

# Comandi nginx

Comandi e snippet di configurazione nginx collaudati – cercabili e copiabili con un clic.  
Testa sempre la configurazione prima del reload.

daniele.michelin //  
cheat sheet

## Servizio e configurazione

Controlla la sintassi della configurazione

```
sudo nginx -t
```

Mostra tutta la configurazione attiva, include compresi

```
sudo nginx -T
```

Ricarica la configurazione senza chiudere le connessioni

```
sudo nginx -s reload
```

Reload tramite systemd (modo standard su Debian/Ubuntu)

```
sudo systemctl reload nginx
```

! Riavvia del tutto il servizio (chiude le connessioni)

```
sudo systemctl restart nginx
```

Prima testa, ricarica solo se va a buon fine

```
sudo nginx -t && sudo systemctl reload nginx
```

## Log e analisi

Segui l'access log in tempo reale

```
sudo tail -f /var/log/nginx/access.log
```

Segui l'error log in tempo reale

```
sudo tail -f /var/log/nginx/error.log
```

Conta i codici di stato HTTP

```
sudo awk '{print $9}' /var/log/nginx/access.log | sort |  
uniq -c | sort -rn
```

Ultime richieste con risposta 5xx

```
sudo awk '$9 ~ /^5/' /var/log/nginx/access.log | tail
```

Top 10 IP client per richieste

```
sudo awk '{print $1}' /var/log/nginx/access.log | sort |  
uniq -c | sort -rn | head
```

! Svuota l'access log (il file resta)

```
sudo truncate -s 0 /var/log/nginx/access.log
```

## Server block e redirect

Imposta i nomi di dominio di un server block

```
server_name example.com www.example.com;
```

Ascolta sulla porta 443 con TLS

```
listen 443 ssl http2;
```

Reindirizza tutto da HTTP a HTTPS

```
return 301 https://$host$request_uri;
```

Reindirizza www al dominio principale

```
return 301 https://example.com$request_uri;
```

Serve file, directory o 404

```
try_files $uri $uri/ =404;
```

Alza il limite di upload (es. Nextcloud)

```
client_max_body_size 512M;
```

## Reverse proxy

Inoltra le richieste a un backend

```
proxy_pass http://127.0.0.1:3000;
```

Passa l'header Host originale al backend

```
proxy_set_header Host $host;
```

Passa la catena degli IP client al backend

```
proxy_set_header X-Forwarded-For  
$proxy_add_x_forwarded_for;
```

Alza il timeout per backend lenti (contro 504)

```
proxy_read_timeout 300s;
```

Inoltra le connessioni WebSocket

```
proxy_http_version 1.1; proxy_set_header Upgrade  
$http_upgrade; proxy_set_header Connection "upgrade";
```

Raggruppa due backend in un upstream

```
upstream app { server 10.0.0.11:8080; server  
10.0.0.12:8080; }
```

## Rate limiting, accessi e gzip

Definisce un limite di richieste per IP (nel blocco http)

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

Applica il limite, ammette brevi picchi

```
limit_req zone=perip burst=20 nodelay;
```

Consente una rete, blocca tutte le altre

```
allow 192.168.1.0/24; deny all;
```

! Crea il file password per Basic Auth (-c sovrascrive!)

```
sudo htpasswd -c /etc/nginx/.htpasswd admin
```

Attiva la compressione gzip

```
gzip on;
```

Comprime altri tipi MIME

```
gzip_types text/css application/javascript  
application/json image/svg+xml;
```

Imposta la cache del browser per i file statici

```
expires 30d;
```

## Debug

Testa un server block in locale con l'header Host

```
curl -sI -H 'Host: example.com' http://127.0.0.1
```

Segue la catena di redirect

```
curl -sIL http://example.com | grep -iE 'HTTP/|location'
```

Attiva il log di debug (solo brevemente, molto verboso)

```
error_log /var/log/nginx/error.log debug;
```

Controlla su quali porte ascolta nginx

```
sudo ss -tlnp | grep nginx
```

Chi occupa la porta 80? (con 'address already in use')

```
sudo lsof -iTCP:80 -sTCP:LISTEN
```

Controlla i permessi lungo il percorso (per 403)

```
sudo namei -l /var/www/example.com/index.html
```