Servidores web Apache y Nginx

Instalación y configuración en Debian

📧 José Domingo Muñoz

🏫 IES Gonzalo Nazareno · Dos Hermanas

📚 SRI · Servicios de Red e Internet

01

Apache HTTP Server

Instalación y estructura de configuración

Instalación de Apache

```
sudo apt update
sudo apt install apache2
```

TRAS LA INSTALACIÓN

- El servicio queda **levantado y habilitado** automáticamente
- Se gestiona con `systemctl` (`apache2.service`)
- Escucha por defecto en el puerto **80** (HTTP) y **443** (HTTPS)
- Sirve contenido desde `/var/www/html/` con la página de bienvenida de Debian

```
systemctl status apache2
systemctl reload apache2     # recargar configuración sin cortar conexiones
systemctl restart apache2    # reiniciar el servicio
```

Estructura de `/etc/apache2/`

RUTA	PARA QUÉ SIRVE
`apache2.conf`	Archivo principal de configuración global
`ports.conf`	Define los puertos en los que escucha el servidor
`sites-available/`	Configuraciones de virtual hosts **disponibles**
`sites-enabled/`	Virtual hosts **activos** (enlaces simbólicos)
`mods-available/`	Módulos disponibles para activar
`mods-enabled/`	Módulos **cargados** (enlaces simbólicos)
`conf-available/`	Fragmentos de configuración disponibles
`conf-enabled/`	Fragmentos **activos** (enlaces simbólicos)

ℹ️ El patrón `*-available` + `*-enabled` permite mantener configuraciones **preparadas pero apagadas** y activarlas con un comando.

Comandos de gestión

SITIOS Y CONFIGURACIONES

```
a2ensite  ejemplo.conf   # activar virtual host
a2dissite ejemplo.conf   # desactivar virtual host
a2enconf  fragmento      # activar fragmento
a2disconf fragmento      # desactivar fragmento
```

MÓDULOS

```
a2enmod  rewrite         # activar módulo
a2dismod rewrite         # desactivar módulo
```

Tras cualquier cambio:

```
systemctl reload apache2
```

Virtual Hosts en Apache

Cada sitio se define en un archivo de `sites-available/`:

```
<VirtualHost *:80>
    ServerAdmin webmaster@ejemplo.com
    ServerName  ejemplo.com
    ServerAlias www.ejemplo.com
    DocumentRoot /var/www/ejemplo

    ErrorLog  ${APACHE_LOG_DIR}/ejemplo_error.log
    CustomLog ${APACHE_LOG_DIR}/ejemplo_access.log combined
</VirtualHost>
```

ACTIVACIÓN

```
sudo a2ensite ejemplo.conf
sudo systemctl reload apache2
```

Directivas en bloques `<Directory>`

Permiten configurar **opciones por directorio** del sistema de ficheros.

```
<Directory /var/www/ejemplo>
    Options Indexes FollowSymLinks
    AllowOverride None
    Require all granted
</Directory>
```

- **`Options`** — comportamiento (`Indexes` para listar, `FollowSymLinks` para enlaces…)
- **`AllowOverride`** — si se permiten archivos `.htaccess`
- **`Require`** — quién puede acceder (`all granted`, `ip 192.168.1`, `valid-user`…)

02

Funcionalidades en Apache

Alias, autenticación y redirecciones

Alias y redirecciones

ALIAS

Asocia una URL a una **ruta del sistema**:

```
Alias "/imagenes" "/srv/recursos/img"

<Directory "/srv/recursos/img">
    Require all granted
</Directory>
```

REDIRECT

Redirige peticiones a otra URL:

```
Redirect "/old" "/new"

Redirect permanent "/uno" \
    "http://www.pagina2.com/dos"
```

`permanent` envía un **301**; sin él, un **302**.

Autenticación básica

Restringe el acceso a un directorio mediante usuario y contraseña.

```
<Directory "/var/www/miweb/privado">
    AuthType Basic
    AuthName "Zona restringida"
    AuthUserFile "/etc/apache2/claves/passwd.txt"
    Require valid-user
</Directory>
```

Autenticación básica

CREAR EL ARCHIVO DE CONTRASEÑAS

```
# Primer usuario: crea el archivo (-c)
sudo htpasswd -c /etc/apache2/claves/passwd.txt usuario1

# Usuarios siguientes: sin -c (no sobreescribe)
sudo htpasswd /etc/apache2/claves/passwd.txt usuario2
```

⚠️ La autenticación básica envía las credenciales en **Base64**: usar **siempre** sobre HTTPS.

Control de acceso

```
<Directory "/var/www/miweb/admin">
    # Por dirección IP
    Require ip 192.168.1
    Require ip 10.0.0.5

    # O por usuario autenticado
    # Require valid-user
</Directory>
```

COMBINACIONES HABITUALES

- `Require all granted` — acceso libre
- `Require all denied` — bloquear todo
- `Require ip <red>` — sólo desde una red
- `Require valid-user` — cualquier usuario autenticado

Módulos habituales en Apache

MÓDULO	FUNCIÓN
`mod_rewrite`	Reescritura de URLs y reglas avanzadas
`mod_proxy`	Proxy inverso y balanceo de carga
`mod_headers`	Manipulación de cabeceras HTTP
`mod_deflate`	Compresión de respuestas
`mod_expires`	Cabeceras de expiración para caché
`mod_ssl`	Soporte de HTTPS / TLS

```
sudo a2enmod rewrite
sudo systemctl reload apache2
```

Logs en Apache

Los registros se guardan por defecto en `/var/log/apache2/`.

ACCESS.LOG

- Registra **todas las peticiones** atendidas
- Incluye IP cliente, fecha, método, URL, código de estado y tamaño

ERROR.LOG

- Errores del servidor y de configuración
- Problemas de permisos y de módulos
- Mensajes de inicio y parada del servicio

```
sudo tail -f /var/log/apache2/error.log
sudo journalctl -u apache2
```

03

Nginx

Instalación y estructura de configuración

Instalación de Nginx

```
sudo apt update
sudo apt install nginx
```

TRAS LA INSTALACIÓN

- El servicio queda activado automáticamente (`nginx.service`)
- Escucha en los puertos **80** y **443**
- Sirve por defecto desde `/var/www/html/`

```
systemctl status nginx
systemctl reload nginx
systemctl restart nginx
nginx -t                  # comprobar la sintaxis antes de recargar
```

Estructura de `/etc/nginx/`

RUTA	PARA QUÉ SIRVE
`nginx.conf`	Archivo principal de configuración
`conf.d/`	Fragmentos de configuración cargados por defecto
`sites-available/`	Configuraciones de virtual hosts disponibles
`sites-enabled/`	Sitios **activos** (enlaces simbólicos)
`snippets/`	Fragmentos **reutilizables** (incluibles desde otros archivos)
`/var/www/`	Directorio raíz por defecto del contenido
`/var/log/nginx/`	Logs de acceso y errores

ℹ️ Nginx en Debian usa el mismo patrón `sites-available/` + `sites-enabled/` que Apache, pero **no** dispone de comandos tipo `a2ensite`: la activación se hace creando el enlace simbólico a mano.

Server Blocks (Virtual Hosts)

```
server {
    listen 80;
    server_name ejemplo.com www.ejemplo.com;

    root  /var/www/ejemplo;
    index index.html index.htm;

    access_log /var/log/nginx/ejemplo_access.log;
    error_log  /var/log/nginx/ejemplo_error.log;

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

Server Blocks (Virtual Hosts)

ACTIVACIÓN

```
sudo ln -s /etc/nginx/sites-available/ejemplo \
           /etc/nginx/sites-enabled/
sudo nginx -t && sudo systemctl reload nginx
```

Directivas principales

IDENTIFICACIÓN DEL SITIO

- `listen` — puerto y protocolo
- `server_name` — nombres atendidos por el bloque
- `root` — directorio raíz del contenido
- `index` — archivos por defecto

RESOLUCIÓN DE PETICIONES

- `location` — mapea URLs a recursos
- `try_files` — orden de búsqueda de archivos
- `alias` — ruta alternativa para una URL
- `proxy_pass` — reenvío al backend

Servidor por defecto

Cuando una petición llega con un `Host` que no coincide con ningún `server_name`, Nginx la atiende con el bloque marcado como `default_server`:

```
server {
    listen 80 default_server;
    server_name _;
    root /var/www/html;
}
```

ℹ️ El nombre `_` es un convenio para indicar "cualquier nombre"; lo que de verdad importa es la opción `default_server` en `listen`.

04

Funcionalidades en Nginx

location, autenticación, redirecciones y snippets

Bloques `location`

Mapean **URLs** a recursos del sistema. Pueden anidarse y combinar con otras directivas:

```
location /imagenes/ {
    alias /srv/recursos/imagenes/;
    autoindex on;
}
```

- **`alias`** sustituye la parte coincidente de la URL por la ruta indicada
- **`autoindex on`** muestra el listado del directorio si no hay `index`

ℹ️ Nginx **sigue enlaces simbólicos** por defecto siempre que el usuario del proceso tenga permisos sobre el destino.

Autenticación básica

```
location /privado/ {
    auth_basic "Zona restringida";
    auth_basic_user_file /etc/nginx/.htpasswd;
}
```

Crear el archivo de contraseñas (requiere el paquete `apache2-utils`):

```
sudo apt install apache2-utils
sudo htpasswd -c /etc/nginx/.htpasswd usuario1
sudo htpasswd    /etc/nginx/.htpasswd usuario2
```

⚠️ El formato del archivo es el mismo que en Apache: una herramienta sirve para los dos servidores.

Control de acceso por IP

```
location /admin/ {
    allow 192.168.1.0/24;
    allow 10.0.0.5;
    deny  all;
}
```

Control de acceso por IP

COMBINAR IP Y AUTENTICACIÓN

```
location /panel/ {
    satisfy any;          # basta con cumplir UNA de las dos

    allow 192.168.1.0/24;
    deny  all;

    auth_basic           "Acceso restringido";
    auth_basic_user_file /etc/nginx/.htpasswd;
}
```

`satisfy all` (por defecto) exigiría cumplir ambas; `satisfy any` permite el acceso si se cumple **cualquiera**.

Redirecciones

RETURN

```
# Permanente (301)
return 301 http://nuevo.ejemplo.com$request_uri;

# Temporal (302)
return 302 /aviso.html;
```

Es la forma **recomendada** y más eficiente de redirigir.

REWRITE

```
rewrite ^/blog/(.*)$ /articulos/$1 permanent;
```

Permite **reescribir** URLs aplicando expresiones regulares. `permanent` genera un **301**.

Snippets reutilizables

Fragmentos guardados en `/etc/nginx/snippets/` que pueden incluirse desde varios `server`:

```
# /etc/nginx/snippets/seguridad.conf
add_header X-Frame-Options       "SAMEORIGIN";
add_header X-Content-Type-Options "nosniff";
```

ℹ️ Idea equivalente al patrón `conf-available/` de Apache: configuración modular que se reutiliza en varios sitios.

```
server {
    listen 80;
    server_name ejemplo.com;
    include snippets/seguridad.conf;
    ...
}
```

Módulos habituales en Nginx

MÓDULO	FUNCIÓN
`ngx_http_rewrite_module`	Reescritura de URLs y redirecciones
`ngx_http_proxy_module`	Proxy inverso y balanceo de carga
`ngx_http_headers_module`	Manipulación de cabeceras HTTP
`ngx_http_ssl_module`	Soporte HTTPS / TLS
`ngx_http_auth_basic_module`	Autenticación básica

ℹ️ A diferencia de Apache, los módulos en Nginx se eligen al **compilar**: en Debian vienen los más habituales ya incluidos en el paquete `nginx`.

Logs en Nginx

Los registros se guardan por defecto en `/var/log/nginx/`.

ACCESS.LOG

- Todas las peticiones atendidas
- IP, método, URL, código, tamaño y tiempo de respuesta

ERROR.LOG

- Errores en la ejecución del servidor
- Errores de configuración
- Problemas con los backends (PHP-FPM, proxy_pass...)

```
sudo tail -f /var/log/nginx/access.log
sudo tail -f /var/log/nginx/error.log
sudo journalctl -u nginx
```

Apache vs Nginx — resumen

APACHE

- Modelo basado en **procesos / hilos**
- Configuración por directorio con `.htaccess`
- Gran catálogo de **módulos cargables**
- Comandos `a2ensite` / `a2enmod`
- Ideal con **PHP** y configuraciones heterogéneas

NGINX

- Modelo **asíncrono** orientado a eventos
- Configuración **centralizada**, sin `.htaccess`
- Mejor rendimiento sirviendo **estáticos**
- Activación de sitios **manual** con enlaces
- Excelente como **proxy inverso** y balanceador

Para profundizar

- Documentación oficial de Apache
- Documentación oficial de Nginx

SRI · UNIDAD 3 — SERVIDORES WEB: APACHE Y NGINX

¡Gracias!

Apache y Nginx → Aplicaciones web dinámicas

📧 José Domingo Muñoz 🏫 IES Gonzalo Nazareno · Dos Hermanas 📚 SRI