

**Introduction to Nginx**

Nginx, pronounced as "engine-x," is a powerful and lightweight web server, reverse proxy server, and load balancer. It's designed to efficiently handle a large number of concurrent connections, making it a popular choice for high-performance websites.

**Architecture:**

Nginx follows an asynchronous, event-driven architecture. Unlike traditional web servers like Apache, which creates a new process for each connection, Nginx uses a single, master process to manage multiple worker processes. Each worker process can handle thousands of connections simultaneously, making Nginx highly efficient and scalable.

**Use Case:**

Nginx is well-suited for scenarios where there is a need for high concurrency, low memory usage, and efficient handling of static content. It's often used as a front-end web server to serve static files and proxy requests to backend application servers.

**Apache2 and Nginx - A Comparison**

**Apache2:**

Apache, also known as Apache HTTP Server, is one of the most widely used web servers globally. It follows a multi-process, multi-threaded architecture, creating a new process or thread for each connection.

**Nginx vs. Apache2:**

* **Performance:** Nginx is renowned for its high performance and low resource utilization, making it more efficient in handling a large number of concurrent connections compared to Apache.
* **Resource Usage:** Nginx consumes less memory per connection, making it more suitable for resource-constrained environments.
* **Scalability:** Due to its asynchronous architecture, Nginx scales better under heavy loads.

**Use Cases and Reverse Proxy**

**Use Cases:**

1. **Static Content Serving:** Nginx excels at serving static content like images, CSS, and JavaScript files, making it ideal for content delivery.
2. **Load Balancing:** Nginx can distribute incoming traffic across multiple servers, ensuring even load distribution and improving overall system performance.

**Reverse Proxy:**

Nginx's reverse proxy capabilities allow it to act as an intermediary between clients and backend servers. It receives client requests and forwards them to the appropriate backend server, handling tasks like SSL termination and load balancing.

**Load Balancing and Conclusion**

**Load Balancing:**

Nginx can efficiently distribute incoming traffic among multiple backend servers to ensure optimal resource utilization and prevent overload on any single server. This improves system reliability, fault tolerance, and overall performance.

**Conclusion:**

While both Apache2 and Nginx are powerful web servers, Nginx's unique architecture makes it a preferred choice for high-traffic websites, especially those requiring efficient handling of static content and load balancing. Consider the specific requirements of your project when choosing between Apache2 and Nginx, as each has its strengths depending on the use case.

* **NGINX installation on EC2 instance o.s Ubuntu**

1. **Update Package Lists:**

sudo apt update

1. **Install Nginx:** Install Nginx using the following command:

sudo apt install nginx

1. **Start Nginx:** Once the installation is complete, start the Nginx service:

sudo service nginx start

This command starts the Nginx service, and it will now be running on your EC2 instance.

1. **Enable Nginx to Start on Boot:** To ensure Nginx starts automatically when your server restarts, enable it as a startup service:

sudo systemctl enable nginx

1. **Verify Nginx Installation:** Open your web browser and enter your EC2 instance's public IP address or domain name. You should see the default Nginx welcome page, indicating that Nginx is successfully installed and running.

**Main Configuration File:**

/etc/nginx/nginx.conf

**Site-Specific Configuration:**

/etc/nginx/sites-available/: This directory contains individual configuration files for different sites or applications.

/etc/nginx/sites-enabled/: This directory contains symbolic links to the configuration files in sites-available for the sites you want to enable.

**Server Block (Virtual Host) Configuration:**

/etc/nginx/sites-available/default: The default server block configuration file. This is often symlinked to sites-enabled and can be modified for your specific needs.

**Nginx Module Configuration:**

/etc/nginx/modules-available/: Configuration files for Nginx modules.

/etc/nginx/modules-enabled/: Symbolic links to the module configuration files you want to enable.

**Logs:**

/var/log/nginx/: Nginx log files, including access and error logs.