UNL Virtual

Facultad de Ingeniería y Ciencias Hídricas

Tecnicatura Universitaria en Software Libre

Software Libre: Tecnologías Web

Profesor: Juan Pablo Taulamet

Trabajo Integrador Final

Alumno: Emanuel Conte Garcia

Año 2025

TP Integrador Final-TW – UNL: Trabajo Practico Final – Tecnologías Web Guia de Ngnix by Emanuel conte

is marked CC0 1.0

I. Introducción

¿Qué es Nginx?
Un servidor web open-source de alto rendimiento, también usado como proxy inverso, balanceador de carga y caché HTTP.

Su diseño está orientado a eventos y le permite manejar miles de conexiones simultáneas con bajo consumo de recursos.

Casos de uso clave:

1. *Hosting de sitios web* estáticos/dinámicos (ej: WordPress).
2. *Acelerar aplicaciones* mediante caché y compresión.
3. *Proteger servidores backend* como firewall de aplicación.
4. *Distribuir tráfico entre servidores* (balanceo de carga).

II. Características Técnicas (Core Features)

Las caracteristicas del server Ngnix son**:**

1. **Alto rendimiento**:
   Usa arquitectura *asíncrona y no bloqueante* (no genera procesos por cada conexión).
   Es ideal para sitios con alto tráfico (ej: Netflix, Dropbox).

2. **Proxy inverso**:
   Recibe peticiones de clientes y las redirige a servidores backend (ej: aplicaciones Python, Node.js), *ocultando la infraestructura* real *evitando ataques DdoS.*

3. **Balanceo de carga**:
   Distribuye tráfico entre múltiples servidores usando algoritmos: Round Robin, Least Connections, IP Hash. Estos algoritmos, diriguen el tráfico de los datos.

4. **Caché HTTP**:
   Almacena respuestas de servidores backend, de esta forma *reduce latencia y carga* del servidor.

5. **SSL/TLS**:
   Soporte para HTTPS moderno (TLS 1.3), terminación SSL y certificados automáticos utilizando Let's Encrypt, que es una autoridad de *certificación gratuita, automatizada y abierta*.

6. **Compresión**:
   Reduce el tamaño de las respuestas comprimiendo utilizando el formato de compresión Gzip y el mas nuevo y de código abierto formato Brotli *para acelerar carga*.

7. **Rewrites y redirecciones**:
   Modifica URLs dinámicamente usando expresiones regulares.

III. Módulos y Extensibilidad

Nginx tiena una *estrucutra extensible y modular*.

Sus módulos dinámicos (plugins) permiten la personalización del servidor.

Poseé los siguientes modulos:

Módulos oficiales

1. **ngx_http_ssl_module**: *Soporte HTTPS*, para transferencias seguras.

2. **ngx_http_v2_module**: *Soporte para HTTP/2,* la principal diferencia es que unas procesan cada petición de forma individual mientrass que la V2 utiliza multiplexacion para procesar diferentes solicitudes.

3. **ngx_stream_module**: *Soporte Proxy para TCP/UDP* (ej: balanceo de carga entre servidores para bases de datos).

Módulos de terceros:

1. **ModSecurity:**Soporte para el *Firewall de aplicación* (WAF) de código abierto.

2. **Lua Nginx Module:***Ejecutar scripts personalizados* (ej: autenticación avanzada).

IV. Escenarios de Uso Avanzados

En esta sección explicaremos algunos ejemplos de uso avanzado.

Microservicios:

Enrutaremiento de servicios según la URL:

Ejemplo de tráfico en Backend:

```
location /api/ {
    proxy_pass http://servidor-api;  # Tráfico a backend
}
```

(location /api/ { proxy_pass http://servidor-api; }**)**

Las peticiones a “tu dominio.com/api/” se redirigen al servidor de API.

Ejemplo Ejemplo de tráfico en Frontend:

```
location /app/ {
    proxy_pass http://frontend;      # Tráfico a frontend
}
```

(location /app/ { proxy_pass http://frontend; }**)**

Las peticiones a “tudominio.com/app/ van al frontend.

Como ventaja, esto permite un solo punto de entrada para multiples servicios.

Seguridad:

Ejemplo 1:

```
limit_req_zone $binary_remote_addr zone=one:10m rate=10r/s;
```

$binary_remote_addr: Identifica por dirección IP.
zone=one:10m: Zona de memoria de 10MB para almacenar estados.
rate=10r/s: Límite de 10 peticiones por segundo por IP.

Esta configuración *limita la tasa de peticiones* contra DdoS.

Ejemplo 2:

```
deny 192.168.1.1;          # Bloquea IP especifica
deny 192.168.1.0/24;       # Bloquea rango de IPs
allow all;                 # Permite el resto
```

Deny, *bloquea IP maliciosas*.

Allow all, permite el resto de las IP’s.

Optimización de medios:

Ejemplo: Con el módulo ngx_http_image_filter_module:

```
location /images/ {
    image_filter resize 300 200;  # Redimensiona a 300x200px
    image_filter_buffer 10M;      # Tamaño máximo de imagen
}
```

Permite redimensionar “on-the-fly”, reducir el peso de las imagenes y transformarlas.

A/B Testing:

Ejemplo: Balance de tráfico.

```
split_clients "${remote_addr}${date_gmt}" $variant {
    50%   http://version-a;  # 50% del tráfico a versión A
    *     http://version-b;  # Resto a versión B
}
```

La función en este caso, según la IP+fecha, divide el tráfico concientemente entre la Versión A y B, es decir, (mismo usuario siempre a ve la misma versión.)

Permite realizar test de conversión y usabilidad.

V. Versiones Disponibles

Nginx Open Source (libre):

Versión base con todas las funcionalidades esenciales.

Nginx Plus (comercial):

Incluye soporte 24/7, panel de monitoreo, actualizaciones automáticas y features avanzadas:

- Balanceo de carga con health checks.
- Dashboard en tiempo real.
- Autenticación JWT.

VI. Comunidad y Soporte

- **Documentación oficial**:
  Guías detalladas en nginx.org/en/docs/ .
- **Foros y Q&A**:
  Comunidad activa en Stack Overflow y nginx forum .
- **Soporte comercial**:
  Para empresas (Nginx Plus) con SLA garantizado.

VII. Comparativa con Otros Servidores

Característica	Nginx	Apache
Modelo de conexiones	Asíncrono	Por proceso
Consumo de memoria	Muy bajo	Moderado
Configuración	Declarativa	.htaccess
Proxy inverso	Nativo	Módulo mod_proxy

VIII. Instalación de Ngnix en Debian

PASO-1:

Necesitamos los siguientes paquetes en caso de no tenerlos instalados:

(curl: Para descargar archivos desde internet;

```
sudo apt install curl gnupg2 ca-certificates lsb-release debian-archive-keyring
```

gnupg2: Para manejar claves GPG de verificación;

ca-certificates: Certificados SSL para conexiones seguras;

lsb-release: Para identificar la versión de Debian;

debian-archive-keyring: Claves de verificación de paquetes Debian).

PASO-2:

Debemos importar la firma oficial de Ngnix.

```
curl https://nginx.org/keys/nginx_signing.key | gpg --dearmor \
    | sudo tee /usr/share/keyrings/nginx-archive-keyring.gpg >/dev/null
```

Descarga la clave de firma de Ngnix.

PASO-3:

Verificar la clave.

```
gpg --dry-run --quiet --no-keyring --import --import-options import-show /usr/share/keyrings/ngi
nx-archive-keyring.gpg
```

Verifica que la huella digital sea:

573BFD6B3D8FBC641079A6ABABF5BD827BD9BF62

PASO-4:

Configuración del repositorio de Ngnix

En este paso podremos optar por la versión “Estable” o la “Mainline”.(La mainline es mas nueva pero menos estable.)

Versión Estable:

```
echo "deb [signed-by=/usr/share/keyrings/nginx-archive-keyring.gpg] \
http://nginx.org/packages/debian `lsb_release -cs` nginx" \
    | sudo tee /etc/apt/sources.list.d/nginx.list
```

Versión Mainline:

```
echo "deb [signed-by=/usr/share/keyrings/nginx-archive-keyring.gpg] \
http://nginx.org/packages/mainline/debian `lsb_release -cs` nginx" \
    | sudo tee /etc/apt/sources.list.d/nginx.list
```

(**lsb_release -cs** detecta automáticamente tu versión de Debian (ej: "bullseye").

La opción **signed-by** especifica qué clave usar para verificar los paquetes.)

PASO-5:

Configurar prioridad del repositorio

Establece que los paquetes de nginx.org tengan prioridad sobre los de los repositorios de Debian.

```
echo -e "Package: *\nPin: origin nginx.org\nPin: release o=nginx\nPin-Priority: 900\n" \
    | sudo tee /etc/apt/preferences.d/99nginx
```

(Prioridad 900 significa que se preferirán estas versiones.**)**

PASO-6:

Instalar Ngnix

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

(apt update: Actualiza la lista de paquetes disponibles.

apt install nginx: Instala Nginx desde el repositorio oficial.)

PASO-7:

Verificación de la instalación

```
sudo systemctl status nginx
nginx -v
```

(sudo systemctl status nginx Muestra el estado del server.

nginx -v Muestra la versión instalada.**)**

IX. Hosteando mi sitio Web en Ngnix

PASO-1:

Estructura de directorios

```
sudo mkdir -p /var/www/tu-dominio/html
sudo chown -R $USER:$USER /var/www/tu-dominio/html
sudo chmod -R 755 /var/www/tu-dominio
```

En el directorio /var/www/ colocamos nuestro dominio, y dentro nuestro sitio.

Asignamos los permisos.

PASO-2:

Creamos un archivo HTML de prueba

```
nano /var/www/tu-dominio/html/index.html
```

Podemos escribir dentro head y un body de prueba para testear nuestro server.

PASO-3:

Crear archivo de configuración del sitio

```
server {
    listen 80;
    listen [::]:80;

    server_name tu-dominio.com www.tu-dominio.com;

    root /var/www/tu-dominio/html;
    index index.html index.htm;

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

Configuramos el puerto de escucha con **listen**.

El nombre del dominio, o varios separados por comas con **server_name**.

La ubicacón del repositorio con **root**.

La ubicación de archivos que se serviran por defecto en **index**.

PASO-4:

Habilitar el sitio

```
# Crear enlace simbólico
sudo ln -s /etc/nginx/sites-available/tu-dominio /etc/nginx/sites-enabled/

# Verificar configuración
sudo nginx -t

# Recargar Nginx
sudo systemctl reload nginx
```

Por una cuesion de orden, creamos el enlace simbolico sites, para tener nuestros sitios habilitados en **sites-enable**.

Verificamos la configuración del server con **sudo nginx -t**.

Y finalmente recargamos la configuración con **sudo systemctl reload nginx**.

X. Fuentes consultadas

Documentación oficial.

https://nginx.org/en/docs/index.html

Otros sitios consultados:

Guía paso a paso de instalacion de ngnix para Debian11, (friendhosting.net)

https://friendhosting.net/es/blog/install-nginx-on-debian-11.php

¿Qué es Ngnix y como funciona (Hostinet)?,

https://www.hostinet.com/formacion/vps-servidores/nginx-que-es-y-como-funciona/

Nginx: características y comparación con Apache (Pingback)

https://pingback.com/es/resources/nginx/

Características técnicas y arquitectura de Nginx (Hackio)

https://www.hackio.com/blog/que-es-nginx

XI. Conclusión

Nginx es cómo una navaja suiza para infraestructuras web: **rápido, seguro y escalable**.

Ideal desde proyectos pequeños hasta arquitecturas de gran escala (CDNs, aplicaciones en la nube).

Su flexibilidad permite usarlo como:

- Servidor web principal.
- Capa de seguridad frente a servidores de aplicación.
- "Traffic manager" en entornos cloud.