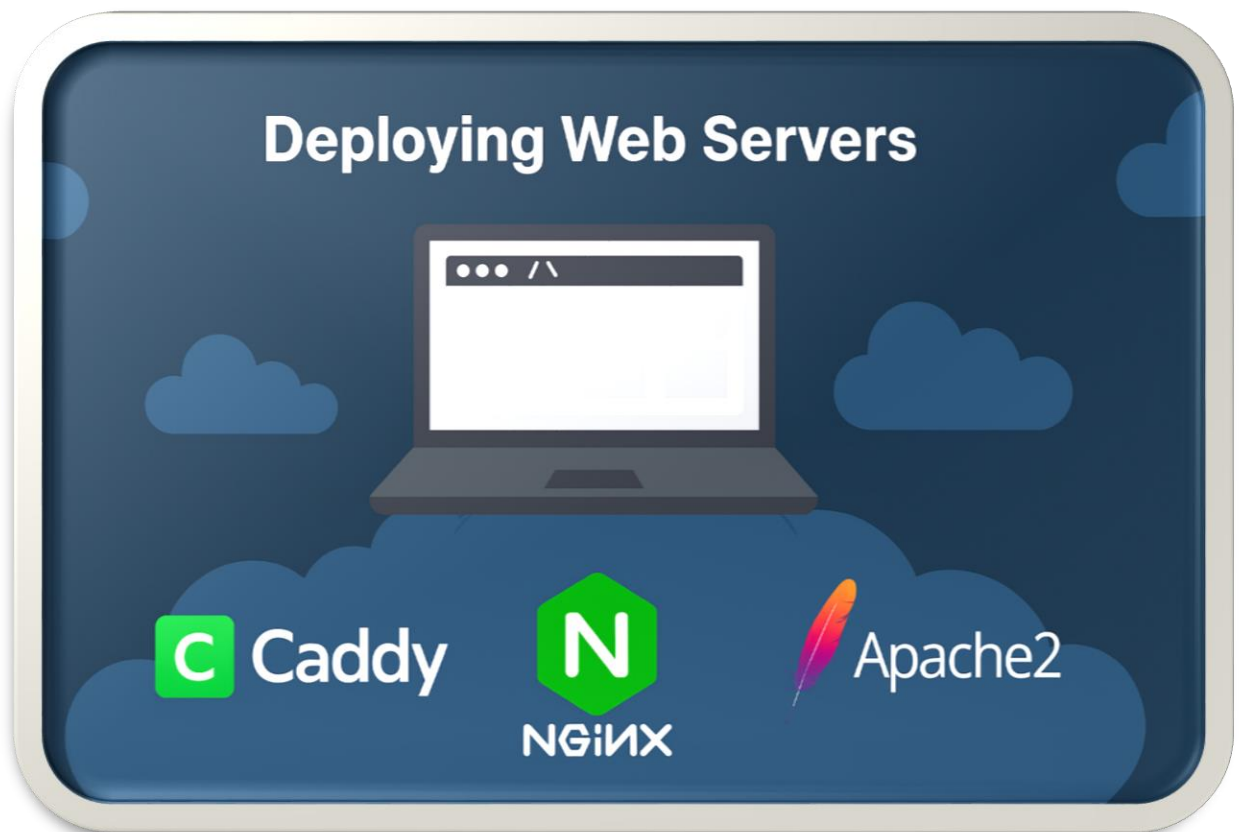


# CONFIGURACIÓN AVANZADA DE SERVIDORES WEB Y HTTPS

Hamza Akdi



ARQUITECTURA EN LA NUBE

13/10/2025

## Indice:

Indice: .....	2
1. Servidor Apache con PHP (Puerto 8080 y HTTPS en 8443) .....	3
1.1 Actualización del sistema .....	3
1.2 Instalación apache 2.....	3
1.3 Configurar Apache en puerto 8080 .....	3
1.4 Modificar el VirtualHost.....	4
1.5 Instalación de PHP y reinicio de Apache2. ....	4
1.6 Verificación estado de Apache. ....	4
1.7 Crear archivo PHP de prueba y verificación en terminal. ....	5
2. INSTALACIÓN Y CONFIGURACIÓN DE NGINX .....	6
2.1 Instalación Nginx.....	6
2.2 Configuración Nginx en puerto 8081.....	6
2.3 Creación de página HTML personalizada. ....	7
2.4 Reinicio y control de estado de NGINX. ....	7
2.5 Prueba Nginx desde la terminal. ....	8
3. Instalación y configuración de Caddy.....	8
3.1 Instalación de dependencias.....	8
3.2 Agregar repositorio de Caddy.....	8
3.3 Actualizar e instalar Caddy. ....	8
3.4 Creación de archivo Markdown de prueba (README.md). ....	9
3.5 Creación de imagen de prueba. ....	9
3.6 Creación de Caddyfile personalizado. ....	9
3.7 Reinicio de Caddy y verificación de estado. ....	10
3.8 Probar archivo Markdown.....	10
4. Configuración de HTTPS con Certbot en Apache. ....	10
4.1 Instalar Certbot y el plugin de Apache. ....	11
4.2 Verificar dominio o usar localhost.....	11
4.3 Habilitar módulo SSL en Apache y creación de configuración SSL para Apache. .....	11
4.4 Cambiar puerto SSL.....	12
4.5 Modificar VirtualHost SSL.....	12

4.6 Habilitar sitio SSL y verificación HTTPS.....	13
5.Verificación final de los tres servidores. ....	14
5.1 Verificar que todos los servicios están activos .....	14
5.2 Verificar puertos en uso. ....	15
5.3 Probar todos los servidores. ....	15

## 1. Servidor Apache con PHP (Puerto 8080 y HTTPS en 8443)

### 1.1 Actualización del sistema

Actualizaremos la lista de paquetes, que mejorará el sistema a las últimas versiones.

```
hamza@A6Alumno08:~$ sudo apt update && sudo apt upgrade -y
[sudo] password for hamza:
Get:1 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Hit:2 http://archive.ubuntu.com/ubuntu noble InRelease
Get:3 http://security.ubuntu.com/ubuntu noble-security/main amd64 Components [21.5 kB]
Get:4 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Components [52.2 kB]
Get:5 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Components [212 B]
Get:6 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Components [212 B]
Get:7 http://archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:8 http://archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:9 http://archive.ubuntu.com/ubuntu noble-updates/main amd64 Packages [1490 kB]
Get:10 https://download.docker.com/linux/ubuntu noble InRelease [48.5 kB]
Get:11 https://download.docker.com/linux/ubuntu noble/stable amd64 Packages [32.7 kB]
Get:12 http://archive.ubuntu.com/ubuntu noble-updates/main amd64 Components [175 kB]
Get:13 http://archive.ubuntu.com/ubuntu noble-updates/main amd64 c-n-f Metadata [15.3 kB]
Get:14 http://archive.ubuntu.com/ubuntu noble-updates/universe amd64 Packages [1487 kB]
```

### 1.2 Instalación apache 2

Instalación del servidor web Apache en el sistema.

```
hamza@A6Alumno08:~$ sudo apt install apache2 -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
apache2 is already the newest version (2.4.58-1ubuntu8.8).
The following package was automatically installed and is no longer required:
  libllvm19
Use 'sudo apt autoremove' to remove it.
0 upgraded, 0 newly installed, 0 to remove and 11 not upgraded.
```

### 1.3 Configurar Apache en puerto 8080

Abriremos el archivo de configuración de puertos. Cambia Listen 80 por Listen 8080

```
hamza@A6Alumno08:~$ sudo nano /etc/apache2/ports.conf
```

Archivo modificado al puerto 8080

```
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>
```

## 1.4 Modificar el VirtualHost.

Cambiaremos <VirtualHost \*:80> por <VirtualHost \*:8080> en el siguiente fichero:

```
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

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined

    # For most configuration files from conf-available/, which are
    # enabled or disabled at a global level, it is possible to
    # include a line for only one particular virtual host. For example the
    # following line enables the CGI configuration for this host only
```

## 1.5 Instalación de PHP y reinicio de Apache2.

Instalación de PHP y su módulo para funcionar con Apache.

```
hamza@A6Alumno08:~$ sudo apt install php libapache2-mod-php -y
```

Reiniciaremos Apache para aplicar los cambios

```
hamza@A6Alumno08:~$ sudo systemctl restart apache2
```

## 1.6 Verificación estado de Apache.

Comprobaremos que Apache está funcionando correctamente en el puerto 8080.

```
hamza@A6Alumno08:~$ 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 Thu 2025-10-09 14:43:03 CEST; 12s ago
     Docs: https://httpd.apache.org/docs/2.4/
   Process: 1988 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SUCCESS)
    Main PID: 1991 (apache2)
      Tasks: 6 (limit: 9350)
     Memory: 12.8M (peak: 14.4M)
        CPU: 30ms
    CGroup: /system.slice/apache2.service
            └─1991 /usr/sbin/apache2 -k start
              1993 /usr/sbin/apache2 -k start
              1994 /usr/sbin/apache2 -k start
              1995 /usr/sbin/apache2 -k start
              1996 /usr/sbin/apache2 -k start
              1997 /usr/sbin/apache2 -k start

Oct 09 14:43:03 A6Alumno08 systemd[1]: Starting apache2.service - The Apache HTTP Server...
Oct 09 14:43:03 A6Alumno08 systemd[1]: Started apache2.service - The Apache HTTP Server.
```

Para comprobar que corre por el puerto 8080

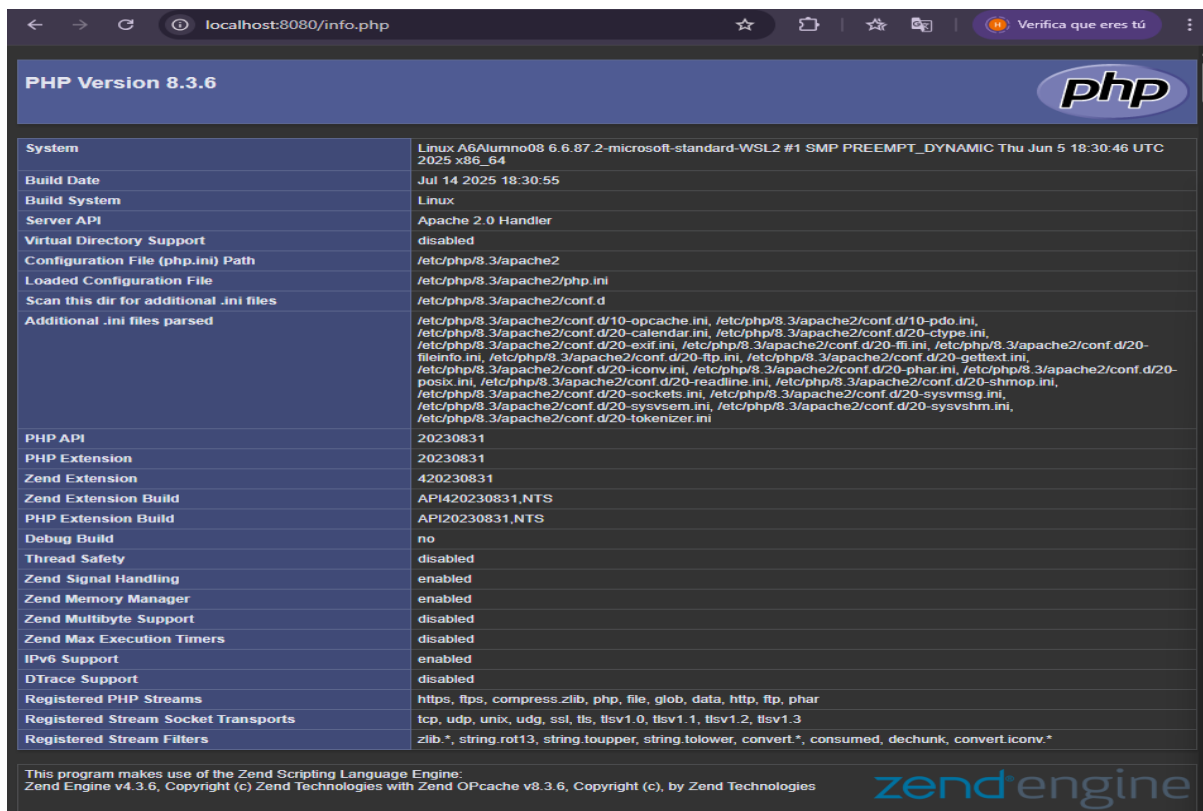
```
hamza@A6Alumno08:~$ sudo netstat -tulnp | grep 8080
tcp6      0      0 :::8080          :::*              LISTEN    1991/apache2
```

## 1.7 Crear archivo PHP de prueba y verificación en terminal.

Creación de un archivo que muestra información del PHP instalado

```
hamza@A6Alumno08:~$ echo "<?php phpinfo(); ?>" | sudo tee /var/www/html/info.php
<?php phpinfo(); ?>
```

Accederemos a <http://localhost:8080/info.php> para verificar.



PHP Version 8.3.6	
<b>System</b>	Linux A6Alumno08 6.6.87.2-microsoft-standard-WSL2 #1 SMP PREEMPT_DYNAMIC Thu Jun 5 18:30:46 UTC 2025 x86_64
<b>Build Date</b>	Jul 14 2025 18:30:55
<b>Build System</b>	Linux
<b>Server API</b>	Apache 2.0 Handler
<b>Virtual Directory Support</b>	disabled
<b>Configuration File (php.ini) Path</b>	/etc/php/8.3/apache2
<b>Loaded Configuration File</b>	/etc/php/8.3/apache2/php.ini
<b>Scan this dir for additional .ini files</b>	/etc/php/8.3/apache2/conf.d
<b>Additional .ini files parsed</b>	/etc/php/8.3/apache2/conf.d/10-opcache.ini, /etc/php/8.3/apache2/conf.d/10-pdo.ini, /etc/php/8.3/apache2/conf.d/20-calendar.ini, /etc/php/8.3/apache2/conf.d/20-ctype.ini, /etc/php/8.3/apache2/conf.d/20-exif.ini, /etc/php/8.3/apache2/conf.d/20-fileinfo.ini, /etc/php/8.3/apache2/conf.d/20-ftp.ini, /etc/php/8.3/apache2/conf.d/20-gettext.ini, /etc/php/8.3/apache2/conf.d/20-iconv.ini, /etc/php/8.3/apache2/conf.d/20-phar.ini, /etc/php/8.3/apache2/conf.d/20-posix.ini, /etc/php/8.3/apache2/conf.d/20-readline.ini, /etc/php/8.3/apache2/conf.d/20-shmop.ini, /etc/php/8.3/apache2/conf.d/20-sockets.ini, /etc/php/8.3/apache2/conf.d/20-sysvmsg.ini, /etc/php/8.3/apache2/conf.d/20-sysvsem.ini, /etc/php/8.3/apache2/conf.d/20-sysvshm.ini, /etc/php/8.3/apache2/conf.d/20-tokenizer.ini
<b>PHP API</b>	20230831
<b>PHP Extension</b>	20230831
<b>Zend Extension</b>	420230831
<b>Zend Extension Build</b>	API420230831,NTS
<b>PHP Extension Build</b>	API20230831,NTS
<b>Debug Build</b>	no
<b>Thread Safety</b>	disabled
<b>Zend Signal Handling</b>	enabled
<b>Zend Memory Manager</b>	enabled
<b>Zend Multibyte Support</b>	disabled
<b>Zend Max Execution Timers</b>	disabled
<b>IPv6 Support</b>	enabled
<b>DTrace Support</b>	disabled
<b>Registered PHP Streams</b>	https, ftps, compress.zlib, php, file, glob, data, http, ftp, phar
<b>Registered Stream Socket Transports</b>	tcp, udp, unix, udg, ssl, tls, tlsv1.0, tlsv1.1, tlsv1.2, tlsv1.3
<b>Registered Stream Filters</b>	zlib.*, string.rot13, string.toupper, string.tolower, convert.*, consumed, dechunk, convert.iconv.*

This program makes use of the Zend Scripting Language Engine:  
 Zend Engine v4.3.6, Copyright (c) Zend Technologies with Zend OPcache v8.3.6, Copyright (c), by Zend Technologies

Comprobación de Apache desde el terminal.

```
hamza@A6Alumno08:~$ 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;}
```

## 2. INSTALACIÓN Y CONFIGURACIÓN DE NGINX

### 2.1 Instalación Nginx.

```
hamza@A6Alumno08:~$ sudo apt install nginx -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
nginx is already the newest version (1.24.0-2ubuntu7.5).
The following package was automatically installed and is no longer required:
  libllvm19
Use 'sudo apt autoremove' to remove it.
0 upgraded, 0 newly installed, 0 to remove and 11 not upgraded.
```

### 2.2 Configuración Nginx en puerto 8081.

Abriremos la configuración por defecto y la cambiaremos de listen 80 por listen 8081.

```
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;
    #
    # Note: You should disable gzip for SSL traffic.
    # See: https://bugs.debian.org/773332
    #
    # Read up on ssl_ciphers to ensure a secure configuration.
    # See: https://bugs.debian.org/765782
    #
    # Self signed certs generated by the ssl-cert package
    # Don't use them in a production server!
    #
    # include snippets/snakeoil.conf;

    root /var/www/html;
```

### 2.3 Creación de página HTML personalizada.

Con este comando crearemos una página HTML con el siguiente mensaje:

```
root@A6Alumno08:/home/hamza# echo "<h1>Servidor Nginx</h1><p>Funcionando en puerto 8081</p>" | tee /var/www/html/index.html
```

## 2.4 Reinicio y control de estado de NGINX.

Reiniciaremos Nginx para aplicar los cambios de configuración.

```
hamza@A6Alumno08:~$ sudo systemctl restart nginx
```

A continuación, verificaremos el estado del servicio Nginx:

```
hamza@A6Alumno08:~$ 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 2025-10-14 09:15:58 CEST; 1min 25s ago
     Docs: man:nginx(8)
  Process: 1751 ExecStartPre=/usr/sbin/nginx -t -q -g daemon on; master_process on; (code=exited, status=0/SUCCESS)
  Process: 1753 ExecStart=/usr/sbin/nginx -g daemon on; master_process on; (code=exited, status=0/SUCCESS)
 Main PID: 1754 (nginx)
    Tasks: 17 (limit: 9350)
   Memory: 12.6M (peak: 14.6M)
      CPU: 53ms
 CGroup: /system.slice/nginx.service
          └─1754 "nginx: master process /usr/sbin/nginx -g daemon on; master_process on;"
            ├─1755 "nginx: worker process"
            ├─1756 "nginx: worker process"
            ├─1757 "nginx: worker process"
            ├─1758 "nginx: worker process"
            ├─1759 "nginx: worker process"
            ├─1760 "nginx: worker process"
            ├─1761 "nginx: worker process"
            ├─1762 "nginx: worker process"
            ├─1763 "nginx: worker process"
            ├─1764 "nginx: worker process"
            └─1766 "nginx: worker process"
```

## 2.5 Prueba Nginx desde la terminal.

Verificamos que funciona correctamente el servicio con el siguiente comando en la terminal.

```
hamza@A6Alumno08:~$ curl http://localhost:8081
<h1>Servidor Nginx</h1><p>Funcionando en puerto 8081</p>
```

## 3. Instalación y configuración de Caddy.

### 3.1 Instalación de dependencias.

Con el siguiente comando instalaremos las herramientas necesarias para añadir los repositorios.

```
hamza@A6Alumno08:~$ sudo apt install -y debian-keyring debian-archive-keyring apt-transport-https curl
```

### 3.2 Agregar repositorio de Caddy.

Añadiremos el repositorio oficial de Caddy con los siguientes comandos.

```
hamza@A6Alumno08:~$ curl -sLf 'https://dl.cloudsmith.io/public/caddy/stable/gpg.key' | sudo gpg --dearmor -o /usr/share/keyrings/caddy-stable-archive-keyring.gpg
hamza@A6Alumno08:~$ curl -sLf 'https://dl.cloudsmith.io/public/caddy/stable/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.cloudsmith.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
```

### 3.3 Actualizar e instalar Caddy.

Con el comando a continuación actualizaremos la lista de paquetes e instalaremos Caddy.

```
hamza@A6Alumno08:~$ sudo apt update && sudo apt install caddy -y
```

Creación de directorio para Caddy

```
hamza@A6Alumno08:~$ sudo mkdir -p /var/www/caddy
```



### 3.4 Creación de archivo Markdown de prueba (README.md).

```
hamza@A6Alumno08:~$ sudo mkdir -p /var/www/caddy
hamza@A6Alumno08:~$ echo "# Bienvenido a Caddy" | sudo tee /var/www/caddy/README.md
# Bienvenido a Caddy
hamza@A6Alumno08:~$ echo "" | sudo tee -a /var/www/caddy/README.md

hamza@A6Alumno08:~$ echo "Este servidor está funcionando correctamente." | sudo tee -a /var/www/caddy/README.md
Este servidor está funcionando correctamente.
hamza@A6Alumno08:~$ echo "" | sudo tee -a /var/www/caddy/README.md

hamza@A6Alumno08:~$ echo "## Características" | sudo tee -a /var/www/caddy/README.md
## Características
hamza@A6Alumno08:~$ echo "- Servidor moderno" | sudo tee -a /var/www/caddy/README.md
- Servidor moderno
hamza@A6Alumno08:~$ echo "- HTTPS automático" | sudo tee -a /var/www/caddy/README.md
- HTTPS automático
hamza@A6Alumno08:~$ echo "- Fácil configuración" | sudo tee -a /var/www/caddy/README.md
- Fácil configuración
hamza@A6Alumno08:~$ cat /var/www/caddy/README.md
# Bienvenido a Caddy

Este servidor está funcionando correctamente.

## Características
- Servidor moderno
- HTTPS automático
- Fácil configuración
hamza@A6Alumno08:~$ curl -o /tmp/test-image.jpg "https://www.python.org/static/apple-touchicon-144x144-precomposed.png"
```

### 3.5 Creación de imagen de prueba.

Tendremos en cuenta que si lo hacemos en WSL hay que hacer ajustes previos.

Descargaremos una imagen de prueba para verificar que Caddy sirve archivos estáticos.

```
hamza@A6Alumno08:~$ curl -o /tmp/test-image.jpg "https://www.python.org/static/apple-touchicon-144x144-precomposed.png"
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
           % Done    0       0             0                 0      0     0    0
100  146  100  146    0     0   805      0  --:--:-- --:--:-- --:--:--   806
```

```
hamza@A6Alumno08:~$ sudo mv /tmp/test-image.jpg /var/www/caddy/test.jpg
```

### 3.6 Creación de Caddyfile personalizado.

Con el siguiente comando abriremos el archivo de configuración de Caddy.

```
hamza@A6Alumno08:~$ sudo nano /etc/caddy/Caddyfile
```

Lo configuraremos de la siguiente manera:

```
GNU nano 7.2 /etc/caddy/Caddyfile

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

    # Enable the static file server.
    file_server browse

    # Another common task is to set up a reverse proxy:
    # reverse_proxy localhost:8080
    @markdown path *.md
    header @markdown Content-Type text/plain
    # Or serve a PHP site through php-fpm:
    # php_fastcgi localhost:9000
}
```

### 3.7 Reinicio de Caddy y verificación de estado.

Reiniciaremos Caddy para aplicar la nueva configuración.

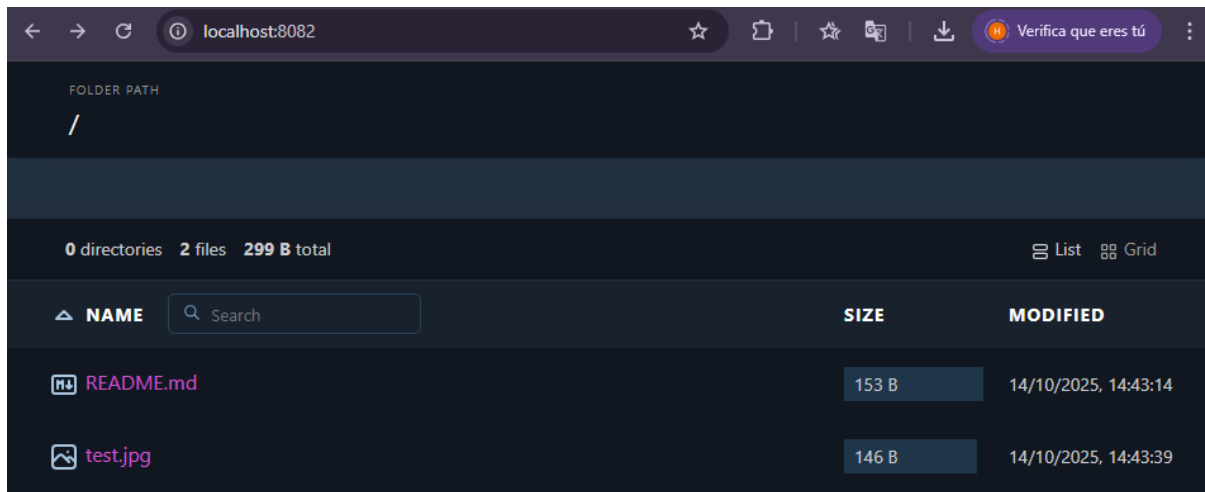
```
hamza@A6Alumno08:~$ sudo systemctl restart caddy
```

Comprobamos que el estado de Caddy está corriendo.

```
hamza@A6Alumno08:~$ sudo systemctl status caddy
● caddy.service - Caddy
   Loaded: loaded (/usr/lib/systemd/system/caddy.service; enabled; preset: enabled)
   Active: active (running) since Tue 2025-10-14 14:46:15 CEST; 5s ago
     Docs: https://caddyserver.com/docs/
   Main PID: 3035 (caddy)
    Tasks: 11 (limit: 9350)
   Memory: 13.3M (peak: 14.0M)
      CPU: 51ms
   CGroup: /system.slice/caddy.service
           └─3035 /usr/bin/caddy run --environ --config /etc/caddy/Caddyfile
```

Probaremos desde la terminal y desde el navegador web como a continuación.

```
hamza@A6Alumno08:~$ curl http://localhost:8082/
```



### 3.8 Probar archivo Markdown.

Verifica que Caddy sirve correctamente archivos Markdown.

```
hamza@A6Alumno08:~$ curl http://localhost:8082/README.md
# Bienvenido a Caddy

Este servidor está funcionando correctamente.

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

## 4. Configuración de HTTPS con Cerbot en Apache.

Cerbot es una herramienta automática que gestiona certificados SSL/TLS de Let's Encrypt. Permite obtener, instalar y renovar certificados sin intervención manual.

#### 4.1 Instalar Certbot y el plugin de Apache.

Con el siguiente comando instalaremos Certbot y su integración con Apache para gestionar certificados SSL.

```
hamza@A6Alumno08:~$ sudo apt install certbot python3-certbot-apache -y
```

## 4.2 Verificar dominio o usar localhost.

Para obtener certificados reales de Let's Encrypt necesitaremos un dominio público. Para esta práctica usaremos certificados autofirmados.

Utilizaremos el comando, el cual creara un certificado autofirmado para practicar HTTPS localmente.

```
hamza@A6Alumno08:~$ sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout /etc/ssl/private/apache-selfsigned.key -out /etc/ssl/certs/apache-selfsigned.crt
```

En el que iremos rellorando los campos que nos solicitan, pudiendo dejarlo por defecto.

[illegible]

### 4.3 Habilitar módulo SSL en Apache y creación de configuración SSL para Apache.

Activaremos el módulo SSL necesario para HTTPS en Apache.

```
hamza@A6Alumno08:~$ 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 self-signed certificate
s.
To activate the new configuration, you need to run:
systemctl restart apache2
```

Entraremos en la ruta con el siguiente comando, donde haremos la configuración SSL.

```
hamza@A6Alumno08:~$ sudo nano /etc/apache2/sites-available/default-ssl.conf
```

Editaremos el archivo y asegurándonos de que incluye estas líneas dentro de <VirtualHost \*:443>. En el Apartado marcado en blanco de la siguiente imagen:

```
GNU nano 7.2 /etc/apache2/sites-available/default-ssl.conf *
<VirtualHost *:443>
    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

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined

    # For most configuration files from conf-available/, which are
    # enabled or disabled at a global level, it is possible to
    # include a line for only one particular virtual host. For example the
    # following line enables the CGI configuration for this host only
    # after it has been globally disabled with "a2disconf".
    #Include conf-available/serve-cgi-bin.conf

    #
    # SSL Engine Switch:
    # Enable/Disable SSL for this virtual host.
    SSLEngine on

    #
    # A self-signed (snakeoil) certificate can be created by installing
    # the ssl-cert package. See
    # /usr/share/doc/apache2/README.Debian.gz for more info.
    # If both key and certificate are stored in the same file, only the
    # SSLCertificateFile directive is needed.
    SSLCertificateFile /etc/ssl/certs/apache-selfsigned.crt
    SSLCertificateKeyFile /etc/ssl/private/apache-selfsigned.key
```

## 4.4 Cambiar puerto SSL

Añadiremos la línea Listen 8443 para que Apache escuche HTTPS en 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 8080

<IfModule ssl_module>
    Listen 8443
</IfModule>

<IfModule mod_gnutls.c>
    Listen 8443
</IfModule>
```

## 4.5 Modificar VirtualHost SSL.

Iremos a la siguiente ruta, donde modificaremos el fichero.

```
hamza@A6Alumno08:~$ sudo nano /etc/apache2/sites-available/default-ssl.conf
```

Cambiaremos del primer párrafo en blanco <VirtualHost \*:443> a <VirtualHost \*:8443>

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

    DocumentRoot /var/www/html

    # Available loglevels: trace8, ..., tracel, 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

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined

    # For most configuration files from conf-available/, which are
    # enabled or disabled at a global level, it is possible to
    # include a line for only one particular virtual host. For example the
    # following line enables the CGI configuration for this host only
    # after it has been globally disabled with "a2disconf".
    #Include conf-available/serve-cgi-bin.conf

    # SSL Engine Switch:
    # Enable/Disable SSL for this virtual host.
    SSLEngine on

    # A self-signed (snakeoil) certificate can be created by installing
    # the ssl-cert package. See
    # /usr/share/doc/apache2/README.Debian.gz for more info.
    # If both key and certificate are stored in the same file, only the
    # SSLCertificateFile directive is needed.
    SSLCertificateFile /etc/ssl/certs/apache-selfsigned.crt
    SSLCertificateKeyFile /etc/ssl/private/apache-selfsigned.key
```

## 4.6 Habilitar sitio SSL y verificación HTTPS

Habilitaremos la configuración SSL en Apache.

```
hamza@A6Alumno08:~$ sudo a2ensite default-ssl.conf
```

Una vez activado el módulo SSL, nos pide que reiniciemos el servicio de Apache para activar la nueva configuración. El cual aplicará todos los cambios realizados anteriormente.

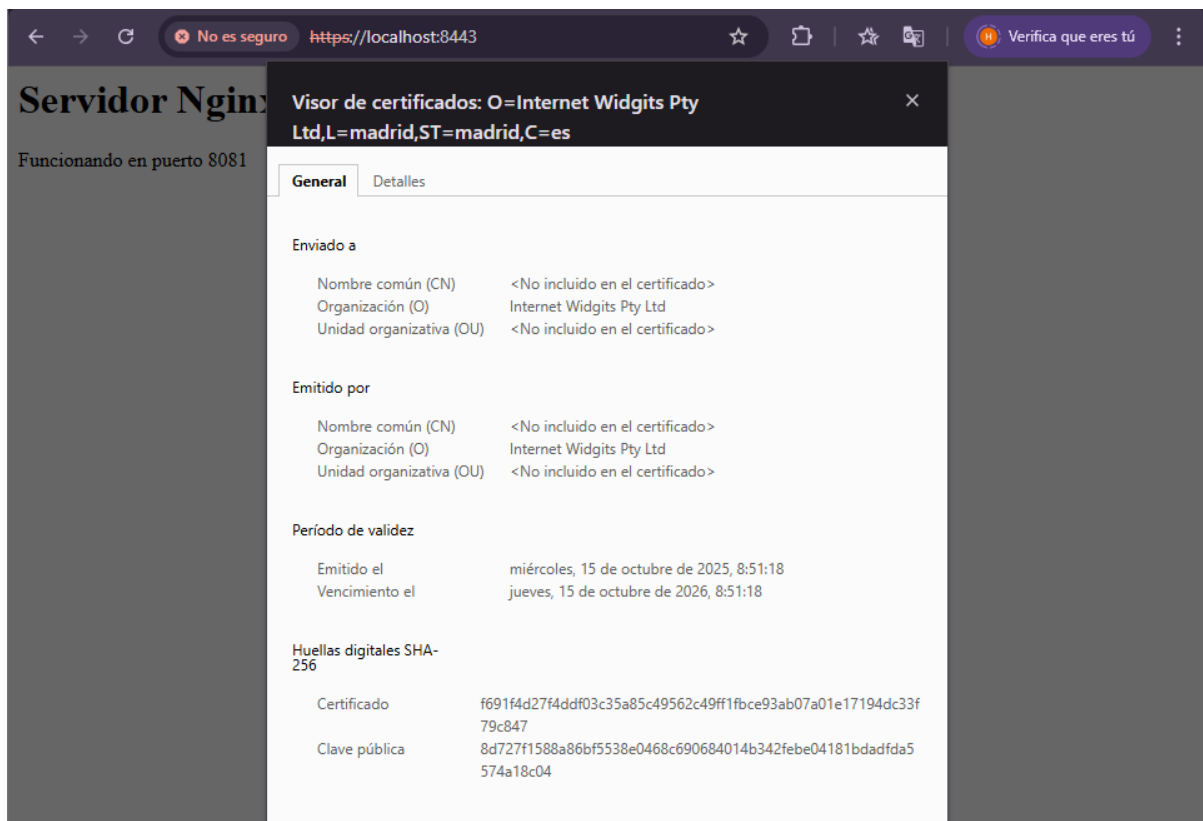
```
hamza@A6Alumno08:~$ sudo systemctl restart apache2
```

Con el siguiente comando verificaremos el funcionamiento en terminal.

Prueba la conexión HTTPS (el flag -k ignora el aviso del certificado autofirmado).

```
hamza@A6Alumno08:~$ curl -i -k https://localhost:8443
HTTP/1.1 200 OK
Date: Wed, 15 Oct 2025 07:18:58 GMT
Server: Apache/2.4.58 (Ubuntu)
Last-Modified: Tue, 14 Oct 2025 07:12:17 GMT
ETag: "39-641191aa60b28"
Accept-Ranges: bytes
Content-Length: 57
Content-Type: text/html

<h1>Servidor Nginx</h1><p>Funcionando en puerto 8081</p>
```



## 5.Verificación final de los tres servidores.

### 5.1 Verificar que todos los servicios están activos

Con este comando mostraremos el estado de los tres servidores simultáneamente.

```
hamza@A6Alumno08:~$ sudo systemctl status apache2 nginx caddy
```

Verificación de servicio activo de Apache:

```
● apache2.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/apache2.service; enabled; preset: enabled)
   Active: active (running) since Wed 2025-10-15 08:57:55 CEST; 37min ago
```

Verificación de servicio activo de 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 Wed 2025-10-15 08:41:52 CEST; 53min ago
     Docs: man:nginx(8)
   Main PID: 263 (nginx)
    Tasks: 17 (limit: 9350)
   Memory: 12.3M (peak: 15.5M)
      CPU: 37ms
```

Verificación de servicio activo de Caddy:

```
● caddy.service - Caddy
   Loaded: loaded (/usr/lib/systemd/system/caddy.service; enabled; preset: enabled)
   Active: active (running) since Wed 2025-10-15 08:41:53 CEST; 53min ago
     Docs: https://caddyserver.com/docs/
   Main PID: 180 (caddy)
    Tasks: 13 (limit: 9350)
   Memory: 43.0M (peak: 51.6M)
      CPU: 405ms
```

## 5.2 Verificar puertos en uso.

Lista los puertos donde están escuchando los servidores:

```
hamza@A6Alumno08:~$ sudo netstat -tulpn | grep -E '8080|8081|8082|8443'
tcp        0      0 0.0.0.0:8081          0.0.0.0:*        LISTEN      263/nginx: master p
tcp6       0      0 :::8443              :::*              LISTEN      1369/apache2
tcp6       0      0 :::8082              :::*              LISTEN      180/caddy
tcp6       0      0 :::8081              :::*              LISTEN      263/nginx: master p
tcp6       0      0 :::8080              :::*              LISTEN      1369/apache2
```

## 5.3 Probar todos los servidores.

Verificaremos que cada servidor responde correctamente en su puerto asignado.

```
hamza@A6Alumno08:~$ curl http://localhost:8080
<h1>Servidor Nginx</h1><p>Funcionando en puerto 8081</p>
hamza@A6Alumno08:~$ curl http://localhost:8081
<h1>Servidor Nginx</h1><p>Funcionando en puerto 8081</p>
hamza@A6Alumno08:~$ curl http://localhost:8082
<!DOCTYPE html>
<html>
```

```
hamza@A6Alumno08:~$ curl -k https://localhost:8443
<h1>Servidor Nginx</h1><p>Funcionando en puerto 8081</p>
```