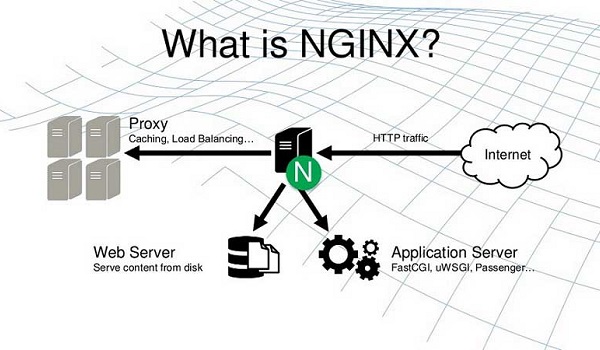
**NGINX là gì?**

***NGINX là một phần mềm web server mã nguồn mở, sử dụng kiến trúc hướng sự kiện (event-driven) không đồng bộ (asynchronous)***. Mục tiêu ban đầu để phục vụ **HTTP cache** nhưng sau được áp dụng vào **reverse proxy**, **HTTP load balancer** và các giao thức truyền mail như **IMAP4, POP3, và SMTP**. Cách thức vận hành của **NGINX là gì?** Cùng tìm hiểu nhé!

NGINX là một phần mềm web server mã nguồn mở đáng tin cậy

**NGINX** chính thức ra đời vào tháng 10/2014. Đây là phần mềm giúp server có tốc độ và khả năng mở rộng lớn nhất, đồng thời, xử lý và thao tác trên hàng nghìn kết nối cùng lúc. Do đó, rất nhiều “ông lớn” công nghệ hiện nay đều lựa chọn **NGINX** như **Google**, **Adobe**, **Netflix**, **WordPress**…

Bạn có thể tìm hiểu thêm về các ***giao thức truyền mail*** trong bài viết: [***POP3, SMTP, IMAP là gì?***](https://wiki.matbao.net/kb/pop3-smtp-imap-la-gi-danh-sach-port-pop3-va-smtp-port-tren-email-server/)

**NGINX hoạt động như thế nào?**

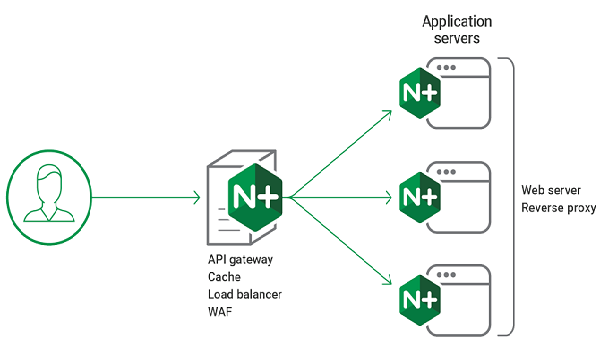
NGINX cũng hoạt động tương tự như các server khác

Về cơ bản, **NGINX** cũng hoạt động tương tự như các web server khác. Khi bạn mở một trang web, trình duyệt của bạn sẽ liên hệ với server chứa website đó. Server sẽ tìm kiếm đúng file yêu cầu của website và gửi về cho bạn. Đây là một trình tự xử lý dữ liệu **single – thread**, nghĩa là các bước được thực hiện theo một trình tự duy nhất. Mỗi yêu cầu sẽ được tạo một thread riêng.

Tuy nhiên, **NGINX**hoạt động theo kiến trúc **bất đồng bộ** (**asynchronous**) **hướng sự kiện** (**event driven**). Nó cho phép các ***threads*** tương đồng được quản lý trong một tiến **process**. Mỗi process hoạt động sẽ bao gồm các thực thể nhỏ hơn, gọi là worker connections dùng để xử lý tất cả threads.

***Worker connections*** sẽ gửi các yêu cầu cho worker process, worker process sẽ gửi nó tới master process, và master process sẽ trả lời các yêu cầu đó. Đó là lý do vì sao một ***worker connection*** có thể xử lý đến 1024 yêu cầu tương tự nhau. Nhờ vậy, **NGINX** có thể xử lý hàng ngàn yêu cầu khác nhau cùng một lúc.

**Các tính năng của NGINX là gì?**

Máy chủ NGINX có nhiều tính năng và ưu điểm vượt trội trong lập trình

***Tại sao nên sử dụng NGINX?*** Để trả lời câu hỏi này ta có thể kể đến các tính năng vượt trội của NGINX như sau:

* Có thể **xử lý hơn 10.000 kết nối** cùng lúc với bộ nhớ thấp;
* **Phục vụ tập tin tĩnh** (static files) và lập chỉ mục tập tin;
* **Tăng tốc proxy** ngược bằng bộ nhớ đệm (cache); cân bằng tải đơn giản và khả năng chịu lỗi;
* Hỗ trợ tăng tốc với bộ nhớ đệm của **FastCGI**, **uWSGI**, **SCGI**, và các **máy chủ memcached**;
* **Kiến trúc modular**; tăng tốc độ nạp trang bằng nén **gzip** tự động;
* **Hỗ trợ mã hoá SSL** và **TLS**;
* **Cấu hình linh hoạt**; lưu lại nhật ký truy vấn;
* **Chuyển hướng lỗi** 3XX-5XX;
* **Rewrite URL** (URL rewriting) dùng regular expressions;
* **Hạn chế tỷ lệ đáp ứng truy vấn**;
* **Giới hạn số kết nối đồng thời** hoặc truy vấn từ 1 địa chỉ;
* Khả năng nhúng mã **PERL**;
* Hỗ trợ và tương thích với **IPv6**; Xem thêm bài viết [***IPv6 là gì?***](https://wiki.matbao.net/kb/ip-la-gi-tong-hop-moi-kien-thuc-can-biet-ve-dia-chi-ip/#ipv6-la-gi)
* Hỗ trợ **WebSockets**;
* Hỗ trợ truyền tải file **FLV** và **MP4**.

**Khác biệt giữa Apache Server và NGINX server là gì?**

So với Apache server, NGINX server có khá nhiều ưu điểm

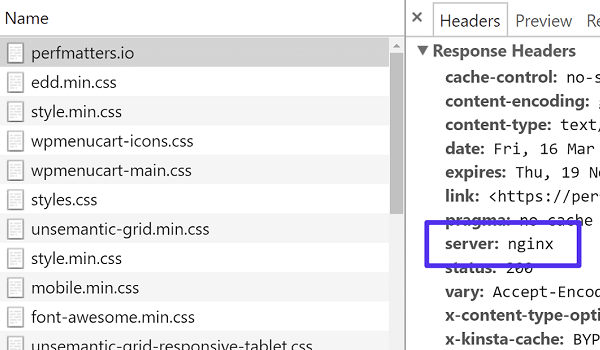
Sự khác biệt cơ bản nhất của **NGINX server** và **Apache server**  là trong cách kết nối được xử lý bởi hai phần mềm.

Trong khi **Apache** sử dụng cơ chế chia luồng (**forked threaded**), hoặc keep-alive, giữ một kết nối mở cho mỗi người dùng thì **NGINX**sử dụng một vòng lặp sự kiện không bị chặn (**non-blocking event loop**), giúp các kết nối vùng (pools connection) hoạt động không đồng bộ thông qua các tiến trình công việc. Nhờ đó, **NGINX**hỗ trợ **CPU** và **RAM** không bị ảnh hưởng bởi những thời điểm có lượng truy cập cao.

Bên cạnh đó, trong khi Apache server có thể xử lý cả nội dung tĩnh bằng cách sử dụng các phương pháp dựa trên file thông thường và nội dung động bằng cách nhúng một bộ xử lý của ngôn ngữ  thì **NGINX**chỉ xử lý được nội dung tĩnh. Vì vậy, cần phải cấu hình server này và bộ vi xử lý dựa trên những giao thức mà nó có thể kết nối được.

So với Apache, NGINX mang nhiều ưu điểm. **NGINX**sở hữu hầu hết các tính năng của Apache. Bên cạnh đó, nó có ưu thể hơn so với Apache nhờ tốc độ xử lý các truy vấn cực cao và hiệu suất sử dụng bộ nhớ máy chủ. Đặc biệt nhất chính là server này lại sử dụng rất ít Ram và CPU đối với một khối lượng truy vấn cực kỳ lớn.

**Hướng dẫn kiểm tra NGINX của Website**

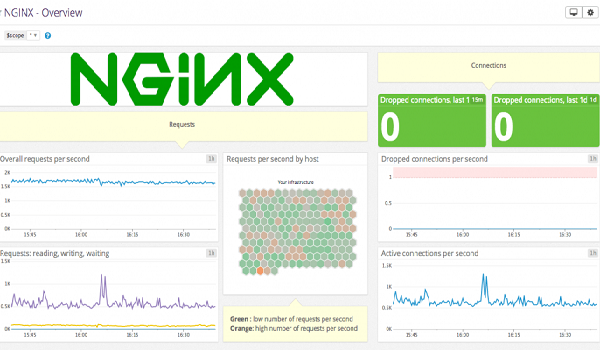
Bạn có thể dựa vào các công cụ sẵn có để kiếm tra website có chạy NGINX

Bạn có thể kiếm tra website có chạy NGINX hay không nhờ vào HTTP header. Bạn có thể sử dụng ***network tab*** trong ***Chrome Devtools*** để thấy HTTP header. Đối với các bạn chưa biết thì thao tác như sau:

* Bạn mở trang cần kiểm tra trên **Chrome**
* Nhấn tổ hợp phím **Ctrl + Shift + I** hoặc **F12**
* Cửa sổ **Chrome Devtools** hiện lên
* Tại đây nếu không thấy **Network tab**, bạn nhấp vào ” **>>** ” để thấy phần mở rộng của tabs và chọn Network.

Ngoài ra, bạn cũng có thể sử dụng [***Pingdom***](https://www.pingdom.com/) hoặc [***GTmetrix***](https://gtmetrix.com/).

**Hướng dẫn cài đặt NGINX Web server**

Bạn có thể tự cài đặt NGINX Web server theo hướng dẫn của chúng tôi

**Bước 1**: **Cài đặt bằng lệnh Yum**

Sử dụng lệnh “**yum install nginx -y**”, **start device** bằng lệnh “**service nginx start**”

**Bước 2**: **Cài đặt Source**

* Cài đặt các package cần thiết, sử dụng các lệnh như sau:

yum groupinstall " Development Tools"  -y

yum install zlib-devel pcre-devel openssl-devel -y

yum install epel-release -y

* Cài đặt thêm các thành phần phụ thuộc của **NGINX** bằng lệnh:

yum install perl perl-devel perl-ExtUtils-Embed libxslt libxslt-devel libxml2 libxml2-devel gd gd-devel GeoIP GeoIP-devel –y

* ***Download source NGINX*** tại: [***https://nginx.org/download/***](https://nginx.org/download/)
* Giải nén và truy cập đường dẫn source **NGINX** vừa giải nén.
* Config từ script bằng lệnh:

./configure --prefix=/etc/nginx --sbin-path=/usr/sbin/nginx --error-log-path=/var/log/nginx/error.log --pid-path=/var/run/nginx.pid --lock-path=/var/run/nginx.lock --user=nginx --group=nginx

make

make install

* Tạo user và phân quyền thư mục:

useradd nginx

chown -R nginx:nginx /etc/nginx/

* Tạo file để chạy lệnh:

cat >> /usr/lib/systemd/system/nginx.service << "EOF"

[Unit]

Description=nginx - high performance web server

Documentation=https://nginx.org/en/docs/

After=network-online.target remote-fs.target nss-lookup.target

Wants=network-online.target

[Service]

Type=forking

PIDFile=/var/run/nginx.pid

ExecStartPre=/usr/sbin/nginx -t -c /etc/nginx/conf/nginx.conf

ExecStart=/usr/sbin/nginx -c /etc/nginx/conf/nginx.conf

ExecReload=/bin/kill -s HUP $MAINPID

ExecStop=/bin/kill -s TERM $MAINPID

[Install]

WantedBy=multi-user.target

EOF

* Start Service **NGINX**

systemctl start nginx

systemctl enable nginx

* Truy cập vào đường dẫn chứa file config của **NGINX**

cd /etc/nginx/conf

* Backup lại file **NGINX**.conf

mv nginx.conf nginx.conf.bak

* Sửa file cấu hình **NGINX**

cat >> nginx.conf << "EOF"

user  nginx;

worker\_processes  1;

error\_log  /var/log/nginx/error.log warn;

pid        /var/run/nginx.pid;

events {

    worker\_connections  1024;

}

http {

    include       mime.types;

    default\_type  application/octet-stream;

    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;

    sendfile        on;

    #tcp\_nopush     on;

    keepalive\_timeout  65;

    #gzip  on;

    include /etc/nginx/conf.d/\*.conf;

}

EOF

* Tạo thư mục chứa các file cấu hình virtualhost cho các website

mkdir -p /etc/nginx/conf.d/

**Bước 3**: **Tiến hành cài đặt php**

* Sử dụng lệnh:

yum install -y php-common php-bcmath php-cli php-devel php-mcrypt php-mysql php-password-compat php-pclzip php-pdo php-gd php-imap php-ldap php-odbc php-pear php-xml php-xmlrpc php-dba php-embedded php-enchant php-mbstring php-intl libssh2 php-pecl-ssh2 php-pecl-memcached php-pecl-memcache php-fpm

* Cấu hình **php-fpm**

sed -i 's/user = apache/user = nginx/'g /etc/php-fpm.d/www.conf

sed -i 's/group = apache/group = nginx/'g /etc/php-fpm.d/www.conf

service php-fpm restart

* Truy cập vào đường dẫn chứa code chung của các website và tạo thư mục riêng chứa code của website, ví dụ: example.com

cd /var/www/html

mkdir example.com

* Truy cập đường dẫn chứa các file cấu hình của **virtualhost**

 cd /etc/nginx/conf.d/

* Backup file cấu hình default

mv default.conf default.conf.bak

* Tạo virtualhost trong **NGINX**

cat >> example.com.conf << "EOF"

server {

    listen       80;

    server\_name  example.com;

location / {

        root   /var/www/html/example.com;

        index  index.html index.php index.htm;

    }

location ~ .php$ {

        root           /var/www/html/example.com;

        fastcgi\_pass   127.0.0.1:9000;

        fastcgi\_index  index.php;

        fastcgi\_param  SCRIPT\_FILENAME  $document\_root$fastcgi\_script\_name;

        include        fastcgi\_params;

    }

}

EOF

* Restart lại service **NGINX**

service nginx restart

**Cấu hình HTTPS trong NGINX**

Dưới đây là hướng dẫn cấu hình HTTPS với NGINX

**Bước 1**: **Cấu hình HTTPS cho NGINX Web Server**: Trong config server, comment dòng listen trên **cổng 80**, chỉ để listen trên **cổng 443**.

**Bước 2**: **Redirect tất cả các truy vấn tới HTTPS** (Cho tất cả các domain)

Thêm một config server có nội dung như sau:

server {

listen 80 default\_server;

server\_name \_;

return 301 https://$host$request\_uri;

}

Trong đó, server\_name là “\_” có nghĩa là bao gồm tất cả các domain của bạn, và **return 301** có nghĩa là bạn muốn cho google biết là bạn thay đổi dường dẫn vĩnh viễn về **HTTPS**.

**Bước 3**: **Redirect chỉ domain của bạn**

Thêm một **config server** có nội dung như sau:

server {

listen 80;

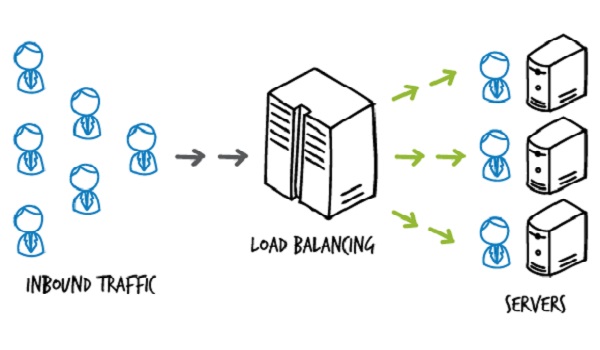
server\_name example.com;

return 301 https://example.com$request\_uri;

}

**Bước 4**: Restart **máy chủ NGINX** và kiểm tra đường dẫn đã được chuyển sang **HTTPS** chưa.

**Sử dụng NGINX làm Load Balancer**

Đây là mô hình Load Balancer với NGINX

**Bước 1**: **Config trên máy 10.10.10.1 như sau**:

upstream proserver {

server 10.10.10.9:9002;

server 10.10.10.10:9002;

}

Trong đó 10.10.10.9 và 10.10.10.10 là 2 servers đang chạy dịch vụ ở cổng 9002.

**Bước 2**: **Config để máy 10.10.10.1 này đón ở cổng 9000 như sau**:

server {

proxy\_buffering off;

client\_max\_body\_size 5M;

listen 9000;

location / {

proxy\_pass http://proserver;

}

}

**Bước 3**:**Start NGINX bằng lệnh: sudo service nginx restart**

**Bước 4: Sửa lại config với tham số weight như sau**:

upstream proserver {

server 10.10.10.9:9002 weight=1;

server 10.10.10.10:9002 weight=2;

}

**NGINX**là máy chủ mới mã nguồn mở mang lại cho website nhiều lợi ích hơn so với các máy chủ khác. Hy vọng với những hướng dẫn của chúng tôi, bạn có thể dễ dàng thực hiện các thao tác đơn giản với web server của mình.