EBENEZER IT · GUÍAS PASO A PASO

GUÍA 02 · EPISODIO 7

Nginx y certificados HTTPS con Let's Encrypt

Tu primera web en el servidor, servida por Nginx, con certificado gratuito, redirecciones correctas, cabeceras de seguridad y renovación automática comprobada. Entendiendo cada línea.

~2 horas
la primera vez, sin prisa

Requisito
Guía 01 terminada

Ubuntu 24.04 LTS
Nginx 1.24 · Certbot

www.ebenezer.org.es · Alejandro Quevedo · Versión pública 1.0 · noviembre 2026

Índice

Antes de empezar

0 Qué vas a construir y valores de esta guía

Parte A · Instalar Nginx y entenderlo

Paso 01 Comprobar el punto de partida

Paso 02 Abrir los puertos 80 y 443

Paso 03 Instalar Nginx

Paso 04 Cómo está organizado Nginx

Parte B · Tu primera web por HTTP

Paso 05 Crear la carpeta de la web

Paso 06 Crear el sitio (server block)

Paso 07 Activar el sitio y quitar el de ejemplo

Parte C · HTTPS con Let's Encrypt

Paso 08 Instalar Certbot

Paso 09 Obtener el certificado

Paso 10 Ajustes de TLS comunes

Paso 11 Configurar el sitio con HTTPS

Paso 12 Rechazar lo que no es tuyo (servidor por defecto)

Paso 13 Compresión

Paso 14 Renovación automática

Paso 15 Comprobación final e instantánea

Anexos

A Problemas frecuentes

B Puertos abiertos al terminar esta guía

C Chuleta de Nginx y Certbot

Cómo usar esta guía

Tienes aquí **todos los pasos y comandos**, en orden. La explicación de cada paso (el porqué de cada línea) está en el **vídeo del episodio**.

Para copiar los comandos sin errores, usa la versión web: **https://www.ebenezer.org.es/guia-02-nginx-https.html**. Al copiar desde un PDF se pueden romper los saltos de línea y las comillas.

ANTES DE EMPEZAR

Qué vas a construir

Al terminar, quien escriba `tudominio.es` en el navegador verá tu web con el candado de conexión segura. Por el camino vas a entender cómo piensa Nginx, porque es la pieza que usarás en casi todas las guías siguientes: WordPress, la VPN, el webmail, Grafana... todo se publica a través de él.

Pieza	Qué hace en esta guía
Nginx	El servidor web. Recibe las peticiones del navegador y responde con tus archivos (y, más adelante, reparte el tráfico hacia otras aplicaciones).
Let's Encrypt	Una autoridad de certificación gratuita. Firma un certificado que demuestra que el dominio es tuyo, para que el navegador confíe en tu web.
Certbot	El programa que pide el certificado a Let's Encrypt y lo renueva antes de que caduque (duran 90 días).

El recorrido de una visita

1. El navegador pregunta al DNS qué IP tiene `tudominio.es` y recibe `TU_IP` (registros A de la Guía 01, paso 3).
2. Se conecta al puerto **443** (HTTPS). Pasa el firewall de IONOS y después UFW.
3. Nginx recibe la conexión, presenta el certificado y mira el nombre que pidió el navegador (`server_name`) para saber qué sitio servir.
4. Busca el archivo en la carpeta del sitio (`root`) y lo devuelve cifrado.
5. Si alguien entra por **http://** (puerto 80), Nginx le contesta «vete a la versión https» (redirección 301).

Valores de esta guía

Los mismos de la Guía 01, más uno nuevo:

Valor	Qué es
`TU_DOMINIO`	Tu dominio sin `www`. Ejemplo: `ebenezer-lab.es`
`TU_IP`	La IP del VPS
`TU_USUARIO`	Tu usuario del servidor
`TU_CORREO`	Un correo real tuyo (Gmail sirve). Let's Encrypt lo usa para avisos importantes de tu cuenta

Recordatorio de la Guía 01

- Entras con `ssh lab`. Los bloques **verdes** van en el servidor y los **azules** en PowerShell de tu PC.
- Antes de empezar, haz una instantánea en IONOS si no la tienes (`01-servidor-base`).
- Para crear archivos usamos el patrón `sudo tee ... <<'EOF'`. En esta guía es **imprescindible** que `'EOF'` lleve comillas: los archivos de Nginx tienen variables como `$uri`, y sin comillas bash las borraría antes de escribir el archivo.

La regla de oro de Nginx: sudo nginx -t

Antes de recargar Nginx, **siempre** comprueba la configuración con `sudo nginx -t`. Si lo escribes sin `sudo` verás errores falsos de «Permission denied», porque un usuario normal no puede leer los certificados ni los registros. Si la comprobación falla, Nginx sigue funcionando con la configuración anterior: no se cae nada mientras no recargues.

PARTE A

Instalar Nginx y entenderlo

Abrimos las puertas de la web, instalamos Nginx y hacemos un recorrido por sus carpetas. Diez minutos entendiendo esto te ahorran horas de buscar por qué una web no carga.

Paso 1 Episodio 7

Comprobar el punto de partida

Por qué: Let's Encrypt va a comprobar desde Internet que el dominio apunta a tu servidor.

Desde tu PC:

EN TU PC · POWERSHELL

```
Resolve-DnsName TU_DOMINIO -Server 1.1.1.1
Resolve-DnsName www.TU_DOMINIO -Server 1.1.1.1
```

Esta vez **sin** `-Type A`: así ves todos los tipos de registro que responde, también los IPv6 (AAAA).

Comprueba antes de seguir

- Los dos nombres devuelven un registro **A** con `TU_IP`.
- **No** aparece ningún registro **AAAA**. Si aparece uno que no es de tu servidor (los de aparcamiento de IONOS), bórralo en el panel DNS: Let's Encrypt prefiere IPv6, iría a esa dirección y el certificado fallaría.
- En el servidor, `ssh lab` entra y `sudo ufw status` muestra solo el 22.

Paso 2 Episodio 7

Abrir los puertos 80 y 443

Por qué: el 443 es HTTPS, por donde irá toda tu web.

2.1 · Firewall de IONOS

En **Servidores & Cloud** → **Red** → **Políticas de firewall**, añade a la política de tu servidor:

Protocolo	Puerto	Origen	Descripción
TCP	80	Todos	HTTP: validación de Let's Encrypt y redirección a HTTPS
TCP	443	Todos	HTTPS: la web

2.2 · UFW

EN EL SERVIDOR

```
sudo ufw allow 80/tcp comment 'HTTP'
sudo ufw allow 443/tcp comment 'HTTPS'
sudo ufw status
```

¿Y el perfil «Nginx Full»?

Al instalar Nginx aparece un perfil de UFW llamado `Nginx Full` que abre 80 y 443 a la vez. Hace lo mismo; prefiero abrir los puertos por número porque así, al leer `ufw status`, ves exactamente qué está abierto sin tener que recordar qué incluye cada perfil.

Comprueba antes de seguir

`sudo ufw status` muestra 22, 80 y 443 (cada uno dos veces: IPv4 y v6) con sus comentarios. En IONOS están los tres.

Paso 3 Episodio 7

Instalar Nginx

Por qué: Nginx viene en los repositorios oficiales de Ubuntu: es una versión estable, la mantienen con parches de seguridad durante toda la vida de Ubuntu 24.04 y se actualiza sola con las actualizaciones automáticas de la Guía 01.

EN EL SERVIDOR

```
sudo apt update
sudo apt install -y nginx
systemctl status nginx --no-pager
nginx -v
```

SALIDA ESPERADA

```
● 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 ...
nginx version: nginx/1.24.0 (Ubuntu)
```

Pruébalo desde tu PC. Abre el navegador y ve a `http://TU_IP` (con `http`, no `https`). O desde PowerShell:

EN TU PC · POWERSHELL

```
curl.exe -I http://TU_IP
```

SALIDA ESPERADA

```
HTTP/1.1 200 OK
Server: nginx/1.24.0 (Ubuntu)
Content-Type: text/html
```

Comprueba antes de seguir

El navegador muestra la página **«Welcome to nginx!»** y `curl.exe -I` responde `200 OK`. Esa página de bienvenida la quitaremos en el paso 7.

Paso 4 Episodio 7

Cómo está organizado Nginx

Por qué: cuando algo falle, tendrás que saber dónde mirar.

EN EL SERVIDOR

```
ls /etc/nginx/
ls -l /etc/nginx/sites-available/ /etc/nginx/sites-enabled/
grep -n 'include' /etc/nginx/nginx.conf
```

Ruta	Qué hay	¿La tocas?
`/etc/nginx/nginx.conf`	La configuración principal: ajustes globales y las órdenes `include` que cargan todo lo demás	Casi nunca
`/etc/nginx/sites-available/`	Un archivo por cada sitio web que **tienes** (activo o no). Es el «almacén»	Sí: aquí creas tus sitios
`/etc/nginx/sites-enabled/`	Enlaces (accesos directos) a los sitios de `sites-available` que están **activos**. Nginx solo carga lo que hay aquí	Creas y borras enlaces
`/etc/nginx/conf.d/`	Ajustes generales en archivos `.conf` que se aplican a todos los sitios	Sí (paso 13)
`/etc/nginx/snippets/`	Trozos de configuración reutilizables, que se incluyen desde varios sitios	Sí (paso 10)
`/var/www/`	Donde viven los archivos de las webs (HTML, imágenes…)	Sí (paso 5)
`/var/log/nginx/`	Registros: `access.log` (cada visita) y `error.log` (los fallos)	Solo para leer

En la salida de `grep` verás estas dos líneas dentro de `nginx.conf`. Son las que conectan todo:

SALIDA ESPERADA

```
include /etc/nginx/conf.d/*.conf;
include /etc/nginx/sites-enabled/*;
```

¿Por qué dos carpetas, available y enabled?
Para poder **desactivar** un sitio sin borrar su configuración. Activar es crear un enlace en `sites-enabled` ; desactivar es borrar el enlace. El archivo de verdad sigue intacto en `sites-available` . En `ls -l` los enlaces se ven así: `default → /etc/nginx/sites-available/default` .

El vocabulario mínimo

Término	Qué es
Directiva	Una orden: un nombre, sus valores y un `;` al final. Ejemplo: `root /var/www/web;` . **Olvidar el punto y coma** es el error más común.
Bloque	Un grupo de directivas entre llaves `{ }` .
server	Un bloque que define **un sitio**: en qué puerto escucha, a qué nombres responde y qué sirve. También se le llama *server block* o *virtual host*.

Término	Qué es
location	Un bloque dentro de `server` que dice qué hacer con ciertas rutas (`/` , `/imagenes/` , `/.well-known/` ...).

Comprueba antes de seguir

Sabes ubicar `sites-available` , `sites-enabled` y los registros, y has visto el enlace `default` .

PARTE B

Tu primera web por HTTP

Creamos una página propia y un sitio de Nginx que responde a tu dominio. Todavía sin cifrar: primero hay que demostrar a Let's Encrypt que el dominio es tuyo, y eso se hace por HTTP.

Paso 5 Episodio 7

Crear la carpeta de la web

Por qué: cada sitio tiene su propia carpeta con sus archivos.

EN EL SERVIDOR

```
sudo mkdir -p /var/www/TU_DOMINIO /var/www/letsencrypt
sudo chown -R TU_USUARIO:TU_USUARIO /var/www/TU_DOMINIO
ls -ld /var/www/TU_DOMINIO /var/www/letsencrypt
```

SALIDA ESPERADA

```
drwxr-xr-x 2 TU_USUARIO TU_USUARIO 4096 ... /var/www/TU_DOMINIO
drwxr-xr-x 2 root       root       4096 ... /var/www/letsencrypt
```

Ahora crea la página de inicio. Como la carpeta es tuya, ya no hace falta `sudo`:

EN EL SERVIDOR

```
tee /var/www/TU_DOMINIO/index.html > /dev/null <<'EOF'
<!doctype html>
<html lang="es">
<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <title>Mi servidor</title>
</head>
<body style="font-family:sans-serif;max-width:40rem;margin:4rem auto;padding:0 1rem">
  <h1>Funciona</h1>
  <p>Esta página la sirve Nginx desde mi propio servidor Ubuntu.</p>
</body>
</html>
EOF
```

Comprueba antes de seguir

`cat /var/www/TU_DOMINIO/index.html` muestra la página.

Paso 6 Episodio 7

Crear el sitio (server block)

Por qué: ahora le decimos a Nginx: «cuando alguien pida `TU_DOMINIO` o `www.TU_DOMINIO` por el puerto 80, sírvele esta carpeta».

EN EL SERVIDOR

```
sudo tee /etc/nginx/sites-available/TU_DOMINIO > /dev/null <<'EOF'
server {
    listen 80;
    listen [::]:80;
    server_name TU_DOMINIO www.TU_DOMINIO;

    # Comprobaciones de Let's Encrypt
    location ^~ /.well-known/acme-challenge/ {
        root /var/www/letsencrypt;
    }

    root /var/www/TU_DOMINIO;
    index index.html;

    location / {
        try_files $uri $uri/ =404;
    }

    access_log /var/log/nginx/TU_DOMINIO.access.log;
    error_log  /var/log/nginx/TU_DOMINIO.error.log;
}
EOF
```

Cada directiva termina en punto y coma

Los bloques (`server {` , `location ... {`) no llevan `;` , pero todo lo que va dentro sí. Si falta uno, `nginx -t` te dirá la línea exacta; mira siempre **la línea anterior** a la que indica, porque ahí suele estar el que falta.

Comprueba antes de seguir

`cat /etc/nginx/sites-available/TU_DOMINIO` muestra el archivo con `$uri` escrito tal cual. Si en su lugar ves `try_files / =404;` , olvidaste las comillas de `'EOF'` : repite el comando.

Paso 7 Episodio 7

Activar el sitio y quitar el de ejemplo

Por qué: el sitio existe en el «almacén», pero aún no está activo.

EN EL SERVIDOR

```
sudo ln -s /etc/nginx/sites-available/TU_DOMINIO /etc/nginx/sites-enabled/
sudo rm /etc/nginx/sites-enabled/default
ls -l /etc/nginx/sites-enabled/
sudo nginx -t
sudo systemctl reload nginx
```

SALIDA ESPERADA

```
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
```

Prueba desde tu PC:

EN TU PC · POWERSHELL

```
curl.exe -I http://TU_DOMINIO
curl.exe -I http://www.TU_DOMINIO
```

Comprueba antes de seguir

- Los dos responden `HTTP/1.1 200 OK`.
- En el navegador, `http://TU_DOMINIO` muestra **«Funciona»**. Si sigues viendo la página de Nginx, recarga con **Ctrl+F5** (el navegador guarda copias de las páginas).
- El navegador marca la página como «No segura»: es lo esperado; lo arreglamos en la parte C.

PARTE C

HTTPS con Let's Encrypt

Pedimos el certificado, configuramos HTTPS nosotros mismos (sin que ningún programa toque la configuración a escondidas) y dejamos la renovación automática probada.

Paso 8 Episodio 7

Instalar Certbot

Por qué: Certbot es el cliente oficial de Let's Encrypt.

EN EL SERVIDOR

```
sudo apt install -y certbot
certbot --version
systemctl list-timers | grep certbot
```

SALIDA ESPERADA

```
certbot 2.9.0
... certbot.timer   certbot.service
```

¿Por qué no certbot --nginx, que lo hace todo solo?
Funciona, pero edita tus archivos de Nginx por su cuenta: añade líneas marcadas `# managed by Certbot` y un bloque de redirección poco legible. Para aprender (y para mantener servidores de clientes) es mejor que la configuración la escribas tú.

Comprueba antes de seguir
`certbot --version` responde y aparece `certbot.timer` en la lista.

Paso 9 Episodio 7

Obtener el certificado

Por qué: con el sitio del paso 6 funcionando por HTTP, ya podemos pedir el certificado para los dos nombres a la vez.

9.1 · Ensayo primero

Let's Encrypt limita cuántos certificados puedes pedir por semana para el mismo dominio. Si algo está mal y repites diez veces, te bloquea durante horas. Por eso primero se ensaya contra su servidor de pruebas, que no tiene límites:

EN EL SERVIDOR

```
sudo certbot certonly --webroot -w /var/www/letsencrypt -d TU_DOMINIO -d www.TU_DOMINIO --dry-run
```

La primera vez te preguntará el correo y si aceptas las condiciones. Responde con `TU_CORREO`, `Y` a las condiciones y `N` a compartir el correo con la EFF.

SALIDA ESPERADA

```
The dry run was successful.
```

Si falla, **no sigas**: mira el Anexo A. Los fallos casi siempre son DNS (paso 1), puertos (paso 2) o la ruta del paso 6.

9.2 · El certificado de verdad

EN EL SERVIDOR

```
sudo certbot certonly --webroot -w /var/www/letsencrypt -d TU_DOMINIO -d www.TU_DOMINIO --
cert-name TU_DOMINIO
```

SALIDA ESPERADA

```
Successfully received certificate.
Certificate is saved at: /etc/letsencrypt/live/TU_DOMINIO/fullchain.pem
Key is saved at:         /etc/letsencrypt/live/TU_DOMINIO/privkey.pem
This certificate expires on 2027-02-05.
```

9.3 · Qué ha creado

EN EL SERVIDOR

```
sudo ls -l /etc/letsencrypt/live/TU_DOMINIO/
```

Archivo	Qué es	Quién lo usa
`fullchain.pem`	Tu certificado + el certificado intermedio de Let's Encrypt, juntos	Nginx (`ssl_certificate`)
`privkey.pem`	La **clave privada**. Nunca sale del servidor ni se comparte	Nginx (`ssl_certificate_key`)
`cert.pem`	Solo tu certificado	Algunos programas antiguos
`chain.pem`	Solo el intermedio	Algunos programas antiguos

Verás que son enlaces (→) a `../../archive/TU_DOMINIO/fullchain1.pem` . En cada renovación Certbot crea `fullchain2.pem` , `3` … y mueve el enlace. Por eso en la configuración siempre se usa la ruta de `live/` : nunca cambia.

Let's Encrypt ya no avisa por correo de la caducidad

Desde 2025 dejó de enviar correos cuando un certificado va a caducar. El correo que diste sirve para avisos de la cuenta, pero vigilar la caducidad es cosa tuya: lo automatizaremos con Uptime Kuma en el episodio 23. Por eso el paso 14 (renovación probada) es tan importante.

Comprueba antes de seguir

`sudo certbot certificates` muestra tu certificado con los dos nombres y `VALID: 89 days` (o similar).

Paso 10 Episodio 7

Ajustes de TLS comunes

Por qué: TLS es el cifrado de HTTPS.

EN EL SERVIDOR

```
sudo tee /etc/nginx/snippets/tls.conf > /dev/null <<'EOF'
# Ebenezer: ajustes TLS comunes para todos los sitios
ssl_protocols TLSv1.2 TLSv1.3;
ssl_prefer_server_ciphers off;
ssl_session_cache shared:TLS:10m;
ssl_session_timeout 1d;
ssl_session_tickets off;
EOF
cat /etc/nginx/snippets/tls.conf
```

¿Y la lista de cifrados (ssl_ciphers)?

No la ponemos: con TLS 1.2 y 1.3 solamente, la lista por defecto de Nginx y OpenSSL 3 en Ubuntu 24.04 ya es segura, y escribirla a mano es una fuente de errores. Esta configuración sigue el perfil «intermedio» que recomienda Mozilla.

Comprueba antes de seguir

El archivo existe. Todavía no lo usa nadie: lo incluimos en el paso siguiente.

Paso 11 Episodio 7

Configurar el sitio con HTTPS

Por qué: ahora reescribimos el sitio entero en su forma definitiva: el puerto 80 solo sirve para Let's Encrypt y para mandar a todos a HTTPS; `www` redirige a la versión sin `www` (una sola dirección oficial, mejor para Google); y el sitio de verdad se sirve cifrado y con cabeceras de seguridad.

EN EL SERVIDOR

```
sudo tee /etc/nginx/sites-available/TU_DOMINIO > /dev/null <<'EOF'
# 1) HTTP: Let's Encrypt y redirección a HTTPS
server {
    listen 80;
    listen [::]:80;
    server_name TU_DOMINIO www.TU_DOMINIO;

    location ^~ /.well-known/acme-challenge/ {
        root /var/www/letsencrypt;
    }

    location / {
        return 301 https://TU_DOMINIO$request_uri;
    }
}

# 2) HTTPS con www: redirige a la versión sin www
server {
    listen 443 ssl http2;
    listen [::]:443 ssl http2;
    server_name www.TU_DOMINIO;

    ssl_certificate     /etc/letsencrypt/live/TU_DOMINIO/fullchain.pem;
    ssl_certificate_key /etc/letsencrypt/live/TU_DOMINIO/privkey.pem;
    include snippets/tls.conf;

    return 301 https://TU_DOMINIO$request_uri;
}

# 3) HTTPS: el sitio de verdad
server {
    listen 443 ssl http2;
    listen [::]:443 ssl http2;
    server_name TU_DOMINIO;

    ssl_certificate     /etc/letsencrypt/live/TU_DOMINIO/fullchain.pem;
    ssl_certificate_key /etc/letsencrypt/live/TU_DOMINIO/privkey.pem;
    include snippets/tls.conf;

    # Cabeceras de seguridad
    add_header Strict-Transport-Security "max-age=86400" always;
    add_header X-Content-Type-Options "nosniff" always;
    add_header X-Frame-Options "SAMEORIGIN" always;
    add_header Referrer-Policy "strict-origin-when-cross-origin" always;

    root /var/www/TU_DOMINIO;
    index index.html;

    location / {
        try_files $uri $uri/ =404;
    }

    access_log /var/log/nginx/TU_DOMINIO.access.log;
    error_log  /var/log/nginx/TU_DOMINIO.error.log;
```

```
}
EOF
```

Bloque 1 · HTTP

Bloque 2 · www

> **En Ubuntu 24.04, http2 va en la línea listen**
>
> Verás en Internet la directiva `http2 on;`. Esa forma existe a partir de Nginx 1.25, y Ubuntu 24.04 trae la 1.24: con ella, `nginx -t` da `unknown directive "http2"`. En la 1.24 se escribe `listen 443 ssl http2;`. Además, todos los bloques que escuchan en el 443 deben llevar las **mismas opciones**: si uno tiene `http2` y otro no, Nginx avisa con `protocol options redefined`.

Bloque 3 · el sitio

> **HSTS: empieza con poco tiempo**
>
> Con HSTS activo, si un día rompes el certificado, los navegadores que ya visitaron la web **no dejarán entrar** hasta que caduque el plazo, sin botón de «continuar de todos modos». En el laboratorio lo dejamos en 1 día. Cuando la web esté estable en producción, súbelo a un año: `max-age=31536000`.

> **add_header no se hereda si añades otro**
>
> Si dentro de un `location` pones algún `add_header`, ese `location` deja de recibir **todas** las cabeceras del bloque `server`, y tendrías que repetirlas. Es una trampa típica de Nginx; de momento no nos afecta.

Aplica y prueba

EN EL SERVIDOR

```
sudo nginx -t
sudo systemctl reload nginx
```

EN TU PC · POWERSHELL

```
curl.exe -I http://TU_DOMINIO
curl.exe -I https://www.TU_DOMINIO
curl.exe -I https://TU_DOMINIO
```

Petición	Respuesta correcta
`http://TU_DOMINIO`	`301` con `Location: https://TU_DOMINIO/`
`https://www.TU_DOMINIO`	`301` con `Location: https://TU_DOMINIO/`
`https://TU_DOMINIO`	`200 OK` (puede poner `HTTP/1.1` o `HTTP/2`: las dos valen) y las cuatro cabeceras de seguridad

> **Comprueba antes de seguir**
>
> - Las tres respuestas coinciden con la tabla.
> - En el navegador aparece el **candado**. Pulsa en él → «La conexión es segura» → «El certificado es válido» y verás que lo emitió **Let's Encrypt**.

Paso 12 Episodio 7

Rechazar lo que no es tuyo (servidor por defecto)

Por qué: si alguien entra poniendo la IP directamente, o con un dominio ajeno que apunte por error a tu IP, Nginx usa el **primer sitio que encuentra** y le sirve tu web.

EN EL SERVIDOR

```
sudo tee /etc/nginx/sites-available/00-por-defecto > /dev/null <<'EOF'
# Peticiones a nombres que no son míos (o a la IP a secas): sin respuesta
server {
    listen 80 default_server;
    listen [::]:80 default_server;
    server_name _;
    return 444;
}

server {
    listen 443 ssl http2 default_server;
    listen [::]:443 ssl http2 default_server;
    server_name _;
    ssl_reject_handshake on;
}
EOF
sudo ln -s /etc/nginx/sites-available/00-por-defecto /etc/nginx/sites-enabled/
sudo nginx -t
sudo systemctl reload nginx
```

EN TU PC · POWERSHELL

```
curl.exe -I http://TU_IP
curl.exe -I -k https://TU_IP
```

Comprueba antes de seguir

- `http://TU_IP` da `Empty reply from server` (es el 444).
- `https://TU_IP` da un error de TLS. En Windows suele ser `SEC_E_ILLEGAL_MESSAGE` o parecido: significa que el servidor cortó la conexión cifrada antes de empezar, que es lo que queremos.
- `https://TU_DOMINIO` sigue dando `200 OK`.

Paso 13 Episodio 7

Compresión

Por qué: los archivos de texto (HTML, CSS, JavaScript) se comprimen muy bien: un HTML de 40 KB viaja como 10 KB.

EN EL SERVIDOR

```
sudo tee /etc/nginx/conf.d/compresion.conf > /dev/null <<'EOF'
# Ebenezer: compresión gzip para los archivos de texto
gzip_vary on;
gzip_proxied any;
gzip_comp_level 5;
gzip_min_length 1024;
gzip_types text/css text/plain text/xml application/javascript application/json application/xml
image/svg+xml;
EOF
sudo nginx -t && sudo systemctl reload nginx
```

Si nginx -t dice «duplicate»

Significa que alguna de estas directivas ya está activa en `nginx.conf` (Ubuntu las trae comentadas con `#`, pero quizá alguien las descomentó). Mira con `grep -n gzip /etc/nginx/nginx.conf` y borra de `compresion.conf` la que esté repetida.

EN TU PC · POWERSHELL

```
curl.exe -s -o NUL -w "%{size_download} bytes\n" -H "Accept-Encoding: gzip" https://TU_DOMINIO
```

Comprueba antes de seguir

Con una página tan pequeña quizá no se note (está por debajo de 1 KB). Lo verás cuando subas una web real: compara el número con y sin la opción `-H`.

Paso 14 Episodio 7

Renovación automática

Por qué: el certificado dura 90 días.

14.1 · Recargar Nginx después de cada renovación

Certbot ejecuta los programas que haya en `/etc/letsencrypt/renewal-hooks/deploy/` cada vez que renueva un certificado con éxito:

EN EL SERVIDOR

```
sudo tee /etc/letsencrypt/renewal-hooks/deploy/recargar-nginx.sh > /dev/null <<'EOF'
#!/bin/sh
# Ebenezer: que Nginx use el certificado recién renovado
systemctl reload nginx
EOF
sudo chmod +x /etc/letsencrypt/renewal-hooks/deploy/recargar-nginx.sh
ls -l /etc/letsencrypt/renewal-hooks/deploy/
```

En `ls -l` el archivo debe empezar por `-rwxr-xr-x`: la `x` es el permiso de ejecución.

14.2 · Cómo recuerda Certbot el método

EN EL SERVIDOR

```
sudo cat /etc/letsencrypt/renewal/TU_DOMINIO.conf
```

SALIDA ESPERADA

```
[renewalparams]
account = ...
authenticator = webroot
webroot_path = /var/www/letsencrypt,
server = https://acme-v02.api.letsencrypt.org/directory
[[webroot_map]]
TU_DOMINIO = /var/www/letsencrypt
www.TU_DOMINIO = /var/www/letsencrypt
```

Certbot guardó cómo obtuvo el certificado y renovará de la misma forma. **No edites este archivo a mano.**

14.3 · Ensayo de renovación

EN EL SERVIDOR

```
sudo certbot renew --dry-run
```

SALIDA ESPERADA

```
Processing /etc/letsencrypt/renewal/TU_DOMINIO.conf
Simulating renewal of an existing certificate for TU_DOMINIO and www.TU_DOMINIO
Congratulations, all simulated renewals succeeded:
  /etc/letsencrypt/live/TU_DOMINIO/fullchain.pem (success)
```

El ensayo no ejecuta los scripts de deploy

En `--dry-run`, Certbot se salta los scripts de `deploy`, porque no hay certificado nuevo que usar. Para comprobar el tuyo a mano: `sudo /etc/letsencrypt/renewal-hooks/deploy/recargar-nginx.sh && echo OK`.

Comprueba antes de seguir

- `sudo certbot renew --dry-run` termina en `all simulated renewals succeeded`.
- El script de `deploy` tiene permiso de ejecución y funciona a mano.

Paso 15 Episodio 7

Comprobación final e instantánea

Por qué: comprobamos la web desde fuera con las mismas herramientas que usan los profesionales, revisamos los registros y guardamos el punto de partida de la siguiente guía.

15.1 · Nota de la configuración TLS

Abre **ssllabs.com/ssltest**, escribe TU_DOMINIO y marca «Do not show the results on the boards». Tarda un par de minutos.

Comprueba antes de seguir

Nota **A**. Para llegar a A+ hace falta HSTS de al menos 6 meses; lo subirás cuando la web sea definitiva (paso 11).

15.2 · Cabeceras de seguridad

En **securityheaders.com** analiza `https://TU_DOMINIO`. Verás las cuatro cabeceras en verde y un aviso por `Content-Security-Policy` y `Permissions-Policy`: son más avanzadas y dependen de cada web; las veremos cuando montes una web real.

15.3 · Los registros

EN EL SERVIDOR

```
sudo tail -n 5 /var/log/nginx/TU_DOMINIO.access.log
sudo tail -n 5 /var/log/nginx/TU_DOMINIO.error.log
```

SALIDA ESPERADA

```
203.0.113.7 - - [07/Nov/2026:19:42:10 +0000] "GET / HTTP/2.0" 200 312 "-" "Mozilla/5.0 (Windows NT 10.0; ...) Chrome/..."
```

El `error.log` debería estar vacío. Si tiene líneas, léelas: Nginx explica muy bien lo que falla.

15.4 · Repaso

Comprobación	Resultado correcto
`sudo nginx -t`	`syntax is ok` sin avisos (`warn`)
`http://TU_DOMINIO`	Redirige a `https://TU_DOMINIO/`
`https://www.TU_DOMINIO`	Redirige a `https://TU_DOMINIO/`
`https://TU_DOMINIO`	Tu página, con candado
`http://TU_IP`	Sin respuesta
`sudo certbot renew --dry-run`	`succeeded`
SSL Labs	A

15.5 · Instantánea

En IONOS, crea una instantánea con el nombre `02-nginx-https`.

Comprueba antes de seguir

Ya tienes una web segura en tu propio servidor. La siguiente guía instala Docker y usa este mismo Nginx como proxy inverso para publicar aplicaciones en contenedores, cada una con su subdominio y su certificado.

ANEXOS

Para cuando algo no sale

Primero `sudo nginx -t`, después el `error.log` del sitio. Con esas dos cosas se resuelve casi todo.

Anexo A · Problemas frecuentes

Nginx

Lo que ves	Qué pasa	Qué hacer
`nginx -t` sin sudo da `Permission denied`	Un usuario normal no puede leer los certificados ni abrir los registros	Usa siempre `sudo nginx -t`. No es un error de tu configuración
`unexpected "}"` o `expecting ";"`	Falta un punto y coma	Mira la línea que indica **y la anterior**
`unknown directive "http2"`	Has usado `http2 on;`, que no existe en Nginx 1.24	Ponlo en el `listen`: `listen 443 ssl http2;`
`protocol options redefined for 0.0.0.0:443`	Unos bloques del 443 tienen `http2` y otros no	Que todos los `listen 443` lleven las mismas opciones
`conflicting server name "..." on 0.0.0.0:80, ignored`	El mismo nombre está en dos bloques del mismo puerto (a menudo, dos archivos activos con el mismo sitio)	`grep -r "server_name" /etc/nginx/sites-enabled/` y deja solo uno
`a duplicate default server`	Hay dos `default_server` en el mismo puerto (por ejemplo, volviste a activar `default`)	`ls -l /etc/nginx/sites-enabled/` y quita el enlace sobrante
Sigue saliendo «Welcome to nginx!»	El enlace `default` sigue activo, o el navegador muestra una copia guardada	Paso 7, y recarga con **Ctrl+F5**
403 Forbidden	Nginx no puede leer la carpeta o no hay `index.html`	`ls -l /var/www/TU_DOMINIO/`: tiene que existir `index.html` y los permisos de «otros» tener `r`
404 Not Found	El archivo no está donde dice `root`	Comprueba la ruta de `root` y el nombre exacto del archivo (mayúsculas incluidas)
Un cambio no se nota	No recargaste Nginx, o es la caché del navegador	`sudo systemctl reload nginx` y **Ctrl+F5**, o prueba con `curl.exe`

Certbot y Let's Encrypt

Lo que ves	Qué pasa	Qué hacer
`Timeout during connect (likely firewall problem)`	Let's Encrypt no llega a tu puerto 80	Revisa el 80 en IONOS y en UFW (paso 2)
`Invalid response from http://.../.well-known/acme-challenge/...: 404`	Llega a tu servidor, pero no encuentra el archivo de prueba	Revisa el `location ^~ /.well-known/acme-challenge/` y que `-w` sea `/var/www/letsencrypt`, la misma carpeta que su `root`

Lo que ves	Qué pasa	Qué hacer
El error muestra una IP que no es la tuya	El DNS apunta a otro sitio, a menudo un registro AAAA de IONOS	Paso 1: borra los AAAA y los A que no sean `TU_IP`
`NXDOMAIN looking up A for ...`	Ese nombre no existe en el DNS	Crea el registro A o espera a que se propague
`too many certificates already issued` / `too many failed authorizations`	Has chocado con los límites de Let's Encrypt	Espera (normalmente 1 hora para los fallos, una semana para los emitidos) y ensaya siempre con `--dry-run`
La web dice que el certificado caducó pero Certbot dice que lo renovó	Nginx no se recargó y sigue usando el viejo en memoria	`sudo systemctl reload nginx` y revisa el script del paso 14.1

HSTS te impide entrar

Si rompiste el certificado y el navegador no te deja pasar por HSTS: arregla el certificado (es la solución real), o espera a que venza el `max-age`. En Chrome puedes borrar la memoria HSTS de un dominio en `chrome://net-internals/#hsts` → «Delete domain security policies».

Anexo B · Puertos abiertos al terminar esta guía

Puerto	Servicio	Firewall de IONOS	UFW	Qué lo protege
22/TCP	SSH	Abierto	Abierto	Solo clave, sin root, `AllowUsers`, fail2ban (Guía 01)
80/TCP	HTTP (Nginx)	Abierto	Abierto	Solo sirve los archivos de Let's Encrypt y redirige; el resto, 444
443/TCP	HTTPS (Nginx)	Abierto	Abierto	TLS 1.2/1.3, cabeceras de seguridad, rechazo de nombres ajenos

```
EN EL SERVIDOR
sudo ss -tlpn | grep -E ':(22|80|443) '
```

Debes ver `sshd` en el 22 y `nginx` en el 80 y el 443, cada uno en `0.0.0.0` (IPv4) y `[::]` (IPv6).

Anexo C · Chuleta de Nginx y Certbot

Para...	Comando
Comprobar la configuración	`sudo nginx -t`
Aplicar cambios sin cortar	`sudo systemctl reload nginx`
Ver la configuración completa que carga Nginx	`sudo nginx -T | less` (sal con **q**)
Activar un sitio	`sudo ln -s /etc/nginx/sites-available/SITIO /etc/nginx/sites-enabled/`
Desactivar un sitio	`sudo rm /etc/nginx/sites-enabled/SITIO`
Ver visitas en directo	`sudo tail -f /var/log/nginx/TU_DOMINIO.access.log`
Ver errores recientes	`sudo tail -n 30 /var/log/nginx/TU_DOMINIO.error.log`
Ver tus certificados y su caducidad	`sudo certbot certificates`
Ensayar la renovación	`sudo certbot renew --dry-run`
Añadir un nombre a un certificado	`sudo certbot certonly --webroot -w /var/www/letsencrypt --cert-name TU_DOMINIO -d TU_DOMINIO -d www.TU_DOMINIO -d NUEVO.TU_DOMINIO` (hay que repetir **todos** los nombres)
Cabeceras de una web	`curl.exe -I https://TU_DOMINIO`
Fecha de caducidad vista desde fuera	`echo | openssl s_client -connect TU_DOMINIO:443 -servername TU_DOMINIO 2>/dev/null | openssl x509 -noout -dates`

Si te atascas

Pega en la conversación: el comando, el error completo, la salida de `sudo nginx -t` y el contenido de `/etc/nginx/sites-available/TU_DOMINIO`. Con eso se ve el problema a la primera.