Servidor Nginx de Evelyn</h1><p>Funcionando en puerto 8081</p>
ubuntu@ip-172-31-28-163:~$
```

. imagen 2.3

Reemplazar la página de bienvenida predeterminada de Nginx con contenido simple y personalizado que indique claramente que el servidor está activo y funcionando en el puerto **8081**.

4. Reiniciar Nginx

```
ubuntu@ip-172-31-28-163:~$ sudo systemctl restart nginx
```

imagen 2.4

Este paso prepara el servidor para la verificación de estado y la prueba funcional en el puerto **8081**, confirmando la coexistencia con Apache.

5. Verificar estado de Nginx

```
ubuntu@ip-172-31-28-163:~$ 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 Tue 2025-11-18 13:54:03 UTC; 17s ago
       Docs: man:nginx(8)
    Process: 8272 ExecStartPre=/usr/sbin/nginx -t -q -g daemon on; master_process on; (code=exited, status=0/SUCCESS)
    Process: 8274 ExecStart=/usr/sbin/nginx -g daemon on; master_process on; (code=exited, status=0/SUCCESS)
   Main PID: 8276 (nginx)
      Tasks: 3 (limit: 1017)
     Memory: 2.4M (peak: 2.6M)
        CPU: 15ms
     CGroup: /system.slice/nginx.service
             ├─8276 "nginx: master process /usr/sbin/nginx -g daemon on; master_process on;"
             ├─8277 "nginx: worker process"
             └─8278 "nginx: worker process"

Nov 18 13:54:03 ip-172-31-28-163 systemd[1]: Starting nginx.service - A high performance web server and a reverse pr>
Nov 18 13:54:03 ip-172-31-28-163 systemd[1]: Started nginx.service - A high performance web server and a reverse pro>
ubuntu@ip-172-31-28-163:~$
```

imagen 2.5

comprobación en que puerto esta escuchando

Confirmar que el servidor web **Nginx** se inició correctamente después de la instalación y la configuración para el puerto 8081.

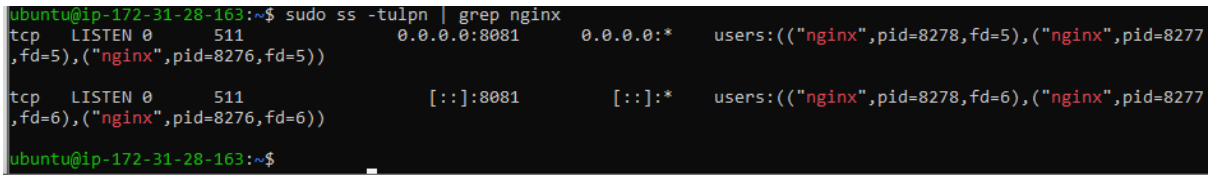


imagen 2.6

Confirmar que el servidor **Nginx** está escuchando activamente en el puerto **8081**, tal como se configuró previamente, y no en el puerto 80 predeterminado.

6. Probar Nginx desde terminal

curl http://localhost:8081

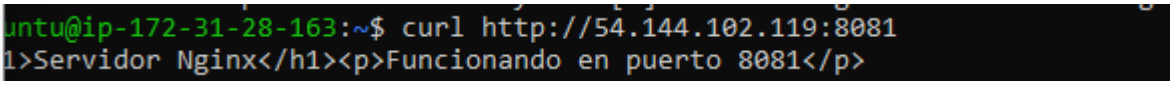


imagen 2.7

7. comprobación en web

imagen 2.8

Esta prueba confirma que las Reglas de Entrada (Security Group) en AWS están configuradas correctamente para permitir el tráfico al puerto 8081, y que Nginx está enlazado a la dirección IP pública de la instancia.

PARTE 3: INSTALACIÓN Y CONFIGURACIÓN DE CADDY

1. Instalar dependencias necesarias

Comando:
sudo apt install -y debian-keyring debian-archive-keyring apt-transport-https curl

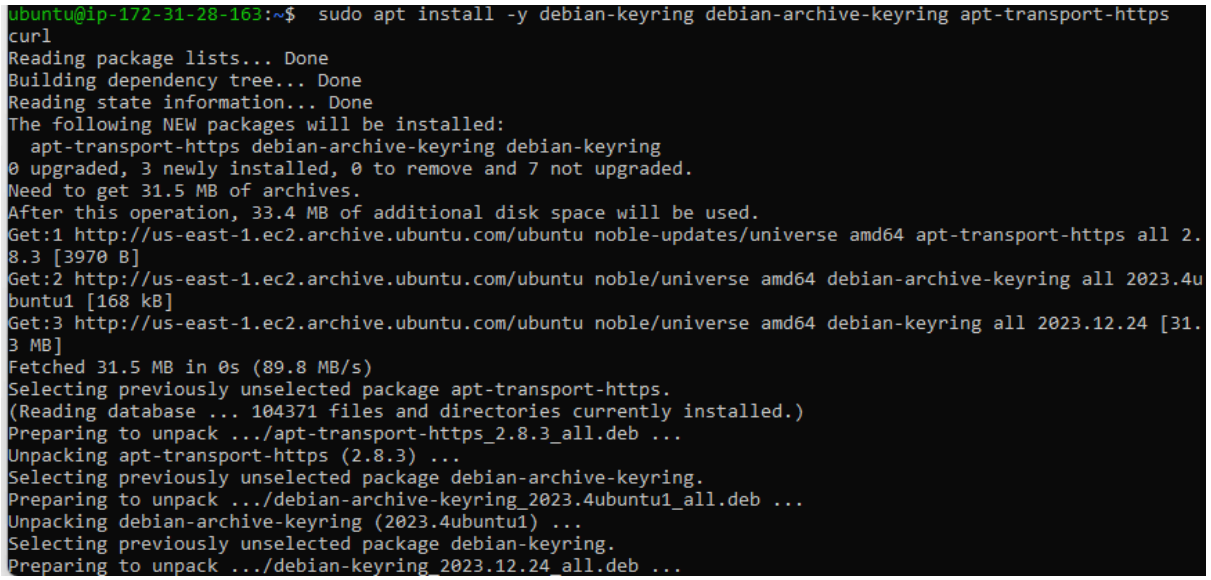


imagen 3.1

La instalación de los *keyrings* y `apt-transport-https` garantiza que el sistema operativo pueda **confiar y verificar** la autenticidad del repositorio de Caddy, asegurando que los paquetes descargados no han sido manipulados.

2. Agregar repositorio de Caddy

curl -1sLf 'https://dl.cloudsmith.io/public/caddy/stable/gpg.key' | sudo gpg - dearmor -o /usr/share/keyrings/caddy-stable-archive-keyring.gpg

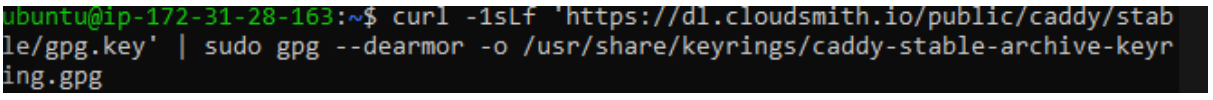


imagen 3.2

La importación de esta clave GPG asegura que los paquetes de Caddy que se descarguen en el siguiente paso provienen del proveedor legítimo y no han sido alterados, lo cual es fundamental para mantener la seguridad del servidor.

curl -1sLf 'https://dl.cloudsmith.io/public/caddy/stable/debian.deb.txt' | sudo tee /etc/apt/sources.list.d/caddy-stable.list

```
ubuntu@ip-172-31-28-163:~$  curl -1sLf 'https://dl.cloudsmith.io/public/caddy/sta
ble/debian.deb.txt' | sudo 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.c
loudsmith.io/public/caddy/stable/deb/debian any-version main

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

imagen 3.3

Descargar la lista de fuentes del repositorio estable de Caddy y guardarla en un nuevo archivo dentro del directorio `/etc/apt/sources.list.d/`. Esto le indica al gestor de paquetes `apt` dónde encontrar los paquetes de instalación de Caddy.

3. Actualizar e instalar Caddy

sudo apt update && sudo apt install caddy -y

```
ubuntu@ip-172-31-28-163:~$
sudo apt update && sudo apt install caddy -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 [1
26 kB]
Get:4 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 Pac
kages [1619 kB]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main Translati
on-en [303 kB]
Get:7 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 Com
ponents [175 kB]
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 c-n
-f Metadata [15.7 kB]
Get:9 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64
 Packages [1500 kB]
Get:10 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd6
4 Components [377 kB]
Get:11 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd6
4 c-n-f Metadata [31.4 kB]
Get:12 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted am
d64 Packages [2307 kB]
Get:13 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted Tr
anslation-en [526 kB]
Get:14 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted am
d64 Components [212 B]
Get:15 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse am
d64 Components [940 B]
```

imagen 3.4

La ejecución de `apt update` es la confirmación de que la lista de fuentes de Caddy se cargó correctamente y el sistema está listo para descargar e instalar el software.

4. Crear directorio para Caddy

sudo mkdir -p /var/www/caddy

```
ubuntu@ip-172-31-28-163:~$  sudo mkdir -p /var/www/caddy
ubuntu@ip-172-31-28-163:~$
```

imagen 3.5

5. Crear archivo Markdown de prueba

Objetivo: Crear contenido de prueba en formato Markdown para el servidor Caddy, ubicado en el directorio raíz de su sitio web (`/var/www/caddy`), y asegurar que Caddy pueda servir correctamente los archivos de texto y el contenido Markdown.

echo "# Bienvenido a Caddy" | sudo tee /var/www/caddy/README.md

```
ubuntu@ip-172-31-28-163:~$ echo "# Bienvenido a Caddy" | sudo tee /var/www/caddy/
README.md
# Bienvenido a Caddy
ubuntu@ip-172-31-28-163:~$
```

echo "" | sudo tee -a /var/www/caddy/README.md

```
ubuntu@ip-172-31-28-163:~$  echo "" | sudo tee -a /var/www/caddy/README.md

ubuntu@ip-172-31-28-163:~$
```

echo "Este servidor está funcionando correctamente." | sudo tee -a /var/www/caddy/README.md

```
ubuntu@ip-172-31-28-163:~$  echo "Este servidor está funcionando correctamente."
| sudo tee -a /var/www/caddy/README.md
Este servidor está funcionando correctamente.
ubuntu@ip-172-31-28-163:~$
```

echo "" | sudo tee -a /var/www/caddy/README.md

```
ubuntu@ip-172-31-28-163:~$ echo "" | sudo tee -a /var/www/caddy/README.md

ubuntu@ip-172-31-28-163:~$
```

echo "## Características" | sudo tee -a /var/www/caddy/README.md

```
ubuntu@ip-172-31-28-163:~$  echo "## Características" | sudo tee -a /var/www/cadd
y/README.md
## Características
ubuntu@ip-172-31-28-163:~$
```

echo "- Servidor moderno" | sudo tee -a /var/www/caddy/README.md

```
ubuntu@ip-172-31-28-163:~$ echo "- Servidor moderno" | sudo tee -a /var/www/caddy
/README.md~
- Servidor moderno
ubuntu@ip-172-31-28-163:~$
```

echo "- HTTPS automático" | sudo tee -a /var/www/caddy/README.md

```
ubuntu@ip-172-31-28-163:~$  echo "- HTTPS automático" | sudo tee -a /var/www/cadd
y/README.md
- HTTPS automático
ubuntu@ip-172-31-28-163:~$
```

echo "- Fácil configuración" | sudo tee -a /var/www/caddy/README.md

```
ubuntu@ip-172-31-28-163:~$ echo "- Fácil configuración" | sudo tee -a /var/www/ca
ddy/README.md
- Fácil configuración
ubuntu@ip-172-31-28-163:~$
```

6. Crear imagen de prueba

Verificación de Servidor de Archivos: Al agregar una imagen (`test.jpg`) además del archivo de texto (`README.md`), se prueba la capacidad del servidor Caddy para funcionar como un servidor de archivos estáticos básico.

curl -o /tmp/test-image.jpg "https://www.python.org/static/apple-touch icon-144x144-precomposed.png"

```
ubuntu@ip-172-31-28-163:~$ curl -o /tmp/test-image.jpg "https://www.python.org/st
atic/apple-touchicon-144x144-precomposed.png"
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   146  100   146    0     0   1525      0 --:--:-- --:--:-- --:--:--  1536
ubuntu@ip-172-31-28-163:~$
```

sudo mv /tmp/test-image.jpg /var/www/caddy/test.jpg

```
ubuntu@ip-172-31-28-163:~$  sudo mv /tmp/test-image.jpg /var/www/caddy/test.jpg
ubuntu@ip-172-31-28-163:~$
```

7. Crear Caddyfile personalizado

sudo nano /etc/caddy/Caddyfile

```
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


Refer to the Caddy docs for more information:
https://caddyserver.com/docs/caddyfile
```

imagen 3.6

La configuración de la cabecera para archivos `.md` es una directiva importante para controlar cómo el navegador interpreta el archivo Markdown, asegurando que la prueba de contenido sea precisa.

8. Reiniciar Caddy

sudo systemctl restart caddy

```
ubuntu@ip-172-31-28-163:~$ sudo systemctl restart caddy
ubuntu@ip-172-31-28-163:~$
```

9. Verificar estado de Caddy

sudo systemctl status caddy

```
ubuntu@ip-172-31-28-163:~$  sudo systemctl status caddy
● caddy.service - Caddy
     Loaded: loaded (/usr/lib/systemd/system/caddy.service; enabled; preset: ena>
     Active: active (running) since Sun 2025-11-23 22:15:55 UTC; 1min 6s ago
       Docs: https://caddyserver.com/docs/
   Main PID: 2763 (caddy)
      Tasks: 7 (limit: 1017)
     Memory: 6.7M (peak: 7.1M)
        CPU: 29ms
     CGroup: /system.slice/caddy.service
             └─2763 /usr/bin/caddy run --environ --config /etc/caddy/Caddyfile

Nov 23 22:15:55 ip-172-31-28-163 caddy[2763]: {"level":"info","ts":1763936155.50>
Nov 23 22:15:55 ip-172-31-28-163 caddy[2763]: {"level":"warn","ts":1763936155.50>
Nov 23 22:15:55 ip-172-31-28-163 caddy[2763]: {"level":"info","ts":1763936155.50>
Nov 23 22:15:55 ip-172-31-28-163 caddy[2763]: {"level":"info","ts":1763936155.50>
Nov 23 22:15:55 ip-172-31-28-163 caddy[2763]: {"level":"info","ts":1763936155.50>
Nov 23 22:15:55 ip-172-31-28-163 caddy[2763]: {"level":"info","ts":1763936155.50>
Nov 23 22:15:55 ip-172-31-28-163 caddy[2763]: {"level":"info","ts":1763936155.50>
Nov 23 22:15:55 ip-172-31-28-163 caddy[2763]: {"level":"info","ts":1763936155.50>
Nov 23 22:15:55 ip-172-31-28-163 systemd[1]: Started caddy.service - Caddy.
lines 1-21/21 (END)
```

imagen 3.7

Confirmar que el tercer servidor web, **Caddy**, se inició correctamente después de su instalación y configuración con el Caddyfile personalizado (puerto 8082).

10. Probar Caddy desde terminal

curl http://localhost:8082/

```
ubuntu@ip-172-31-28-163:~$ curl http://localhost:8082/
<!DOCTYPE html>
<html>

<head>
    <title>/</title>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <style>
        * {
            padding: 0;
            margin: 0;
        }

        body {
            font-family: sans-serif;
            text-rendering: optimizespeed;
            background-color: #ffffff;
        }

        a {
            color: #006ed3;
            text-decoration: none;
        }

        a:hover,
        h1 a:hover {
            color: #319cff;
        }

        a:visited {
            color: #800080;
        }

        header,
        #summary {
            padding-left: 5%;
            padding-right: 5%;
        }
```

imagen 3.8

El comando devuelve código HTML/CSS (parcialmente visible). Este código corresponde a la página que Caddy genera automáticamente cuando se habilita la función `file_server browse` en el Caddyfile. Esta página es un listado navegable de los archivos dentro del directorio raíz (`/var/www/caddy`).

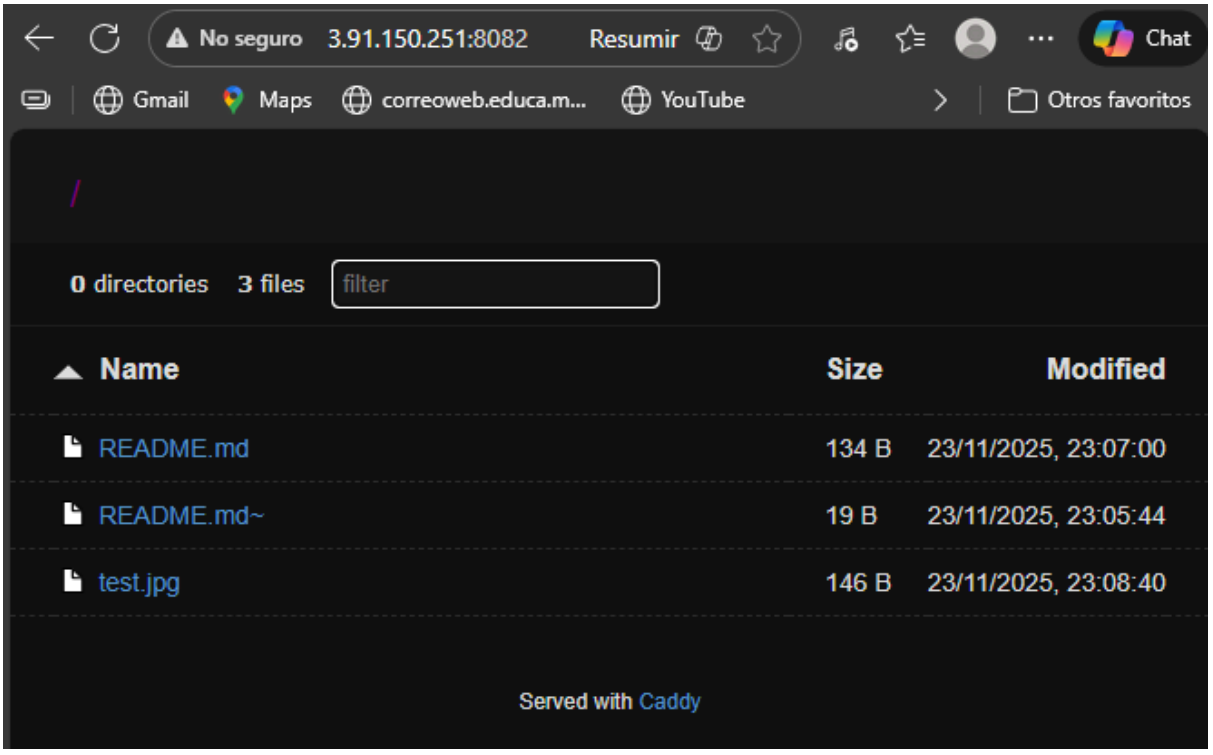


imagen 3.9

El navegador muestra la lista de archivos del directorio raíz del servidor Caddy:

- **`README.md`** (el archivo Markdown de prueba).
- **`README.md~`** (un archivo temporal creado por el editor, con 19 B).
- **`test.jpg`** (la imagen descargada de prueba).

11. Probar archivo Markdown

curl http://localhost:8082/README.md

```
ubuntu@ip-172-31-28-163:~$ curl http://localhost:8082/README.md
# Bienvenido a Caddy

Este servidor está funcionando correctamente.

## Características
- HTTPS automático
- Fácil configuración
ubuntu@ip-172-31-28-163:~$
```

imagen 3.10

El comando devuelve el contenido plano del archivo `README.md`, incluyendo el formato Markdown (los símbolos #, ## y -). El texto incluye la bienvenida, el estado de funcionamiento, y las características del servidor.

http://3.91.150.251:8082/README.md

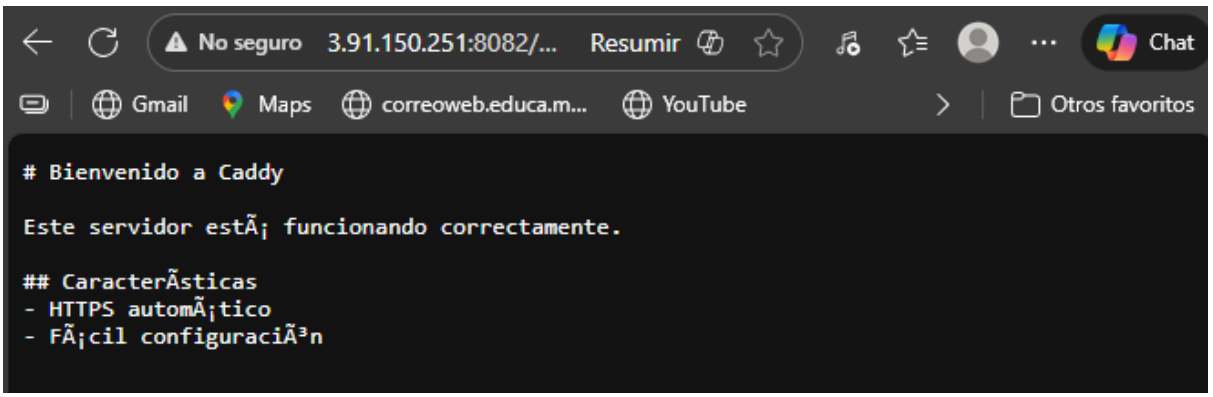


imagen 3.11

El navegador muestra el contenido plano del archivo `README.md`, incluyendo las marcas Markdown (#, ##, -) sin renderizarlas como HTML.

PARTE 4: CONFIGURACIÓN DE HTTPS CON CERTBOT EN APACHE

El objetivo es habilitar el protocolo HTTPS en el servidor Apache y configurarlo para que escuche en un puerto seguro (específicamente, el puerto 8443) utilizando un certificado autofirmado para fines de prueba.

1. Instalar Certbot y el plugin de Apache

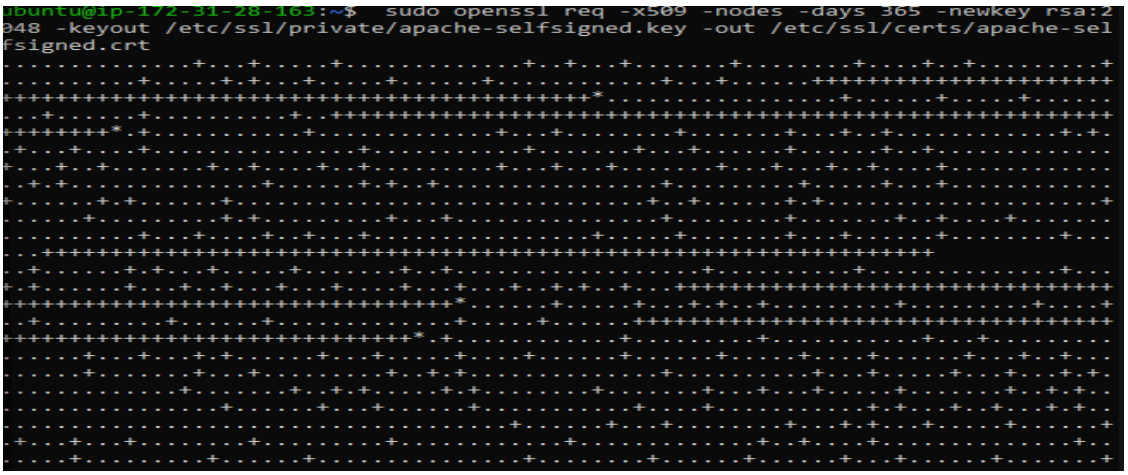
```
ubuntu@ip-172-31-28-163:~$  sudo apt install certbot python3-certbot-apache -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  augeas-lenses libaugeas0 python3-acme python3-augeas python3-certbot
  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 11 not upgraded.
Need to get 1657 kB of archives.
```

imagen 4.1

El proceso de instalación se inicia, identificando 12 paquetes nuevos a instalar, incluyendo `certbot`, `python3-certbot-apache`, y varias dependencias de Python

2. Verificar dominio o usar localhost

sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout /etc/ssl/private/apache-selfsigned.key -out /etc/ssl/certs/apache-selfsigned.crt



```
Country Name (2 letter code) [AU]:evelyn
String too long, must be at most 2 bytes long
Country Name (2 letter code) [AU]:evelyn
String too long, must be at most 2 bytes long
Country Name (2 letter code) [AU]:evelyn
String too long, must be at most 2 bytes long
Country Name (2 letter code) [AU]:evelyn
String too long, must be at most 2 bytes long
Country Name (2 letter code) [AU]:evelyn
String too long, must be at most 2 bytes long
Country Name (2 letter code) [AU]:es
State or Province Name (full name) [Some-State]:madrid
Locality Name (eg, city) []:madrid
Organization Name (eg, company) [Internet Widgits Pty Ltd]:MiLab
Organizational Unit Name (eg, section) []:IT
Common Name (e.g. server FQDN or YOUR name) []:3.91.150.251
Email Address []:evelyn@ejemplo.com
ubuntu@ip-172-31-28-163:~$
```

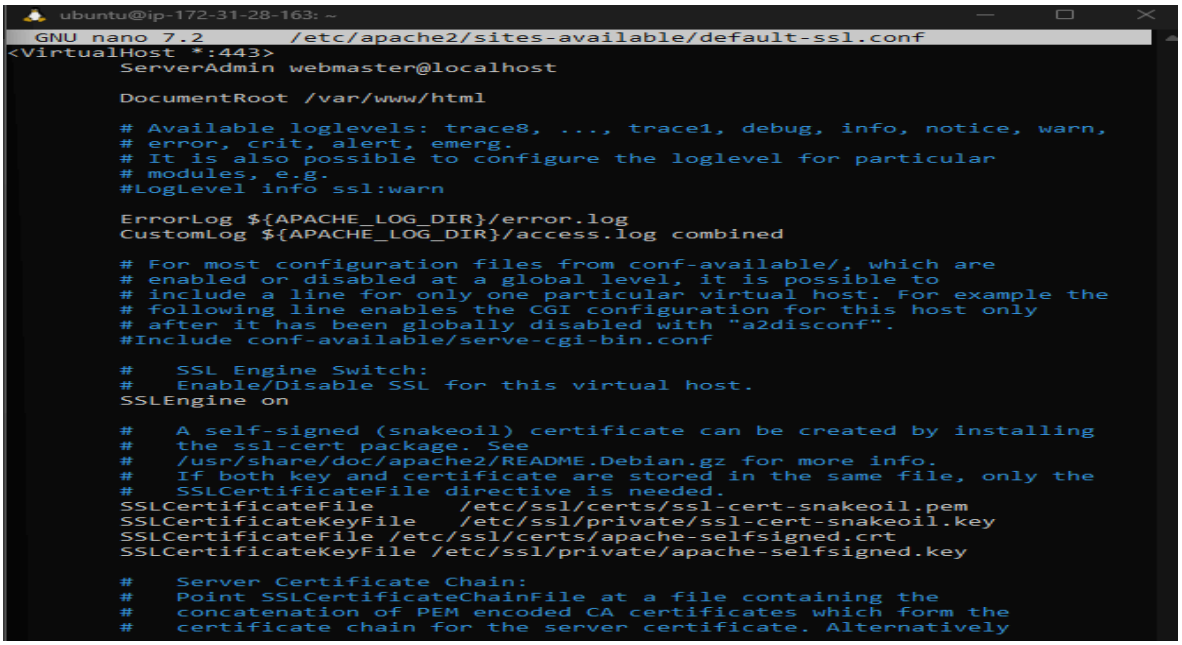
3. Habilitar módulo SSL en Apache

sudo a2enmod ssl

```
ubuntu@ip-172-31-28-163:~$  sudo a2enmod ssl
Considering dependency mime for ssl:
Module mime already enabled
Considering dependency socache_shmcb for ssl:
Enabling module socache_shmcb.
Enabling module ssl.
See /usr/share/doc/apache2/README.Debian.gz on how to configure SSL and create se
lf-signed certificates.
To activate the new configuration, you need to run:
  systemctl restart apache2
ubuntu@ip-172-31-28-163:~$
```

4. Crear configuración SSL para Apache

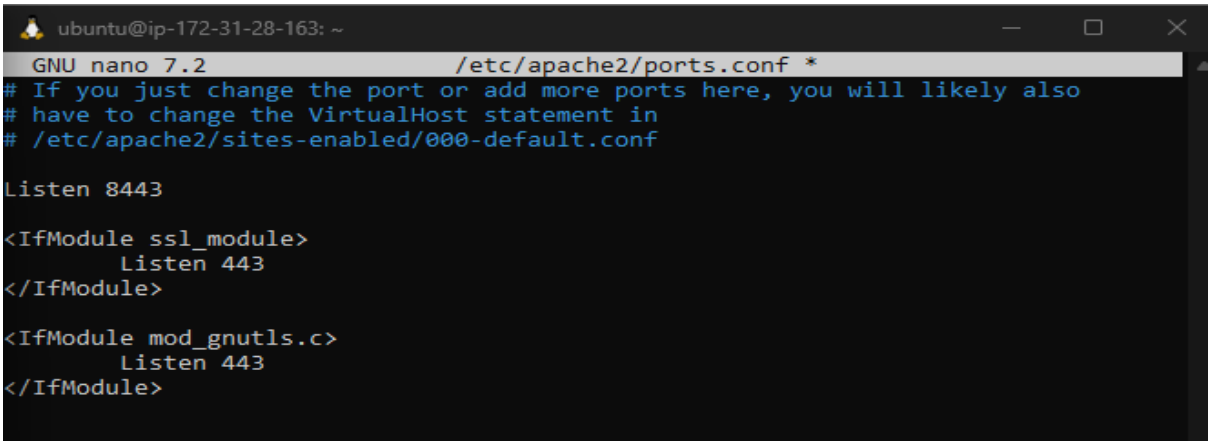
sudo nano /etc/apache2/sites-available/default-ssl.conf



El comando activa exitosamente el módulo **`ssl`** y su dependencia, **`socache_shmcb`**. Se muestra un mensaje indicando que para aplicar los cambios, es necesario **reiniciar Apache** (`systemctl restart apache2`).

5. Cambiar puerto SSL

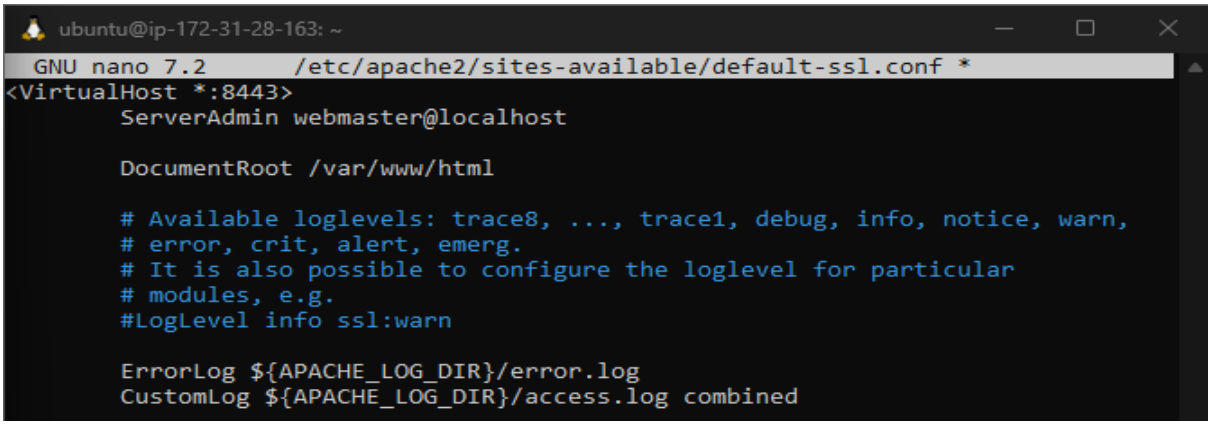
sudo nano /etc/apache2/ports.conf



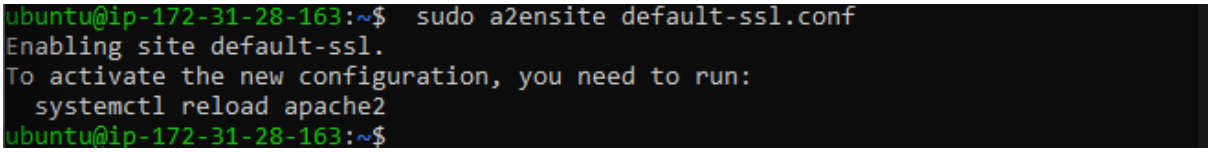
Configurar un puerto alternativo para las conexiones SSL/TLS (HTTPS) de Apache, ya que el puerto 443 estándar podría haber estado en conflicto o se quería reservar para otro uso o configuración.

6. Modificar VirtualHost SSL

sudo nano /etc/apache2/sites-available/default-ssl.conf

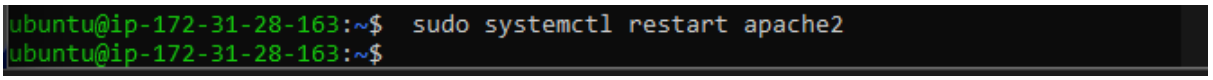


7. Habilitar sitio SSL



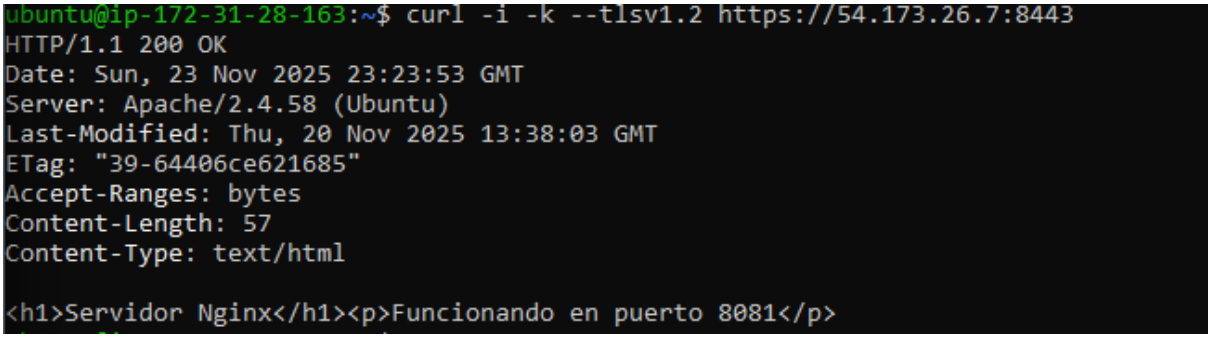
8. Reiniciar Apache

sudo systemctl restart apache2



9. Verificar HTTPS

curl -i -k https://localhost:8443

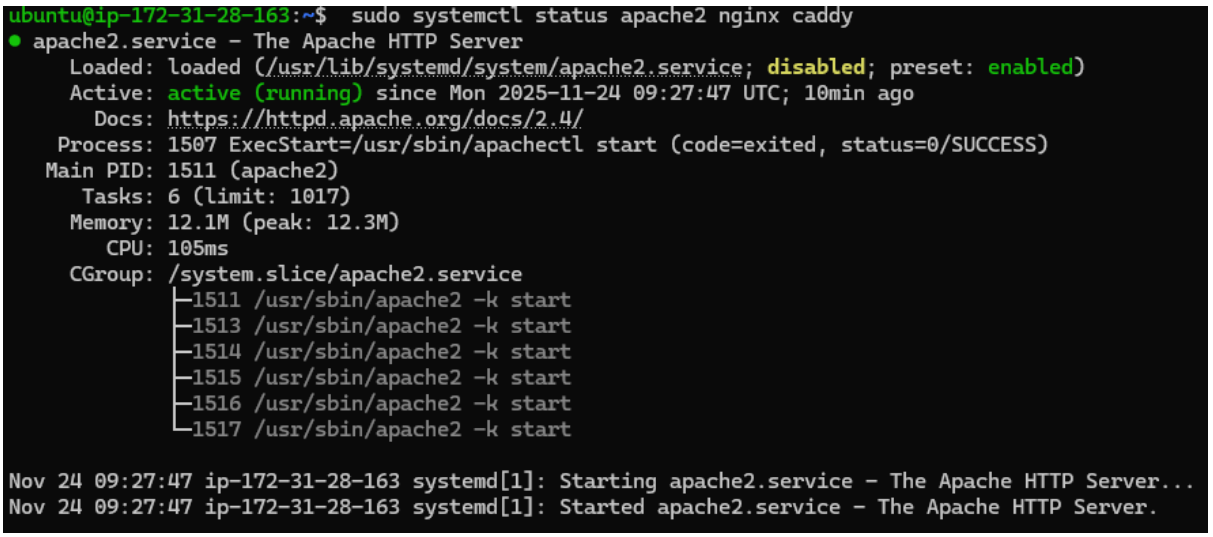


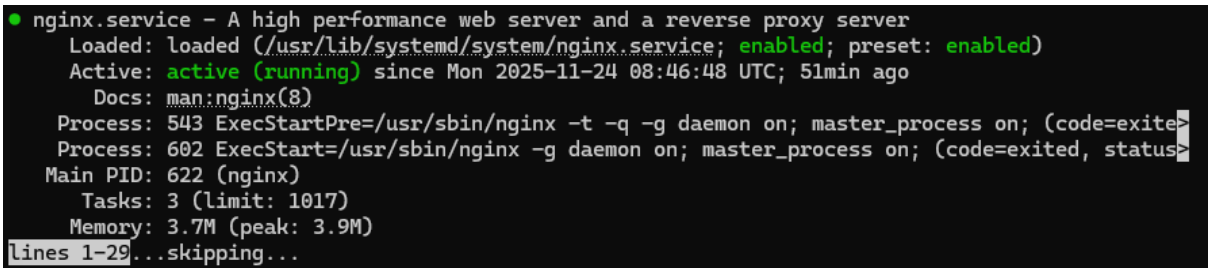
`HTTP/1.1 200 OK`: Confirma que la conexión SSL fue exitosa y Apache devolvió una respuesta correcta. `Server: Apache/2.4.58 (Ubuntu)`: Confirma que Apache es el servidor que responde.

PARTE 5: VERIFICACIÓN FINAL DE LOS TRES SERVIDORES

1. Verificar que todos los servicios están activos

sudo systemctl status apache2 nginx caddy





2. Verificar puertos en uso

sudo netstat -tulpn | grep -E '8080|8081|8082|8443'

```
ubuntu@ip-172-31-28-163:~$ sudo netstat -tulpn | grep -E '8080|8081|8082|8443'
tcp        0      0 0.0.0.0:8081            0.0.0.0:*               LISTEN      622/nginx: master
 p
tcp6       0      0 :::8443                 :::*                    LISTEN      1511/apache2

tcp6       0      0 :::8081                 :::*                    LISTEN      622/nginx: master
 p
tcp6       0      0 :::8082                 :::*                    LISTEN      530/caddy

ubuntu@ip-172-31-28-163:~$
```

3. Probar todos los servidores

curl http://localhost:8080

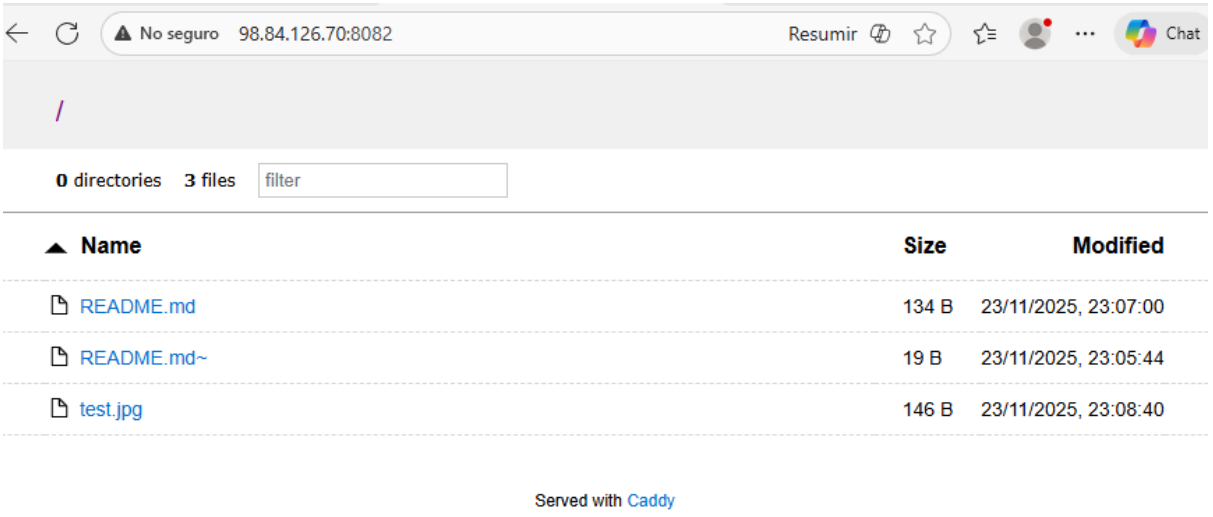
```
ubuntu@ip-172-31-28-163:~$ curl http://localhost:8080
curl: (7) Failed to connect to localhost port 8080 after 0 ms: Couldn't connect to server
```

curl http://localhost:8081

```
ubuntu@ip-172-31-28-163:~$  curl http://localhost:8081
<h1>Servidor Nginx</h1><p>Funcionando en puerto 8081</p>
```

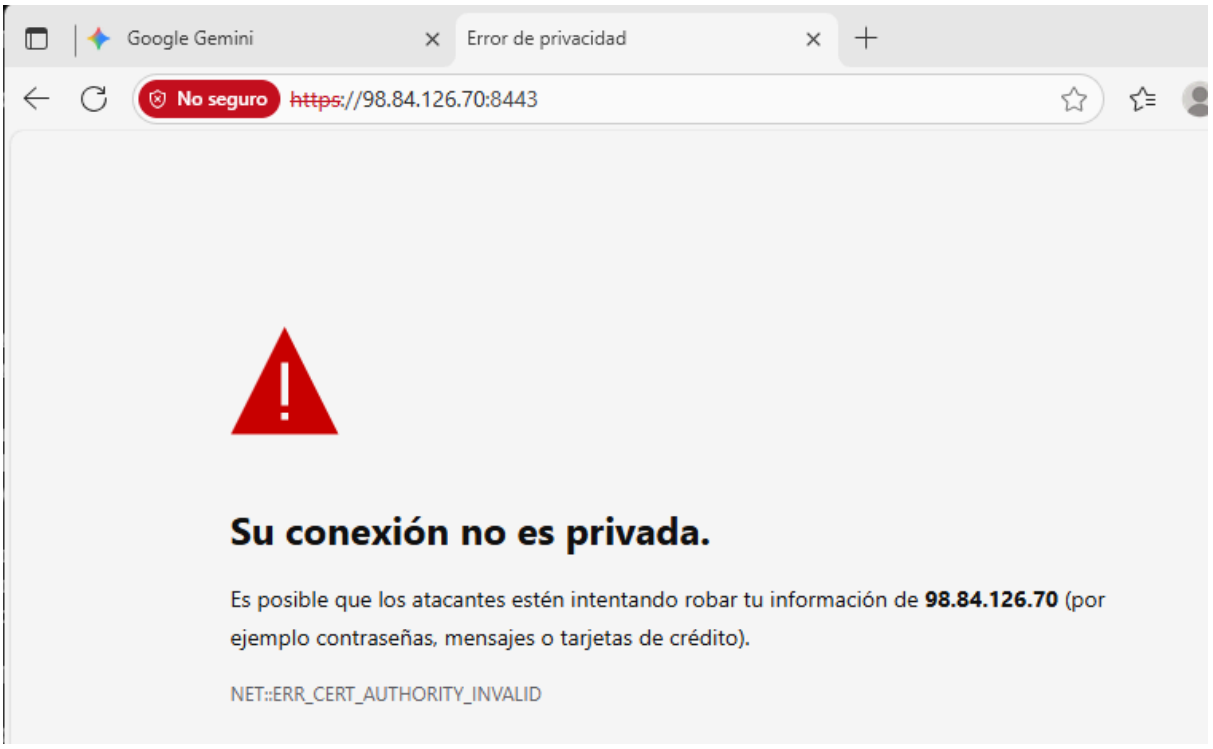
curl http://localhost:8082

```
ubuntu@ip-172-31-28-163:~$ curl http://localhost:8082
<!DOCTYPE html>
<html>

<head>
    <title>/</title>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <style>
        * {
            padding: 0;
            margin: 0;
        }

        body {
            font-family: sans-serif;
```



curl -k https://localhost:8443

```
ubuntu@ip-172-31-28-163:~$  curl -k https://localhost:8443
<h1>Servidor Nginx</h1><p>Funcionando en puerto 8081</p>
ubuntu@ip-172-31-28-163:~$
```



COMENTARIOS

El código de respuesta **`HTTP/1.1 200 OK`** confirma que el servidor Apache pudo establecer una conexión segura (SSL/TLS) a través del puerto **8443**. El uso del *flag* `-k` en `curl` permitió omitir el error de certificado autofirmado para completar la prueba de transporte seguro.