NGINX 快速参考

Server 块、反向代理、SSL、负载均衡、日志

安装

按系统安装

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

服务管理

sudo systemctl start nginx	启动 Nginx
sudo systemctl stop nginx	停止 Nginx
sudo systemctl reload nginx	重载配置（不中断服务）
sudo systemctl enable nginx	设置开机自启
nginx -t	测试配置语法
nginx -T	测试并输出完整配置
nginx -s reload	向运行中进程发送重载信号

基本配置

文件位置

/etc/nginx/nginx.conf	主配置文件
/etc/nginx/conf.d/	站点配置插件目录（*.conf）
/etc/nginx/sites-available/	可用站点配置（Debian）
/etc/nginx/sites-enabled/	已启用站点配置的符号链接
/var/log/nginx/	访问日志和错误日志
/var/www/html/	默认文档根目录

最小配置

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

配置结构

http { }	HTTP 服务器设置（顶级）
server { }	虚拟主机定义
location { }	URI 匹配块
upstream { }	后端服务器组
events { }	连接处理设置

Server 块

基于名称的虚拟主机

```
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;
}
```

默认与兜底服务器

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

HTTPS 重定向

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

Location 块

匹配优先级（从高到低）

= /path	精确匹配（最高优先级）
^~ /path	前缀匹配，跳过正则
~ regex	区分大小写的正则
~* regex	不区分大小写的正则
/path	前缀匹配（最低优先级）

Location 示例

```
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;
}
```

依次尝试文件、目录、兜底路径——SPA 必备

反向代理

基本代理

```
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;
}
```

WebSocket 代理

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

代理指令

proxy_pass	后端 URL
proxy_set_header	向后端传递自定义请求头
proxy_read_timeout	后端响应超时（默认 60s）
proxy_buffering off	禁用响应缓冲
proxy_redirect	重写后端的 Location 响应头

SSL / TLS

HTTPS 服务器

```
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;
}
```

使用 Certbot 申请 Let's Encrypt

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

SSL 最佳实践

ssl_protocols TLSv1.2 TLSv1.3	禁用旧版 TLS
ssl_prefer_server_ciphers on	由服务器选择加密套件
ssl_session_cache shared:SSL:10m	会话复用以提升性能
add_header Strict-Transport-Security	HSTS 响应头
ssl_stapling on	OCSP Stapling 加速握手

负载均衡

Upstream 块

```
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;
    }
}
```

负载均衡算法

（默认）	轮询
least_conn	最少活跃连接
ip_hash	客户端 IP 会话粘性
hash $request_uri	按 URI 一致性哈希

服务器选项

weight=3	发送 3 倍流量
max_fails=3	标记为不可用前的失败次数
fail_timeout=30s	标记服务器为不可用的持续时间
backup	仅在其他服务器宕机时使用
down	永久标记服务器为离线

静态文件与缓存

提供静态文件

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

Gzip 压缩

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

缓存指令

expires 30d	设置 Expires 和 Cache-Control max-age
expires off	禁用 expires 响应头
etag on	启用 ETag 响应头（默认）
sendfile on	通过内核高效提供文件
tcp_nopush on	优化数据包发送

日志

日志配置

```
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;
```

错误日志级别

debug	详细（需要 --with-debug 编译）
info	信息性
notice	正常但值得注意
warn	警告
error	错误（默认）
crit	严重问题

条件日志

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

跳过 2xx/3xx 的日志记录以减少日志量

安全

限流

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

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

访问控制

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

安全响应头

X-Frame-Options DENY	防止点击劫持
X-Content-Type-Options nosniff	防止 MIME 类型嗅探
X-XSS-Protection "1; mode=block"	XSS 过滤器（旧版浏览器）
Content-Security-Policy	控制资源加载来源
Referrer-Policy no-referrer	控制引用信息

常见模式

SPA（单页应用）

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

CORS 响应头

```
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;
}
```

常用变量

$host	请求的 Host 头
$uri	当前 URI（规范化）
$request_uri	含查询字符串的原始 URI
$remote_addr	客户端 IP 地址
$scheme	http 或 https
$args	查询字符串参数
$status	响应状态码