

Référence rapide Nginx

Blocs serveur, proxy, SSL, équilibrage de charge, journalisation

Installation

Installer selon le système

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>

Gestion du service

<code>sudo systemctl start nginx</code>	Démarrer Nginx
<code>sudo systemctl stop nginx</code>	Arrêter Nginx
<code>sudo systemctl reload nginx</code>	Recharger la config (sans interruption)
<code>sudo systemctl enable nginx</code>	Activer au démarrage
<code>nginx -t</code>	Tester la syntaxe de la configuration
<code>nginx -T</code>	Tester et afficher la config complète
<code>nginx -s reload</code>	Envoyer un signal de rechargement au processus

Configuration de base

Emplacements des fichiers

<code>/etc/nginx/nginx.conf</code>	Fichier de configuration principal
<code>/etc/nginx/conf.d/</code>	Configurations de sites (*.conf)
<code>/etc/nginx/sites-available/</code>	Configurations disponibles (Debian)
<code>/etc/nginx/sites-enabled/</code>	Liens symboliques vers les configs actives
<code>/var/log/nginx/</code>	Journaux d'accès et d'erreurs
<code>/var/www/html/</code>	Racine de document par défaut

Configuration minimale

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

Structure de la configuration

<code>http { }</code>	Paramètres du serveur HTTP (niveau supérieur)
<code>server { }</code>	Définition d'hôte virtuel
<code>location { }</code>	Bloc de correspondance d'URI
<code>upstream { }</code>	Groupe de serveurs backend
<code>events { }</code>	Paramètres de gestion des connexions

Blocs serveur

Hôtes virtuels basés sur le nom

```
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;
}
```

Serveur par défaut et attrape-tout

```
server {
    listen 80 default_server;
    server_name _;
    return 444; # fermer la connexion
}
```

Redirection HTTPS

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

Blocs location

Priorité des correspondances (haute à basse)

<code>= /path</code>	Correspondance exacte (priorité maximale)
<code>^~ /path</code>	Préfixe, ignorer les regex
<code>~ regex</code>	Regex sensible à la casse
<code>~* regex</code>	Regex insensible à la casse
<code>/path</code>	Préfixe (priorité minimale)

Exemples de location

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

try_files

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

Tester le fichier, puis le dossier, puis le fallback — indispensable pour les SPA

Proxy inverse

Proxy de base

```
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";
}
```

Directives proxy

<code>proxy_pass</code>	URL du backend
<code>proxy_set_header</code>	Transmettre des en-têtes personnalisés au backend
<code>proxy_read_timeout</code>	Délai d'attente de la réponse backend (60s par défaut)
<code>proxy_buffering off</code>	Désactiver la mise en tampon de la réponse
<code>proxy_redirect</code>	Réécrire les en-têtes Location du backend

SSL / TLS

Serveur 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 avec Certbot

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

Bonnes pratiques SSL

<code>ssl_protocols TLSv1.2 TLSv1.3</code>	Désactiver les anciennes versions TLS
<code>ssl_prefer_server_ciphers on</code>	Le serveur choisit le chiffrement
<code>ssl_session_cache shared:SSL:10m</code>	Réutilisation des sessions pour les performances
<code>add_header Strict-Transport-Security</code>	En-tête HSTS
<code>ssl_stapling on</code>	Agrafage OCSP pour un handshake plus rapide

Équilibrage de charge

Bloc 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;
    }
}
```

Méthodes d'équilibrage

<code>(par défaut)</code>	Round-robin
<code>least_conn</code>	Moins de connexions actives
<code>ip_hash</code>	Sessions persistantes par IP client
<code>hash \$request_uri</code>	Hachage cohérent par URI

Options des serveurs

<code>weight=3</code>	Envoyer 3× plus de trafic
<code>max_fails=3</code>	Échecs avant de marquer hors service
<code>fail_timeout=30s</code>	Durée pour marquer le serveur hors service
<code>backup</code>	Utiliser seulement si les autres sont indisponibles
<code>down</code>	Marquer le serveur définitivement hors ligne

Fichiers statiques et cache

Servir des fichiers statiques

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

Référence rapide Nginx

Compression Gzip

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

Directives de cache

expires 30d	Définir Expires et Cache-Control max-age
expires off	Désactiver l'en-tête expires
etag on	Activer l'en-tête ETag (défaut)
sendfile on	Service de fichiers efficace via le noyau
tcp_nopush on	Optimiser l'envoi des paquets

Journalisation

Configuration des journaux

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

```
# Format de journal personnalisé
log_format main '$remote_addr - $status '
                '"$request" $body_bytes_sent';
access_log /var/log/nginx/access.log main;
```

Niveaux du journal d'erreurs

debug	Verbeux (nécessite --with-debug)
info	Informationnel
notice	Normal mais notable
warn	Avertissements
error	Erreurs (défaut)
crit	Problèmes critiques

Journalisation conditionnelle

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

Ignorer les réponses 2xx/3xx pour réduire le volume des journaux

Sécurité

Limitation de débit

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

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

Contrôle d'accès

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

En-têtes de sécurité

X-Frame-Options DENY	Prévenir le clickjacking
X-Content-Type-Options nosniff	Prévenir le sniffing MIME
X-XSS-Protection "1; mode=block"	Filtre XSS (navigateurs anciens)
Content-Security-Policy	Contrôler les sources de chargement des ressources
Referrer-Policy no-referrer	Contrôler les informations de référent

Motifs courants

SPA (application monopage)

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

En-têtes 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 utiles

\$host	En-tête Host de la requête
\$uri	URI courante (normalisée)
\$request_uri	URI originale avec chaîne de requête
\$remote_addr	Adresse IP du client
\$scheme	http ou https
\$args	Paramètres de la chaîne de requête
\$status	Code de statut de la réponse