Nginx Cheatsheet

A quick reference guide for Nginx, covering configuration, common directives, and useful commands for managing your web server.

Basic Configuration

Configuration File Structure

Nginx configuration files are typically located in `/etc/nginx/`.

- `nginx.conf`: Main configuration file.
- `sites-available/`: Stores virtual host configuration files.
- `sites-enabled/`: Contains symlinks to enable virtual host configurations from `sites-available`.
- `conf.d/`: Additional configuration snippets.

The main configuration file (`nginx.conf`) usually includes the following blocks:

- `http`: Contains directives related to HTTP traffic.
- `server`: Defines virtual host configurations.
- `location`: Specifies how Nginx handles requests for specific URIs.

Essential Directives

`listen`	Specifies the port (and optionally address) that the server should listen on. **Example:** `listen 80;`
`server_name`	Defines the domain names that this server block should handle. **Example:** `server_name example.com www.example.com;`
`root`	Specifies the root directory for serving files. **Example:** `root /var/www/example.com;`
`index`	Defines the default files to serve if no specific file is requested. **Example:** `index index.html index.htm;`
`location`	Configures how Nginx handles requests for specific URIs. **Example:** `location / { ... }`
`error_log`	Specifies the file to log errors to and the log level. **Example:** `error_log /var/log/nginx/error.log error;`
`access_log`	Specifies the file to log access requests to. **Example:** `access_log /var/log/nginx/access.log combined;`

Example Server Block

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

    location / {
        try_files $uri $uri/ =404;
    }

    error_log /var/log/nginx/error.log error;
    access_log /var/log/nginx/access.log
combined;
}
```

Common Commands

Service Management

Start Nginx	`sudo systemctl start nginx`
Stop Nginx	`sudo systemctl stop nginx`
Restart Nginx	`sudo systemctl restart nginx`
Reload Nginx	`sudo systemctl reload nginx` (graceful restart)
Check Nginx status	`sudo systemctl status nginx`
Enable Nginx on boot	`sudo systemctl enable nginx`
Disable Nginx on boot	`sudo systemctl disable nginx`

Configuration Testing

Test Nginx configuration file for syntax errors before applying changes:

```
sudo nginx -t
```

If the configuration is valid, you'll see output like:

```
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
```

Reverse Proxy & Load Balancing

Basic Reverse Proxy

Nginx can act as a reverse proxy, forwarding client requests to backend servers.

```
location / {
    proxy_pass http://backend_server;
    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;
}
```

Explanation of directives:

- `proxy_pass`: Specifies the address of the backend server.
- `proxy_set_header`: Sets HTTP headers to be passed to the backend server. It's important to forward the original host, IP, and protocol information.

Load Balancing

Nginx can distribute traffic across multiple backend servers using load balancing.

```
upsream backend {
    server backend1.example.com;
    server backend2.example.com;
}

server {
    location / {
        proxy_pass http://backend;
    }
}
```

Different load balancing methods can be specified using the `upstream` block:

- `round-robin` (default): Distributes requests in a round-robin fashion.
- `ip_hash`: Distributes requests based on the client's IP address.
- `least_conn`: Sends requests to the server with the fewest active connections.

SSL/TLS Configuration

Basic SSL Configuration

To enable SSL/TLS, you need an SSL certificate and key. You can obtain these from a Certificate Authority (CA) or generate a self-signed certificate (for testing purposes only).

```
sudo openssl req -x509 -nodes -days 365 -
newkey rsa:2048 -keyout
/etc/ssl/private/nginx-selfsigned.key -out
/etc/ssl/certs/nginx-selfsigned.crt
```

Configure the server block to listen on port 443 and specify the certificate and key files.

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

    ssl_certificate /etc/ssl/certs/nginx-
selfsigned.crt;
    ssl_certificate_key
/etc/ssl/private/nginx-selfsigned.key;

    # ... other configuration ...
}
```

Redirect HTTP to HTTPS

To automatically redirect HTTP traffic to HTTPS, add a separate server block for port 80.

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

Optimizing SSL Configuration

Use strong ciphers and enable HTTP Strict Transport Security (HSTS) to improve security.

```
ssl_protocols TLSv1.2 TLSv1.3;
ssl_prefer_server_ciphers on;
ssl_ciphers
'EECDH+AESGCM:EDH+AESGCM:AES256+EECDH:AES256+E
DH';
ssl_session_cache shared:SSL:10m;
ssl_session_timeout 10m;
add_header Strict-Transport-Security "max-
age=31536000; includeSubDomains" always;
```