nginx cheat sheet

Field-tested nginx commands and config snippets for running servers – searchable and copyable in one click. Always test the config first, then reload.

daniele.michelin // cheat sheet

Service & config

Check the configuration for syntax errors

```
sudo nginx -t
```

Dump the full active config including includes

```
sudo nginx -T
```

Reload the config without dropping connections

```
sudo nginx -s reload
```

Reload via systemd (standard on Debian/Ubuntu)

```
sudo systemctl reload nginx
```

! Fully restart the service (drops connections)

```
sudo systemctl restart nginx
```

Test first, reload only on success

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

Logs & analysis

Follow the access log live

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

Follow the error log live

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

Count HTTP status codes

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

Latest requests answered with 5xx

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

Top 10 client IPs by requests

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

! Truncate the access log (file stays)

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

Server blocks & redirects

Set the domain names of a server block

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

Listen on port 443 with TLS

```
listen 443 ssl http2;
```

Redirect all HTTP to HTTPS

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

Redirect www to the apex domain

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

Serve file, directory or 404

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

Raise the upload limit (e.g. Nextcloud)

```
client_max_body_size 512M;
```

Reverse proxy

Forward requests to a backend

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

Pass the original Host header to the backend

```
proxy_set_header Host $host;
```

Pass the client IP chain to the backend

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

Raise timeout for slow backends (against 504)

```
proxy_read_timeout 300s;
```

Proxy WebSocket connections

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

Group two backends as an upstream pool

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

Rate limiting, access & gzip

Define a per-IP request limit (in http block)

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

Apply the limit, allow short bursts

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

Allow one network, deny everyone else

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

! Create a Basic Auth password file (-c overwrites!)

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

Turn on gzip compression

```
gzip on;
```

Compress additional MIME types

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

Set a browser cache for static files

```
expires 30d;
```

Debugging

Test a server block locally via Host header

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

Follow the redirect chain

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

Enable debug logging (briefly, very verbose)

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

Check which ports nginx listens on

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

Who holds port 80? (on 'address already in use')

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

Check permissions along the path (for 403)

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