

# TEMARIO: ARQUITECTURA EN LA NUBE

## - CONFIGURACIÓN AVANZADA DE SERVIDORES WEB Y HTTPS

### CONCEPTOS BÁSICOS PARA LA CLASE

#### Las Herramientas que Vamos a Usar

##### **Apache (*httpd*):**

Es el servidor web más veterano y popular del mundo. Como un chef experimentado, puede hacer de todo y es muy configurable. Lleva décadas funcionando en millones de sitios web.

##### **Nginx:**

Es el servidor web moderno y eficiente. Como un camarero rápido, puede atender muchas peticiones a la vez sin cansarse. Es perfecto para sitios con mucho tráfico.

##### **Caddy:**

Es el servidor web más nuevo y fácil de usar. Su superpoder es que configura **HTTPS** automáticamente, algo que con otros servidores requiere esfuerzo manual. Es como tener un mayordomo que lo hace todo por ti.

##### **Certbot:**

Es una herramienta que obtiene certificados **SSL/TLS** gratuitos de Let's Encrypt. Estos certificados son como el candado que ves en tu navegador cuando una página es segura (**HTTPS**).

#### ¿Qué es HTTPS y por qué es importante?

**HTTP** es como enviar postales: cualquiera puede leer lo que pones.

**HTTPS** es como enviar cartas en sobres cerrados con sello de lacre: nadie puede leer el contenido ni falsificarlo.

Cuando ves el candado en tu navegador, significa que la comunicación está cifrada y es segura.

#### ¿Por qué hacer esta práctica?

En el mundo real, las empresas no usan un solo servidor web. A veces tienen varios funcionando al mismo tiempo para diferentes propósitos:

- Uno para la página principal

- Otro para aplicaciones internas

- Otro como proxy para distribuir la carga

Además, **TODOS los sitios profesionales usan HTTPS**. Aprender a configurarlo es esencial para cualquier profesional de IT.

Al final de esta práctica comprenderás cómo funcionan los servidores web profesionales, cómo pueden coexistir en la misma máquina y cómo proteger las comunicaciones con **HTTPS**.

## COMANDOS Y PROCEDIMIENTOS

### PARTE 1: INSTALACIÓN Y CONFIGURACIÓN DE APACHE

#### 1. Actualizar el sistema

Comando:

```
sudo apt update && sudo apt upgrade -y
```

**Descripción:** Actualiza la lista de paquetes y mejora el sistema a las últimas versiones.

#### 2. Instalar Apache2

Comando:

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

**Descripción:** Instala el servidor web Apache en tu sistema.

#### 3. Configurar Apache en puerto 8080

Comando:

```
sudo nano /etc/apache2/ports.conf
```

**Descripción:** Abre el archivo de configuración de puertos. Cambia Listen 80 por Listen 8080.

```
# /etc/apache2/sites-enabled/000-default.conf

Listen 8080

<IfModule ssl_module>
    Listen 443
</IfModule>

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

#### 4. Modificar el VirtualHost

Comando:

```
sudo nano /etc/apache2/sites-available/000-default.conf
```

Descripción: Cambia <VirtualHost \*:80> por <VirtualHost \*:8080>.

```
GNU nano 7.2      /etc
<VirtualHost *:8080>
```

#### 5. Instalar PHP

Comando:

```
sudo apt install php libapache2-mod-php -y
root@ubuntu:/home/ChrisUser# apt install php libapache2-mod-php -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
php is already the newest version (2:8.3+93ubuntu2).
libapache2-mod-php is already the newest version (2:8.3+93ubuntu2).
The following package was automatically installed and is no longer required:
  liblvm19
Use 'sudo apt autoremove' to remove it.
0 upgraded, 0 newly installed, 0 to remove and 16 not upgraded.
```

Descripción: Instala PHP y su módulo para funcionar con Apache.

#### 6. Reiniciar Apache

Comando:

```
sudo systemctl restart apache2
```

Descripción: Reinicia Apache para aplicar los cambios.

```
root@ubuntu:/home/ChrisUser# systemctl restart apache2
root@ubuntu:/home/ChrisUser# systemctl status apache2
● apache2.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/apache2.service; enabled; preset: en>
   Active: active (running) since Fri 2025-10-10 07:03:29 UTC; 12s ago
     Docs: https://httpd.apache.org/docs/2.4/
 Process: 5048 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SU>
 Main PID: 5056 (apache2)
   Tasks: 6 (limit: 4603)
  Memory: 21.7M (peak: 22.5M)
    CPU: 82ms
   CGroup: /system.slice/apache2.service
           ├─5056 /usr/sbin/apache2 -k start
           ├─5059 /usr/sbin/apache2 -k start
           ├─5060 /usr/sbin/apache2 -k start
           ├─5061 /usr/sbin/apache2 -k start
           ├─5062 /usr/sbin/apache2 -k start
           └─5063 /usr/sbin/apache2 -k start

Oct 10 07:03:29 ubuntu systemd[1]: Starting apache2.service - The Apache HTTP S>
Oct 10 07:03:29 ubuntu apachectl[5050]: AH00558: apache2: Could not reliably de>
Oct 10 07:03:29 ubuntu svstemd[1]: Started apache2.service - The Apache HTTP Se>
```

## 7. Verificar estado de Apache

**Comando:**

```
sudo systemctl status apache2
```

**Descripción:** Comprueba que Apache está funcionando correctamente en el **puerto 8080**.

```
root@ubuntu:/home/ChrisUser# echo "<?php phpinfo(); ?>" | sudo tee /var/www/html/info.php
<?php phpinfo(); ?>
```

## 8. Crear archivo PHP de prueba

**Comando:**

```
echo "<?php phpinfo(); ?>" | sudo tee /var/www/html/info.php Descripción:
```

Crea un archivo que muestra información del PHP instalado.

```
root@ubuntu:/home/ChrisUser# 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;}
```

## 9. Probar Apache desde terminal

**Comando:**

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

**Descripción:** Verifica que Apache sirve correctamente el contenido PHP.

Hola espaÃ±a desde Nginx

Servidor funcionando correctamente amai

## PARTE 2: INSTALACIÓN Y CONFIGURACIÓN DE NGINX

### 1. Instalar Nginx

**Comando:**

```
sudo apt install nginx -y
```

**Descripción:** Instala el servidor web Nginx en tu sistema.

## 2. Configurar Nginx en puerto 8081

**Comando:**

```
sudo nano /etc/nginx/sites-available/default
```

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

    # SSL configuration
    #
```

**Descripción:** Abre la configuración por defecto. Cambia `listen 80` por `listen 8081`.

## 3. Crear página HTML personalizada

**Comando:**

```
echo "<h1>Servidor Nginx</h1><p>Funcionando en puerto 8081</p>" | sudo tee
/usr/share/nginx/html/index.html
```

**Descripción:** Crea una página HTML identifiable para Nginx.

```
root@ubuntu:/home/ChrisUser# nano /etc/nginx/sites-available/default
root@ubuntu:/home/ChrisUser# echo "<h1>Servidor Nginx</h1><p>Funcionando en puerto 8081
</p>" | tee /usr/share/nginx/html/index.html
<h1>Servidor Nginx</h1><p>Funcionando en puerto 8081</p>
```

## 4. Reiniciar Nginx

**Comando:**

```
sudo systemctl restart nginx
```

**Descripción:** Reinicia Nginx para aplicar los cambios de configuración.

```
root@ubuntu:/home/ChrisUser# systemctl restart nginx
root@ubuntu:/home/ChrisUser# 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 2025-10-10 07:16:01 UTC; 7s ago
     Docs: man:nginx(8)
 Process: 6250 ExecStartPre=/usr/sbin/nginx -t -q -g daemon on; master_process on;
 Process: 6253 ExecStart=/usr/sbin/nginx -g daemon on; master_process on; (code=exi
 Main PID: 6255 (nginx)
    Tasks: 5 (limit: 4603)
   Memory: 3.7M (peak: 4.5M)
      CPU: 17ms
     CGroup: /system.slice/nginx.service
             ├─6255 "nginx: master process /usr/sbin/nginx -g daemon on; master_proces
             ├─6256 "nginx: worker process"
             ├─6257 "nginx: worker process"
             ├─6258 "nginx: worker process"
             └─6259 "nginx: worker process"

Oct 10 07:16:01 ubuntu systemd[1]: Starting nginx.service - A high performance web serv
Oct 10 07:16:01 ubuntu systemd[1]: Started nginx.service - A high performance web serv
lines 1-19/19 (END)
```

## 5. Verificar estado de Nginx

Comando:

```
sudo systemctl status nginx
```

**Descripción:** Comprueba que Nginx está funcionando correctamente en el puerto 8081.

```
root@ubuntu:/home/ChrisUser# curl http://localhost:8081
<h1>Servidor Nginx</h1><p>Funcionando en puerto 8081</p>
```

## 6. Probar Nginx desde terminal

Comando:

```
curl http://localhost:8081
```

**Descripción:** Verifica que Nginx sirve correctamente el contenido HTML.

# 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
```

**Descripción:** Instala herramientas necesarias para añadir repositorios externos.

```
root@ubuntu:/home/ChrisUser# apt install -y debian-keyring debian-archive-keyring apt-transport-https curl
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
curl is already the newest version (8.5.0-2ubuntu10.6).
The following package was automatically installed and is no longer required:
  libllvm19
Use 'sudo apt autoremove' to remove it.
The following NEW packages will be installed:
  apt-transport-https debian-archive-keyring debian-keyring
0 upgraded, 3 newly installed, 0 to remove and 16 not upgraded.
Need to get 31.5 MB of archives.
After this operation, 33.4 MB of additional disk space will be used.
Get:1 http://es.archive.ubuntu.com/ubuntu noble-updates/universe amd64 apt-transport-https all 2.8.3 [3,970 B]
Get:2 http://es.archive.ubuntu.com/ubuntu noble/universe amd64 debian-archive-keyring all 2023.4ubuntu1 [168 kB]
Get:3 http://es.archive.ubuntu.com/ubuntu noble/universe amd64 debian-keyring all 2023.12.24 [31.3 MB]
Fetched 31.5 MB in 8s (4,095 kB/s)
Selecting previously unselected package apt-transport-https.
```

## 2. Agregar repositorio de Caddy

### Comando:

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

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

**Descripción:** Añade el repositorio oficial de Caddy a tu sistema.

```
root@ubuntu:/home/ChrisUser# curl -1sLf 'https://dl.cloudsmith.io/public/caddy/stable/gpg.key' | sudo gpg --dearmor -o /usr/share/keyrings/caddy-stable-archive-keyring.gpg
File '/usr/share/keyrings/caddy-stable-archive-keyring.gpg' exists. Overwrite? (y/N) y
```

## 3. Actualizar e instalar Caddy

### Comando:

```
sudo apt update && sudo apt install caddy -y
```

**Descripción:** Actualiza la lista de paquetes e instala Caddy.

```
root@ubuntu:/home/ChrisUser# curl -1sLf '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
```

## 4. Crear directorio para Caddy

Comando:

```
sudo mkdir -p /var/www/caddy
```

**Descripción:** Crea un directorio específico para los archivos de Caddy.

```
root@ubuntu:/home/ChrisUser# sudo apt update && sudo apt install caddy -y
Get:1 https://dl.cloudsmith.io/public/caddy/stable/deb/debian any-version InRelease [14
.8 kB]
Hit:2 http://es.archive.ubuntu.com/ubuntu noble InRelease
Hit:3 http://es.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:4 http://security.ubuntu.com/ubuntu noble-security InRelease
Hit:5 http://es.archive.ubuntu.com/ubuntu noble-backports InRelease
Get:6 https://dl.cloudsmith.io/public/caddy/stable/deb/debian any-version/main amd64 Pa
ckages [4,329 B]
Hit:7 https://repo.zabbix.com/zabbix/7.4/release/ubuntu noble InRelease
Hit:8 https://repo.zabbix.com/zabbix-tools/debian-ubuntu noble InRelease
Hit:9 https://repo.zabbix.com/zabbix/7.4/stable/ubuntu noble InRelease
Fetched 19.1 kB in 1s (14.7 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
16 packages can be upgraded. Run 'apt list --upgradable' to see them.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following package was automatically installed and is no longer required:
  libllvm19
Use 'sudo apt autoremove' to remove it.
The following NEW packages will be installed:
```

## 5. Crear archivo Markdown de prueba

Comando:

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

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

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

echo "## Características" | sudo tee -a /var/www/caddy/README.md echo "- Servidor
moderno" | sudo tee -a /var/www/caddy/README.md echo "- HTTPS automático" | sudo
tee -a /var/www/caddy/README.md echo "- Fácil configuración" | sudo tee -a
/var/www/caddy/README.md
```

**Descripción:** Crea un archivo Markdown con contenido de ejemplo.

```
root@ubuntu:/home/ChrisUser# mkdir -p /var/www/caddy
```

## 6. Crear imagen de prueba (*cuidado wsl hay que hacer ajustes*)

## Comando:

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

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

**Descripción:** Descarga una imagen de prueba para verificar que Caddy sirve archivos estáticos.

```
root@ubuntu:/home/ChrisUser# 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  213k      0 --:--:-- --:--:-- --:--:-- 218k
root@ubuntu:/home/ChrisUser#
```

```
GNU nano 7.2                               prueba.md
echo "# Bienvenido a Caddy" | sudo tee /var/www/caddy/README.md
echo "" | sudo tee -a /var/www/caddy/README.md
echo "Este servidor está funcionando correctamente." | sudo tee -a /var/www/caddy/README.md
echo "" | sudo tee -a /var/www/caddy/README.md
echo "## Características" | sudo tee -a /var/www/caddy/README.md
echo "- Servidor moderno" | sudo tee -a /var/www/caddy/README.md
echo "- HTTPS automático" | sudo tee -a /var/www/caddy/README.md
echo "- Fácil configuración" | sudo tee -a /var/www/caddy/README.md
```

## 7. Crear Caddyfile personalizado

### Comando:

```
sudo nano /etc/caddy/Caddyfile
```

**Descripción:** Abre el archivo de configuración de Caddy.

Escribe el siguiente contenido:

```
:8082 {
root * /var/www/caddy
file_server browse

@markdown path *.md
header @markdown Content-Type text/plain
}
```

```
: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  
  
    # Or serve a PHP site through php-fpm:  
    # php_fastcgi localhost:9000  
  
    @markdown path * .md  
    header @markdown Content-Type text/plain  
}  
}
```

## 8. Reiniciar Caddy

**Comando:**

```
sudo systemctl restart caddy
```

**Descripción:** Reinicia Caddy para aplicar la nueva configuración.

## 9. Verificar estado de Caddy

**Comando:**

```
sudo systemctl status caddy
```

**Descripción:** Comprueba que Caddy está funcionando en el puerto 8082.

```

root@ubuntu:/home/ChrisUser# systemctl restart caddy
root@ubuntu:/home/ChrisUser# systemctl status caddy
● caddy.service - Caddy
   Loaded: loaded (/usr/lib/systemd/system/caddy.service; enabled; preset: enabled)
   Active: active (running) since Fri 2025-10-10 08:07:32 UTC; 13s ago
     Docs: https://caddyserver.com/docs/
 Main PID: 9934 (caddy)
    Tasks: 8 (limit: 4603)
   Memory: 11.4M (peak: 12.1M)
      CPU: 89ms
     CGroup: /system.slice/caddy.service
             └─9934 /usr/bin/caddy run --environ --config /etc/caddy/Caddyfile

Oct 10 08:07:32 ubuntu caddy[9934]: {"level":"info","ts":1760083652.4989953,"logger":">>
Oct 10 08:07:32 ubuntu caddy[9934]: {"level":"info","ts":1760083652.4993832,"logger":">>
Oct 10 08:07:32 ubuntu caddy[9934]: {"level":"warn","ts":1760083652.499729,"logger": "h">>
Oct 10 08:07:32 ubuntu caddy[9934]: {"level":"warn","ts":1760083652.4997363,"logger": ">
Oct 10 08:07:32 ubuntu caddy[9934]: {"level":"info","ts":1760083652.4997566,"logger": ">
Oct 10 08:07:32 ubuntu caddy[9934]: {"level":"info","ts":1760083652.4999633,"msg": "aut">>
Oct 10 08:07:32 ubuntu caddy[9934]: {"level":"info","ts":1760083652.5001457,"msg": "ser">>
Oct 10 08:07:32 ubuntu systemd[1]: Started caddy.service - Caddy.
Oct 10 08:07:32 ubuntu caddy[9934]: {"level":"info","ts":1760083652.5054193,"logger":">>
Oct 10 08:07:32 ubuntu caddy[9934]: {"level":"info","ts":1760083652.505889,"logger": "t">>

```

## 10. Probar Caddy desde terminal

**Comando:**

```
curl http://localhost:8082/
```

**Descripción:** Lista los archivos disponibles en el servidor Caddy.

```

root@ubuntu:/home/ChrisUser# 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="68dd4da6-3c4b-4859-9ea9-29854aa2ed6a">
* { 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;
}

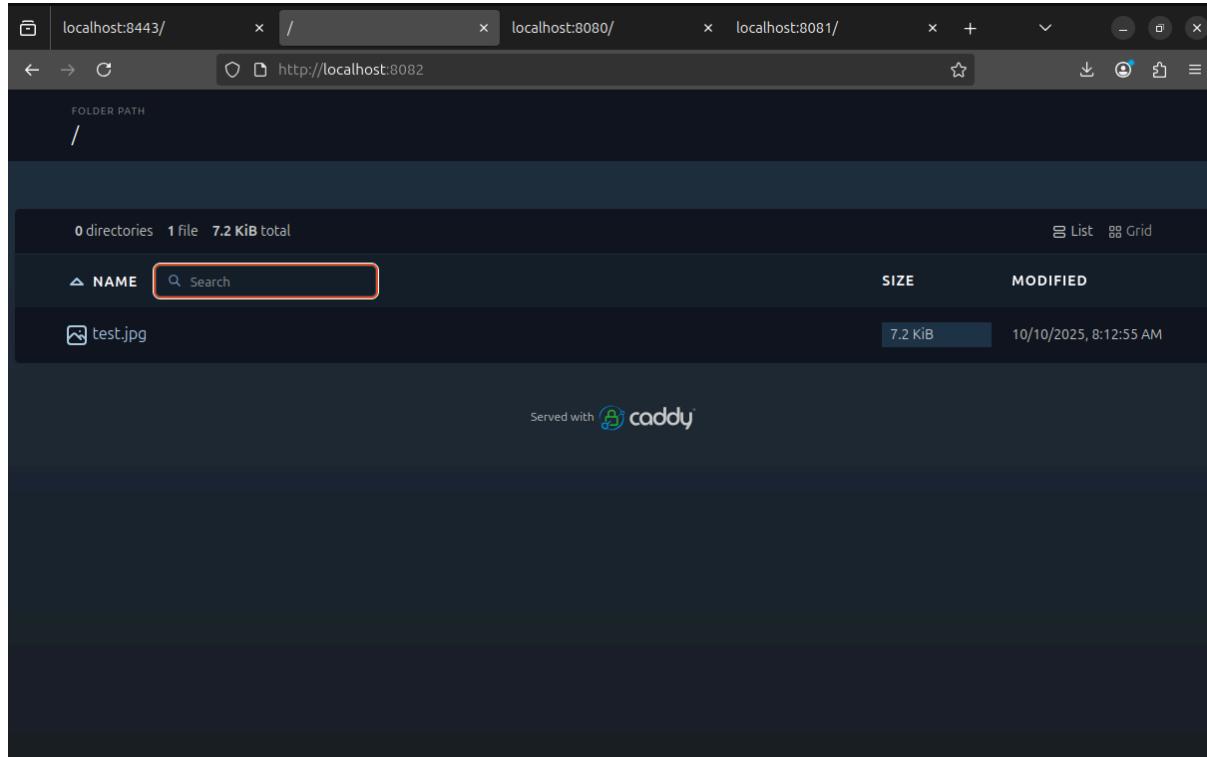
```

## 11. Probar archivo Markdown

**Comando:**

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

**Descripción:** Verifica que Caddy sirve correctamente archivos Markdown.



## PARTE 4: CONFIGURACIÓN DE HTTPS CON CERTBOT EN

### APACHE 1. Instalar Certbot y el plugin de Apache

**Comando:**

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

**Descripción:** Instala Certbot y su integración con Apache para gestionar certificados SSL.

```
root@ubuntu:/home/ChrisUser# apt install certbot python3-certbot-apache -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following package was automatically installed and is no longer required:
  libllvm19
Use 'sudo apt autoremove' to remove it.
The following additional packages will be installed:
  augeas-lenses libaugeas0 python3-acme python3-augeas python3-certbot
  python3-configargparse python3-icu python3-josepy python3-openssl
  python3-parsedatetime python3-rfc3339
Suggested packages:
  augeas-doc python-certbot-doc python3-certbot-nginx augeas-tools python-acme-doc
  python-certbot-apache-doc python-openssl-doc python3-openssl-dbg
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-openssl python3-parsedatetime python3-rfc3339
0 upgraded, 13 newly installed, 0 to remove and 16 not upgraded.
Need to get 1,705 kB of archives.
After this operation, 8,858 kB of additional disk space will be used.
Get:1 http://es.archive.ubuntu.com/ubuntu noble/universe amd64 augeas-lenses all 1.14.1-1build2 [323 kB]
Get:2 http://es.archive.ubuntu.com/ubuntu noble/universe amd64 libaugeas0 amd64 1.14.1-
```

## 2. Verificar dominio o usar localhost

**Nota:** Para obtener certificados reales de Let's Encrypt necesitas un dominio público. Para esta práctica usaremos certificados autofirmados.

**Comando:**

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

**Descripción:** Crea un certificado autofirmado para practicar HTTPS localmente. Completa los campos solicitados (*puedes usar valores por defecto*).

### 3. Habilitar módulo SSL en Apache

## Comando:

**sudo a2enmod ssl**

**Descripción:** Activa el módulo SSL necesario para HTTPS en Apache.

```
root@ubuntu:/home/ChrisUser# 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 certificates.
To activate the new configuration, you need to run:
  systemctl restart apache2
root@ubuntu:/home/ChrisUser#
```

## 4. Crear configuración SSL para Apache

## Comando:

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

**Descripción:** Edita el archivo y asegúrate de que incluye estas líneas dentro de <VirtualHost \*:443>:

## SSLEngine on

**SSLCertificateFile /etc/ssl/certs/apache-selfsigned.crt**

SSLCertificateKeyFile /etc/ssl/private/apache-selfsigned.key

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

## 5. Cambiar puerto SSL

Comando:

```
sudo nano /etc/apache2/ports.conf
```

**Descripción:** Añade la línea Listen 8443 para que Apache escuche HTTPS en puerto 8443.

```
root@ubuntu:/home/ChrisUser# systemctl reload apache2
root@ubuntu:/home/ChrisUser# a2ensite default-ssl.conf
Site default-ssl already enabled
```

```
● apache2.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/apache2.service; enabled; preset: enabled)
   Active: active (running) since Fri 2025-10-10 08:32:56 UTC; 2min 43s ago
     Docs: https://httpd.apache.org/docs/2.4/
 Process: 15667 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SUCCESS)
 Main PID: 15671 (apache2)
   Tasks: 6 (limit: 4603)
  Memory: 14.4M (peak: 15.1M)
    CPU: 85ms
   CGroup: /system.slice/apache2.service
           ├─15671 /usr/sbin/apache2 -k start
           ├─15673 /usr/sbin/apache2 -k start
           ├─15674 /usr/sbin/apache2 -k start
           ├─15675 /usr/sbin/apache2 -k start
           ├─15676 /usr/sbin/apache2 -k start
           └─15677 /usr/sbin/apache2 -k start

Oct 10 08:32:56 ubuntu systemd[1]: Starting apache2.service - The Apache HTTP Server.
Oct 10 08:32:56 ubuntu apachectl[15670]: AH00558: apache2: Could not reliably determi
Oct 10 08:32:56 ubuntu systemd[1]: Started apache2.service - The Apache HTTP Server.

● nginx.service - A high performance web server and a reverse proxy server
   Loaded: loaded (/usr/lib/systemd/system/nginx.service; enabled; preset: enabled)

```

## 6. Modificar VirtualHost SSL

Comando:

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

Descripción: Cambia <VirtualHost \*:443> por <VirtualHost \*:8443>.

```
Listen 8443
Listen 8080

<IfModule ssl_module>
    Listen 443
</IfModule>

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

## 7. Habilitar sitio SSL

Comando:

```
sudo a2ensite default-ssl.conf
```

Descripción: Activa la configuración SSL en Apache.

## 8. Reiniciar Apache

## Comando:

`sudo systemctl restart apache2`

**Descripción:** Aplica todos los cambios de configuración SSL.

## 9. Verificar HTTPS

## Comando:

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

**Descripción:** Prueba la conexión HTTPS (el flag -k ignora el aviso del certificado)

```


# Servidor Nginx



Funcionando en puerto 8081

root@ubuntu:/home/ChrisUser# sudo netstat -tulpn | grep -E '8080|8081|8082|8443'  
tcp        0      0 0.0.0.0:8081          0.0.0.0:*              LISTEN      1569  
nginx: master  
tcp6       0      0 :::8443            :::*                  LISTEN      1760  
apache2  
tcp6       0      0 :::8080            :::*                  LISTEN      1760  
apache2  
tcp6       0      0 :::8081            :::*                  LISTEN      1569  
nginx: master  
tcp6       0      0 :::8082            :::*                  LISTEN      1570  
taddy
```