

Referencia Rápida de Nginx

Bloques de servidor, proxy, SSL, balanceo de carga, logging

Instalación

Instalar por SO

Ubuntu / Debian	<code>sudo apt install nginx</code>
RHEL / CentOS	<code>sudo dnf install nginx</code>
macOS	<code>brew install nginx</code>
Alpine	<code>apk add nginx</code>
Docker	<code>docker run -p 80:80 nginx</code>

Gestión del servicio

<code>sudo systemctl start nginx</code>	Iniciar Nginx
<code>sudo systemctl stop nginx</code>	Detener Nginx
<code>sudo systemctl reload nginx</code>	Recargar configuración (sin tiempo de inactividad)
<code>sudo systemctl enable nginx</code>	Habilitar al inicio
<code>nginx -t</code>	Verificar sintaxis de configuración
<code>nginx -T</code>	Verificar y mostrar configuración completa
<code>nginx -s reload</code>	Señalar al proceso en ejecución que recargue

Configuración básica

Ubicaciones de archivos

<code>/etc/nginx/nginx.conf</code>	Archivo de configuración principal
<code>/etc/nginx/conf.d/</code>	Configuraciones de sitios adicionales (*.conf)
<code>/etc/nginx/sites-available/</code>	Configuraciones de sitios disponibles (Debian)
<code>/etc/nginx/sites-enabled/</code>	Symlinks a configuraciones activas
<code>/var/log/nginx/</code>	Logs de acceso y de error
<code>/var/www/html/</code>	Raíz de documentos por defecto

Configuración mínima

```
server {
    listen 80;
    server_name example.com;
    root /var/www/mysite;
    index index.html;
}
```

Estructura de configuración

<code>http { }</code>	Configuración del servidor HTTP (nivel superior)
<code>server { }</code>	Definición de host virtual
<code>location { }</code>	Bloque de coincidencia de URI
<code>upstream { }</code>	Grupo de servidores backend
<code>events { }</code>	Configuración de manejo de conexiones

Bloques de servidor

Hosts virtuales basados en nombre

```
server {
    listen 80;
    server_name site-a.com;
    root /var/www/site-a;
}
server {
    listen 80;
    server_name site-b.com;
    root /var/www/site-b;
}
```

Por defecto y captura total

```
server {
    listen 80 default_server;
    server_name _;
    return 444; # drop connection
}
```

Redirección a HTTPS

```
server {
    listen 80;
    server_name example.com;
    return 301 https://$host$request_uri;
}
```

Bloques de location

Prioridad de coincidencia (mayor a menor)

<code>= /path</code>	Coincidencia exacta (mayor prioridad)
<code>^~ /path</code>	Coincidencia de prefijo, omitir regex
<code>~ regex</code>	Regex sensible a mayúsculas
<code>~* regex</code>	Regex insensible a mayúsculas
<code>/path</code>	Coincidencia de prefijo (menor prioridad)

Ejemplos de location

```
location = / {
    # exact root only
}
location /api/ {
    proxy_pass http://backend;
}
location ~* \.(jpg|png|gif)$ {
    expires 30d;
}
```

try_files

```
location / {
    try_files $uri $uri /index.html;
}
```

Intenta archivo, luego directorio, luego alternativa — esencial para SPAs

Proxy inverso

Proxy básico

```
location /api/ {
    proxy_pass http://localhost:3000/;
    proxy_set_header Host $host;
    proxy_set_header X-Real-IP $remote_addr;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_set_header X-Forwarded-Proto $scheme;
}
```

Proxy WebSocket

```
location /ws/ {
    proxy_pass http://localhost:3000;
    proxy_http_version 1.1;
    proxy_set_header Upgrade $http_upgrade;
    proxy_set_header Connection "upgrade";
}
```

Directivas de proxy

<code>proxy_pass</code>	URL del backend
<code>proxy_set_header</code>	Pasar cabeceras personalizadas al backend
<code>proxy_read_timeout</code>	Tiempo de espera para respuesta del backend (60s por defecto)
<code>proxy_buffering off</code>	Desactivar buffering de respuesta
<code>proxy_redirect</code>	Reescribir cabeceras Location del backend

SSL / TLS

Servidor HTTPS

```
server {
    listen 443 ssl;
    server_name example.com;

    ssl_certificate /etc/ssl/certs/example.crt;
    ssl_certificate_key /etc/ssl/private/example.key;
    ssl_protocols TLSv1.2 TLSv1.3;
    ssl_ciphers HIGH:!aNULL:!MD5;
}
```

Let's Encrypt con Certbot

```
sudo certbot --nginx -d example.com
sudo certbot renew --dry-run
```

Buenas prácticas SSL

<code>ssl_protocols TLSv1.2 TLSv1.3</code>	Deshabilitar versiones antiguas de TLS
<code>ssl_prefer_server_ciphers on</code>	El servidor elige el cifrado
<code>ssl_session_cache shared:SSL:10m</code>	Reutilización de sesión para rendimiento
<code>add_header Strict-Transport-Security ssl_stapling on</code>	Cabecera HSTS Grapado OCSP para handshake más rápido

Balanceo de carga

Bloque upstream

```
upstream backend {
    server 10.0.0.1:3000;
    server 10.0.0.2:3000;
    server 10.0.0.3:3000;
}
server {
    location / {
        proxy_pass http://backend;
    }
}
```

Métodos de balanceo de carga

<code>(default)</code>	Round-robin
<code>least_conn</code>	Menos conexiones activas
<code>ip_hash</code>	Sesiones pegajosas por IP del cliente
<code>hash \$request_uri</code>	Hash consistente por URI

Opciones de servidor

<code>weight=3</code>	Enviar 3 veces más tráfico
<code>max_fails=3</code>	Fallos antes de marcar como inactivo
<code>fail_timeout=30s</code>	Tiempo para marcar servidor como inactivo
<code>backup</code>	Usar solo cuando los demás estén caídos
<code>down</code>	Marcar servidor como permanentemente fuera de línea

Archivos estáticos y caché

Servir archivos estáticos

```
location /static/ {
    alias /var/www/assets/;
    expires 30d;
    add_header Cache-Control "public, immutable";
}
```

Referencia Rápida de Nginx

Compresión Gzip

```
gzip on;
gzip_types text/plain text/css
      application/json application/javascript;
gzip_min_length 1000;
gzip_comp_level 5;
```

Directivas de caché

expires 30d	Establecer Expires y Cache-Control max-age
expires off	Desactivar la cabecera expires
etag on	Habilitar cabecera ETag (por defecto)
sendfile on	Servicio eficiente de archivos via kernel
tcp_nopush on	Optimizar el envío de paquetes

Logging

Configuración de logs

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

# Custom log format
log_format main '$remote_addr - $status '
               '$request' $body_bytes_sent';
access_log /var/log/nginx/access.log main;
```

Niveles de log de error

debug	Detallado (requiere --with-debug)
info	Informativo
notice	Normal pero notable
warn	Advertencias
error	Errores (por defecto)
crit	Problemas críticos

Logging condicional

```
map $status $loggable {
    ~^[23] 0;
    default 1;
}
access_log /var/log/nginx/access.log combined if=$loggable;
```

Omitir logging de respuestas 2xx/3xx para reducir el volumen de logs

Seguridad

Limitación de velocidad

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

location /api/ {
    limit_req zone=api burst=20 nodelay;
}
```

Control de acceso

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

Cabeceras de seguridad

X-Frame-Options DENY	Prevenir clickjacking
X-Content-Type-Options nosniff	Prevenir MIME sniffing
X-XSS-Protection "1; mode=block"	Filtro XSS (navegadores legados)
Content-Security-Policy	Controlar fuentes de carga de recursos
Referrer-Policy no-referrer	Controlar información de referencia

Patrones comunes

SPA (aplicación de página única)

```
location / {
    root /var/www/app;
    try_files $uri $uri/ /index.html;
}
```

Cabeceras CORS

```
location /api/ {
    add_header Access-Control-Allow-Origin *;
    add_header Access-Control-Allow-Methods
               "GET, POST, PUT, DELETE, OPTIONS";
    if ($request_method = OPTIONS) {
        return 204;
    }
    proxy_pass http://backend;
}
```

Variables útiles

\$host	Cabecera Host de la petición
\$uri	URI actual (normalizada)
\$request_uri	URI original con cadena de consulta
\$remote_addr	Dirección IP del cliente
\$scheme	http o https
\$args	Parámetros de cadena de consulta
\$status	Código de estado de la respuesta