**NGINX**

**Introduction to NGINX**

Nginx is a robust and efficient web server that is widely used in modern web infrastructure. Unlike traditional web servers that use a multi-threaded architecture, Nginx employs a non-threaded, event-driven architecture. This design allows Nginx to handle many more connections simultaneously, making it highly suitable for high-traffic websites. Beyond serving static web pages, Nginx is versatile and can be configured to perform a variety of roles, including load balancing, HTTP caching, and acting as a reverse proxy. These capabilities make it a cornerstone in many complex web architectures.

NGINX (pronounced "engine-x") is a high-performance web server. Created by Igor Sysoev in 2004, NGINX was designed to solve the C10K problem (handling 10,000+ concurrent connections) and has since become one of the most popular web servers globally.

**Key Features**

* **High Performance**: Handles thousands of concurrent connections with minimal resource usage
* **Stability**: Designed to run continuously without downtime
* **Flexibility**: Functions as web server, reverse proxy, load balancer, and more
* **Asynchronous Event-Driven Architecture**: Non-blocking I/O operations
* **Modular Design**: Core functionality can be extended with modules

## How NGINX Works

NGINX operates on a **master-worker** architecture

* **Master Process**:
  + Reads configuration files.
  + Manages worker processes.
* **Worker Processes**:
  + Handle client requests.
  + Each worker can handle thousands of connections concurrently using non-blocking I/O.

**Forward Proxy vs. Reverse Proxy**

* **Forward Proxy**: In a traditional HTTP connection, a client sends a request directly to a server. However, with a forward proxy (such as a VPN), the client sends the request to the proxy, which then forwards it to the server. The server is unaware of the original client; it only interacts with the proxy. This setup is useful for privacy and bypassing geo-restrictions.
* **Reverse Proxy**: In contrast, a reverse proxy sits between the client and multiple servers. The client sends a request to the reverse proxy, which then decides which server should handle the request. The client remains unaware of which specific server processes the request. Nginx is a popular choice for a reverse proxy because of its efficiency and flexibility.

For example

* **/admin** requests could be routed to Server 1.
* **/settings** requests could be routed to Server 2.

Nginx can handle these types of routing efficiently, ensuring that the correct server processes each request based on predefined rules.

## Key Concepts

* **Web Server**: Just like other web servers, NGINX handles requests from browsers and serves them the web pages they ask for. But it does this with exceptional speed, especially when dealing with static content like images, videos, and plain HTML files.
* **Reverse Proxy**: NGINX can sit in front of your web servers, acting as a middleman between the outside world and your servers. It’s like a gatekeeper that decides which server should handle each request, helping to balance the load and protect your servers from direct exposure to the internet.
* **Load Balancer**: If your website is getting a lot of traffic, you don’t want one server to do all the work. NGINX can distribute the incoming traffic across multiple servers, ensuring no single server is overwhelmed.
* **Caching System**: Instead of generating the same web page over and over for every user, NGINX can store a copy of the page and serve it quickly to anyone who asks, saving time and server resources.

## File Structure and Locations

|  |  |
| --- | --- |
| **Purpose** | **Location** |
| Main config file | /etc/nginx/nginx.conf |
| Server block configs | /etc/nginx/sites-available/ |
| Enabled configs | /etc/nginx/sites-enabled/ |
| Access logs | /var/log/nginx/access.log |
| Error logs | /var/log/nginx/error.log |
| Default web root | /var/www/html/ |

## Key Commands

|  |  |
| --- | --- |
| **Action** | **Command** |
| Start NGINX | sudo systemctl start nginx |
| Stop NGINX | sudo systemctl stop nginx |
| Restart NGINX | sudo systemctl restart nginx |
| Reload config | sudo nginx -s reload |
| Test config | sudo nginx -t |
| Check status | sudo systemctl status nginx |

**For Linux :**

•Update your package index : sudo yum update

•Install NGINX : sudo yum install nginx

•Start NGINX : sudo systemctl start nginx

•Enable NGINX to start on boot : sudo systemctl enable nginx

**Verifying Installation**

* Check version :nginx -v
* Test configuration syntax :sudo nginx –t
* Start NGINX :sudo systemctl start nginx
* For systems with system: **:**sudo service nginx start

**Main Directories**

* /etc/nginx/: Main configuration directory
* /var/log/nginx/: Log files location
* /var/www/html/: Default document root
* /usr/share/nginx/html/: Alternative document root

**Key Configuration Files**

* /etc/nginx/nginx.conf: Main configuration file
* /etc/nginx/conf.d/\*.conf: Additional configuration files
* /etc/nginx/sites-available/: Available virtual host configurations
* /etc/nginx/sites-enabled/: Enabled virtual host configurations (symlinks)
* /etc/nginx/mime.types: MIME type mappings

**Default Log Files**

* /var/log/nginx/access.log: Records all requests
* /var/log/nginx/error.log: Records errors and diagnostic information

**Configuration**

Let’s configure it to serve a simple HTML page. By default, NGINX serves files from the /var/www/html directory on Linux

* Create a new HTML file : echo "<h1>Welcome to NGINX!</h1>" | sudo tee /var/www/html/index.html
* Edit the NGINX configuration file : The main configuration file is located at /etc/nginx/nginx.conf on Linux

example

server {

listen 80;

server\_name localhost;

location / {

root /var/www/html;

index index.html;

}

}

* Restart NGINX to apply the changes : sudo systemctl restart nginx
* Check NGINX Status : sudo systemctl status nginx
* Check the NGINX Error Log : sudo tail -f /var/log/nginx/error.log

**Reverse Proxy Configuration**

Replace the default server block or add a new one to define the reverse proxy settings:

server {

listen 80;

server\_name example.com;

location / {

proxy\_pass http://127.0.0.1:8080;

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;

}

}

**Load Balancing**

upstream backend {

server backend1.local;

server backend2.local;

}

server {

listen 80;

**Monitoring and Logging**

Nginx logs all requests and errors, which is essential for troubleshooting and monitoring.

log\_format main '$remote\_addr - $remote\_user [$time\_local] "$request" '

'$status $body\_bytes\_sent "$http\_referer" '

'"$http\_user\_agent" "$http\_x\_forwarded\_for"';

access\_log /var/log/nginx/access.log main;

location / {

proxy\_pass http://backend;

}

}

**Common Errors**

• 403 Forbidden: Typically means Nginx can’t access the requested file.

• 500 Internal Server Error: Often indicates a problem with the backend server or a misconfiguration in Nginx.

**Conclusion**

Nginx is a powerful and versatile web server that has become a cornerstone in the modern web infrastructure. From its origins as a solution to the C10K problem — handling thousands of concurrent connections — it has grown into a multi-purpose tool that can act as a reverse proxy, load balancer, and even a caching server. Its lightweight architecture, coupled with high performance and reliability, makes Nginx the go-to choice for many of the world’s largest and most trafficked websites.