**Cài đặt Nginx.**

==============================================

**Mục Lục.**

* [1. Tổng quan về nginx](https://github.com/hocchudong/ghichep-nginx/blob/master/docs/nginx-install.md#1)
* [2. Cài đặt](https://github.com/hocchudong/ghichep-nginx/blob/master/docs/nginx-install.md#2)

==============================================

**1. Tổng quan về Nginx.**

* Nginx là sản phẩm mã nguồn mở dành cho web server . Là một reverse proxy cho các giao thức HTTP, SMTP , POP3 và IMAP. Nhằm nâng cao hiệu suất xử lý khi sử dụng lượng RAM thấp . Được cấp phép bởi BSD chạy trên nền tảng UNIX, Linux và các biến thể BSD , Mac OS , Solaris, AIX, HP-UX và windows.
* Nginx có thể triển khai nội dung của các trang web động bằng cách sử lý FastCGI, SCGI cho các scripts . Và có thể sử dụng như là một server cân bằng tải . Sau đó vấn đề C10K xuất hiện nói cách khác để cho phép mỗi máy chủ web phải có khả năng xử lý 10.000 khách hàng cùng một lúc. Cần phải phát triển một mạng lưới I / O tốt hơn và công nghệ quản lý chủ đề đã được xuất hiện. Sự xuất hiện của NGinx không phải là kết quả của một nỗ lực để giải quyết vấn đề C10K (như là một vấn đề phổ biến) nhưng “vấn đề C10K” đã thành công trong việc đưa ra các nỗ lực để nâng cao hiệu suất phát triển mạng máy chủ
* Igor Sysoev phát triển nginx từ cách đây hơn 9 năm. Vào tháng 10/2004, phiên bản 0.1.0 được phát hành rộng rãi theo giấy phép BSD. Công dụng của nginx ngoài máy chủ web, còn có thể làm proxy nghịch cho Web và làm proxy email (SMTP/POP3/IMAP). Theo thống kê của Netcraft, trong số 1 triệu website lớn nhất thế giới, có 6,52% sử dụng nginx. Tại Nga, quê hương của nginx, có đến 46,9% sử dụng máy chủ này. Nginx chỉ đứng sau Apache và IIS (của Microsoft).
* Nginx cung cấp gần như tất cả các chức năng máy chủ web:

**2. Cài đặt.**

* Thêm nginx repository bằng việc tạo một file */etc/yum.repos.d/nginx.repo* bằng câu lệnh sau:
* sudo vi /etc/yum.repos.d/nginx.repo

sau đó thêm nội dung này vào file:

[nginx]

name=nginx repo

baseurl=http://nginx.org/packages/mainline/centos/7/$basearch/

gpgcheck=0

enabled=1

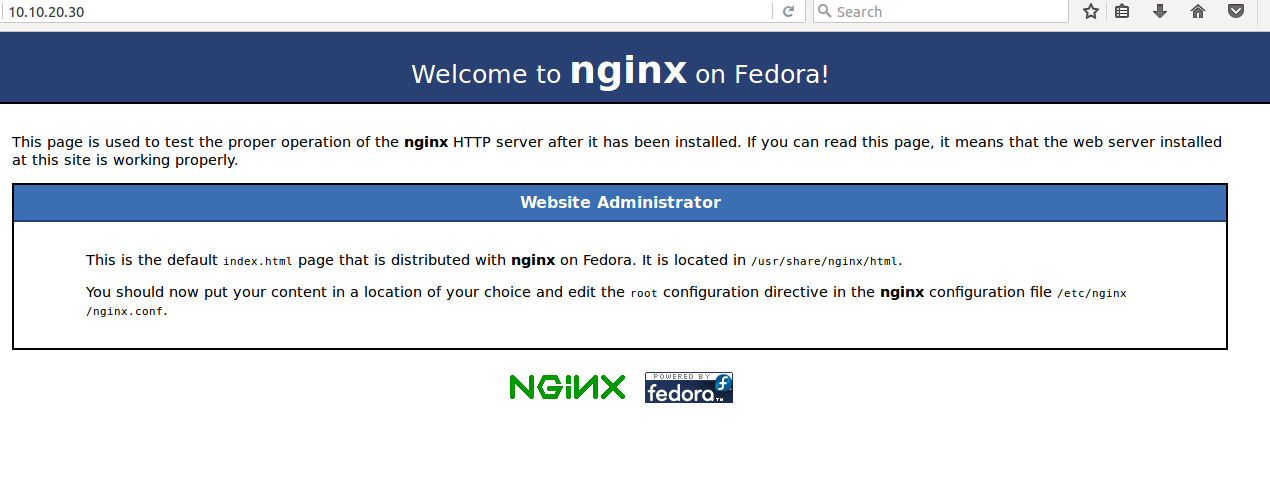
* Update lại hệ thống:
* sudo yum update
* Cài đặt nginx sử dụng câu lệnh:
* sudo yum install nginx
* Khởi động nginx :

systemctl start nginx

* Khởi động firewall-cmd :
* systemctl start firewalld

systemctl enable firewalld

* Cấu hình firewall và restart lại dịch vụ:
* sudo firewall-cmd --permanent --zone=public --add-service=http
* sudo firewall-cmd --permanent --zone=public --add-service=https
* sudo firewall-cmd --reload
* Truy cập vào địa chỉ để kiểm tra :

[](https://github.com/hocchudong/ghichep-nginx/blob/master/images/centos_install.png)

* Cấu hình để nginx tự khởi động sau mỗi lần restart server bằng việc sử dụng câu lệnh:

# systemctl enable nginx

# Cấu hình nginx .

=============================================

## Mục lục.

* [I. giới thiệu về file cấu hình](https://github.com/hocchudong/ghichep-nginx/blob/master/docs/nginx-conf.md#i)
* [II. Giải thích file cấu hình](https://github.com/hocchudong/ghichep-nginx/blob/master/docs/nginx-conf.md#ii)
  + [1. Main block](https://github.com/hocchudong/ghichep-nginx/blob/master/docs/nginx-conf.md#1)
  + [2. Event block](https://github.com/hocchudong/ghichep-nginx/blob/master/docs/nginx-conf.md#2)
  + [3. HTTP block](https://github.com/hocchudong/ghichep-nginx/blob/master/docs/nginx-conf.md#3)

=============================================

## I. Giới thiệu về file cấu hình.

* Mặc định nginx có đường dẫn /etc/nginx/nginx.conf.
* Nginx quản lý cấu hình theo Derective và Block chúng có thể nằm lồng ghép với nhau. Những derective không thuộc block nào sẽ nhóm lại gọi là Main Block những cấu hình trên Block này sẽ ảnh hường tới toàn bộ server.
* Nếu một derective nằm trong block nào đó thì nó có ý nghĩa trong block đó và các block con bên trong , khi derective được định nghĩa lại trong các block con thì nó chỉ có tác dụng trong block con đó.
* File cấu hình :
* # For more information on configuration, see:
* # \* Official English Documentation: http://nginx.org/en/docs/
* # \* Official Russian Documentation: http://nginx.org/ru/docs/
* user nginx;
* worker\_processes auto;
* error\_log /var/log/nginx/error.log;
* pid /run/nginx.pid;
* # Load dynamic modules. See /usr/share/nginx/README.dynamic.
* include /usr/share/nginx/modules/\*.conf;
* events {
* worker\_connections 1024;
* }
* http {
* 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;
* tcp\_nodelay on;
* keepalive\_timeout 65;
* types\_hash\_max\_size 2048;
* include /etc/nginx/mime.types;
* default\_type application/octet-stream;
* # Load modular configuration files from the /etc/nginx/conf.d directory.
* # See http://nginx.org/en/docs/ngx\_core\_module.html#include
* # for more information.
* }
* error\_page 500 502 503 504 /50x.html;
* location = /50x.html {
* }
* }
* # Settings for a TLS enabled server.
* #
* # server {
* # listen 443 ssl http2 default\_server;
* # listen [::]:443 ssl http2 default\_server;
* # server\_name \_;
* # root /usr/share/nginx/html;
* #
* # ssl\_certificate "/etc/pki/nginx/server.crt";
* # ssl\_certificate\_key "/etc/pki/nginx/private/server.key";
* # ssl\_session\_cache shared:SSL:1m;
* # ssl\_session\_timeout 10m;
* # ssl\_ciphers HIGH:!aNULL:!MD5;
* # ssl\_prefer\_server\_ciphers on;
* #
* # # Load configuration files for the default server block.
* # include /etc/nginx/default.d/\*.conf;
* #
* # location / {
* # }
* #
* # error\_page 404 /404.html;
* # location = /40x.html {
* # }
* #
* # error\_page 500 502 503 504 /50x.html;
* # location = /50x.html {
* # }
* # }
* }

## II. Giải thích file cấu hình.

### 1. MAIN BLOCK .

* User nginx; : Cấu hình quy định worker processes được chạy với tài khoản nào , ở đây là nginx.
* worker\_processes auto; : Cấu hình chỉ ra rằng web server được xử lý bằng 1 CPU core (processor) , giá trị này tương ứng với số CPU Core có trên máy chủ. Để kiểm tra số lượng CPU Core trên máy chủ chúng ta dùng lệnh :
* nproc
* # hoặc

cat /proc/cpuinfo

* error\_log /var/log/nginx/error.log; Đường dẫn đến file log của nginx.
* pid /run/nginx.pid; số PID của master process , nginx sử dụng master process để quản lý worker process.

### 2. Event Block .

* worker\_connections 1024; Giá trị này liên quan đến worker processes, 1024 có nghĩa là mỗi worker process sẽ chịu tải là 1024 kết nối cùng lúc . Nếu chúng ta có 2 worker process thì khả năng chịu tải của server là 2048 kết nối tại một thời điểm. Giá trị này chúng ta có thể tùy thuộc vào phần cứng của máy chủ (giá trị 1024/worker process không phải là mặc định).

### 3. HTTP Block .

* log\_format main '$remote\_addr - $remote\_user [$time\_local] "$request" ' '$status $body\_bytes\_sent "$http\_referer" ' '"$http\_user\_agent" "$http\_x\_forwarded\_for"';
* Định nghĩa một mẫu log có tên là main được sử dụng bởi access\_log , các thông tin được đưa vào file tương ứng với các

biến như $remote\_addr, $remote\_user ,....

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

Chỉ ra đường dẫn tới file log .

* sendfile on; Cấu hình này gọi đến function sendfile để xử lý việc truyền file .
* tcp\_nopush on;
* tcp\_nodelay on;
* keepalive\_timeout 65; Xác định thời gian chờ trước khi đóng 1 kết nối, ở đây là 65s.
* include /etc/nginx/mime.types; default\_type application/octet-stream;

Gọi tới file chứa danh sách các file extension trong nginx

* types\_hash\_max\_size 2048;

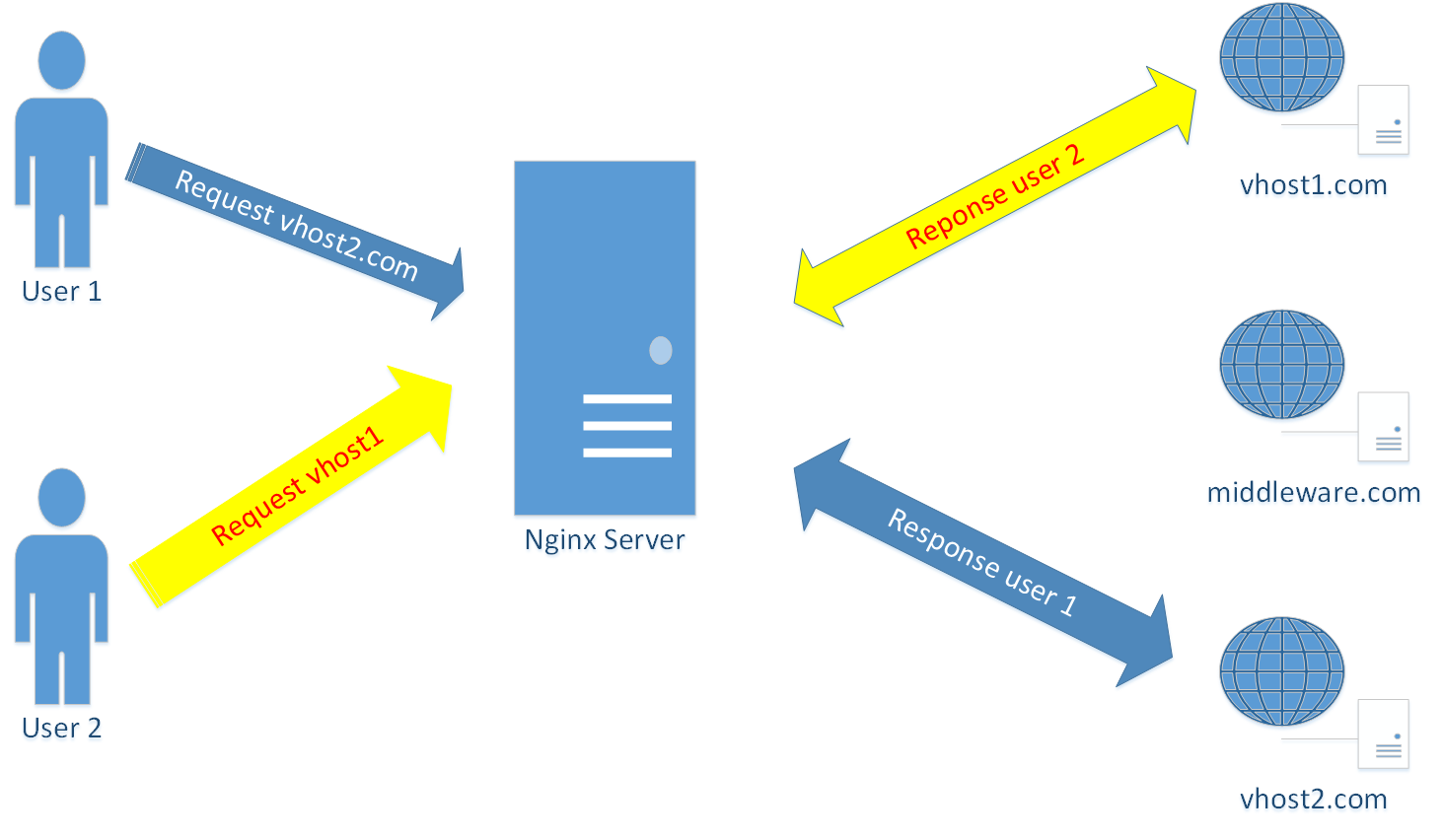
# Cách tạo virtual host (Server Block)

# Mục lục

1. [Virtual Host là gì?](https://github.com/hocchudong/ghichep-nginx/blob/master/docs/nginx-virtualhost.md#virtual-host)
2. [Cấu hình nhiều sử dụng nhiều Virtual Host trên cùng một server](https://github.com/hocchudong/ghichep-nginx/blob/master/docs/nginx-virtualhost.md#multiple-host)

# Nội dung

#### Virtual Host là gì?

* + Virtual Host là một kỹ thuật cho phép nhiều website có thể dùng chung một địa chỉ ip. Thuật ngữ này được sử dụng với các website sử dụng Apache server. Trong các website sử dụng Nginx server thì nó được gọi là Server Block
  + Đây là kỹ thuật dùng để cấu hình cho web server khi bạn muốn có nhiều nhiều website với các tên miền khác nhau được sử dụng chung trên cùng một máy chủ. [](https://github.com/hocchudong/ghichep-nginx/blob/master/images/virhost.png)

#### Cấu hình sử dụng nhiều Virtual Host trên cùng một server

## Lưu ý: Nội dung cấu hình được thực hiện trên hệ điều hành CentOS 7 sử dụng thông qua giao diện dòng lệnh.

* + Nội dung trong phần này sẽ nói về cách để bạn thực hiện cấu hình server sao cho có thể nhiều website cùng sử dụng chung một địa chỉ ip duy nhất. Để thực hiện cấu hình, ta cần trải qua các bước sau:
    - Bước 1: Cài đặt nginx
    - Bước 2: Cấu hình domain chính cho server.
    - Bước 3: Cấu hình tạo ra các virtual host (Server Block)
    - Bước 4: Thực hiện trỏ host trên client để kiểm tra kết quả

Đó là 4 bước chính để thực hiện cấu hình cho nội dung phần này. Chi tiết như sau:

* + Bước 1: Cài đặt nginx

Xem chi tiết [tại đây](https://github.com/hocchudong/ghichep-nginx/blob/master/docs/nginx-install.md#2)

* + Bước 2: Cấu hình domain chính cho server
    - Ta tiến hành sửa nội dung của file cấu hình chính cho nginx tại /etc/nginx/nginx.conf. Tìm tới dòng có nội dung server\_name \_; trong vùng cấu hình của http. Thay \_ bằng tên miền mà bạn muốn sử dụng. Ví dụ:
    - http {
    - ...
    - server {
    - ...
    - server\_name www.middleware.com middleware.com;
    - ...
    - }
    - ...
    - }
  + Bước 3: Cấu hình tạo ra các Virtual Host.
    - Bước 1: Tạo virtual host thứ nhất.
    - # vi /etc/nginx/conf.d/vhost1.com.conf

Tiếp theo ta cần thêm nội dung cấu hình cho file vừa tạo trên với nội dung:

server {

listen 80;

server\_name vhost1.com www.vhost1.com;

access\_log /var/log/nginx/access-vhost1.com.log;

error\_log /var/log/nginx/error-vhost1.com.log;

root /usr/share/nginx/vhost1.com;

index index.php index.html index.htm;

}

* + - * Tạo thư mục chứa website cho virtual host này:
      * # mkdir /usr/share/nginx/vhost1.com
      * # chown nginx:nginx -R /usr/share/nginx/vhost1.com
      * Tạo một file index.html để kiểm tra kết quả:
      * # vi /usr/share/nginx/vhost1.com/index.html
      * Thêm vào file nội dung sau:
      * <DOCTYPE html>
      * <html>
      * <head>
      * <title>www.vhost1.com</title>
      * </head>
      * <body>
      * <h1>Success: You Have Set Up a Virtual Host</h1>
      * <h1>www.vhost1.com and vhost1.com</h1>
      * </body>
      * </html>
    - Bước 2: Tạo ra virtual host thứ hai.
    - # vi /etc/nginx/conf.d/vhost2.com.conf

Tiếp theo ta cần thêm nội dung cấu hình cho file vừa tạo trên với nội dung:

server {

listen 80;

server\_name vhost2.com www.vhost2.com;

access\_log /var/log/nginx/access-vhost2.com.log;

error\_log /var/log/nginx/error-vhost2.com.log;

root /usr/share/nginx/vhost2.com;

index index.php index.html index.htm;

}

* + - * Tạo thư mục chứa website cho virtual host này:
      * # mkdir /usr/share/nginx/vhost2.com
      * # chown nginx:nginx -R /usr/share/nginx/vhost2.com
      * Tạo một file index.html để kiểm tra kết quả:
      * # vi /usr/share/nginx/vhost2.com/index.html
      * Thêm vào file nội dung sau:
      * <DOCTYPE html>
      * <html>
      * <head>
      * <title>www.vhost2.com</title>
      * </head>
      * <body>
      * <h1>Success: You Have Set Up a Virtual Host</h1>
      * <h1>www.vhost2.com and vhost2.com</h1>
      * </body>
      * </html>
    - Bước 4: Tiến hành cấu hình trỏ host trên client để kiểm tra bằng việc thêm nội dung sau vào file C:\Windows\System32\drivers\etc/hosts trên client theo dạng:
    - ip-address server\_name[s]

Ví dụ:

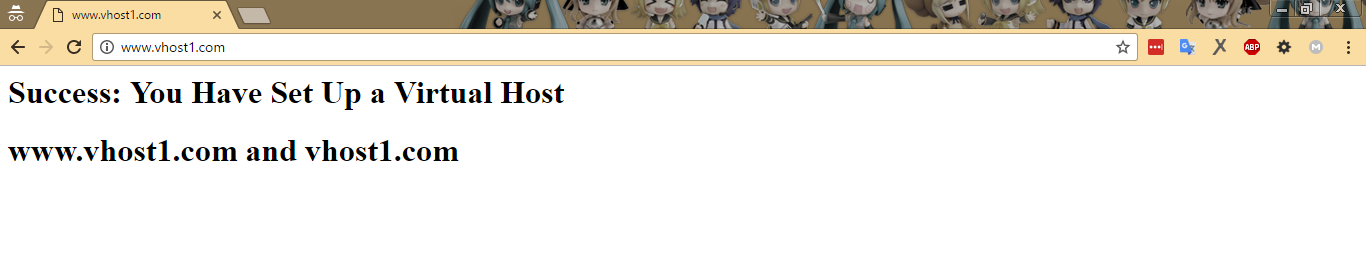
192.168.19.35 middleware.com www.middleware.com vhost1.com www.vhost1.com www.vhost2.com vhost2.com

Kết quả nếu thực hiện đúng, sẽ trong thấy giống như thế này:

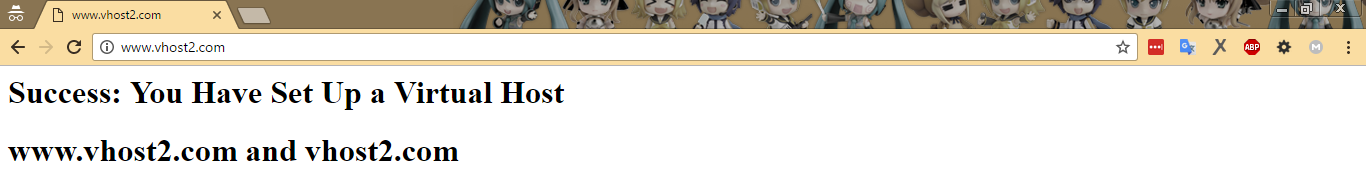
- Khi ta truy cập: middleware.com hoặc www.middleware.com qua trình duyệt của client:

[](https://github.com/hocchudong/ghichep-nginx/blob/master/images/middleware.png)

- Khi ta truy cập: vhost1.com hoặc www.vhost1.com qua trình duyệt của client:

[](https://github.com/hocchudong/ghichep-nginx/blob/master/images/vhost1.png)

- Khi ta truy cập: vhost2.com hoặc www.vhost2.com qua trình duyệt của client:

[](https://github.com/hocchudong/ghichep-nginx/blob/master/images/vhost2.png)

* Chúc các bạn thành công.

# Xác thực để vào page .

* Cài đặt gói httpd-tools

yum -y install httpd-tools

* Dùng trình soạn thảo vi để mở file /etc/nginx/nginx.conf

vi /etc/nginx/nginx.conf

* Thêm vài Block server các thông số sau :
* location /auth {
* auth\_basic "Basic Auth";
* auth\_basic\_user\_file "/etc/nginx/.htpasswd";
* }

# Trong đó /auth là folder mà chúng ta sẽ sử dụng để khi client truy cập vào folder đó cần phải xác thực mới xem được.

* Tạo tài khoản và mật khẩu để xác thực :
* htpasswd -c /etc/nginx/.htpasswd cent
* New password: # set password
* Re-type new password:
* Adding password for user cent

# cent là tài khoản chúng ta tạo để xác thực.

* Restart lại nginx :

systemctl restart nginx

* Tại Doucument Root tạo ra thư mục auth và file index.html :
* mkdir auth
* cd auth

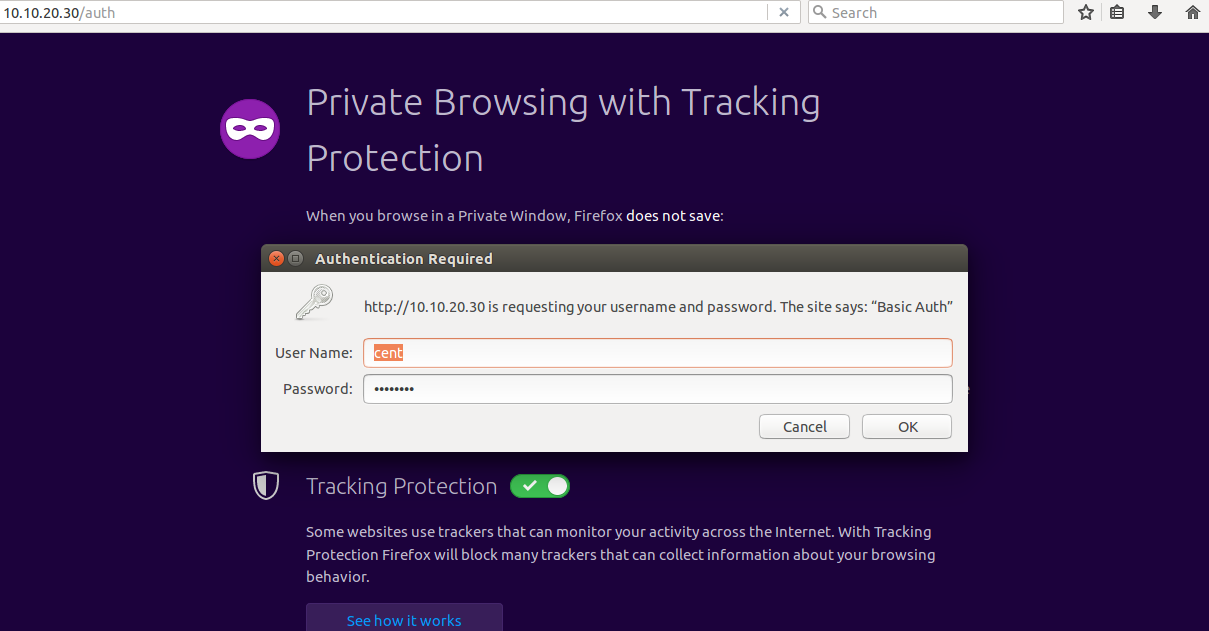
vi index.html

* Thêm nội dung vào file index.html

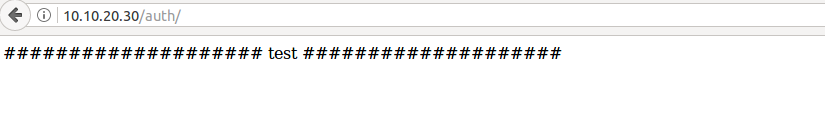
#################### test ####################

## Kiểm tra :

* Truy cập vào địa chỉ chúng ta cấu hình xác thực :

[](https://github.com/hocchudong/ghichep-nginx/blob/master/images/auth.png)

* Nhập tài khoản và mật khẩu vừa thiết lập :

[](https://github.com/hocchudong/ghichep-nginx/blob/master/images/okauth.png)

# Cấu hình Nginx làm reverse proxy cho apache .

====================================================

# Mục lục.

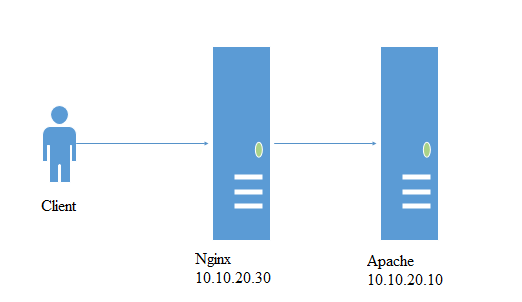
* [Mục đích](https://github.com/hocchudong/ghichep-nginx/blob/master/docs/nginx-reverse-proxy.md#md)
* [Mô hình](https://github.com/hocchudong/ghichep-nginx/blob/master/docs/nginx-reverse-proxy.md#mh)
* [Cài đặt chi tiết](https://github.com/hocchudong/ghichep-nginx/blob/master/docs/nginx-reverse-proxy.md#cd)

====================================================

## Mục đích :

* Apache là một Open Source Webserver phổ biến nhất hiện nay bởi vì có rất nhiều software tuyệt vời hỗ trợ như: cPanel, DirectAdmin,... Điều mà Nginx chưa có.
* Tuy nhiên có một nhược điểm của Apache đó là nó kém linh hoạt, xử lý khá chậm và chiếm rất nhiều bộ nhớ mỗi khi cần xử lý dữ liệu, dù dữ liệu đó là tính hay động.
* Còn với nginx luôn có khả năng xử lý nhanh hơn apache , linh hoạt hơn và nhẹ hơn apache rất nhiều. Cách cấu hình nginx theo đánh giá của nhiều người thì là gọn gàng và đơn giản hơn.
* Nginx rất đa nhiệm do đó để tối ưu hóa hơn cho webserver người ta thường sử dụng song hành Nginx và Apache . Việc sử dụng song hành như thế không gây ảnh hưởng gì , thậm chí còn tiết kiệm được nhiều tài nguyên hơn ,website tải nhanh hơn. Kỹ thuật đơn giản nhất để song hành nginx và apache là làm proxy trung gian để gửi dữ liệu đã xử lý thông tin qua apache đến trình duyệt người dùng. Ở đây chúng ta sẽ xử lý các thông tin về PHP, Python ,... qua module của apache còn nginx sẽ đọc dữ liệu nhận được , xử lý các file tĩnh , cache (nginx sẽ làm tốt hơn trong việc xử lý cache).

## Mô hình.

[](https://github.com/hocchudong/ghichep-nginx/blob/master/images/proxy_reverse.png)

## Cài đặt chi tiết .

### 1. Trên node nginx .

* Thêm repo nginx:

yum install epel-release

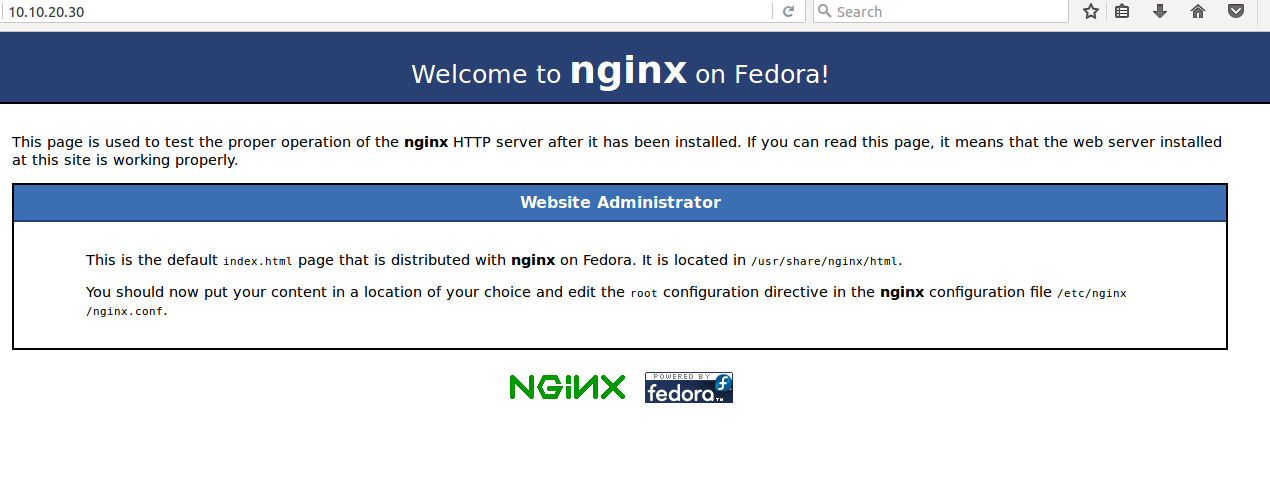
* Cài đặt nginx :
* yum install nginx
* Khởi động nginx :

systemctl start nginx

* Khởi động firewall-cmd :
* systemctl start firewalld

systemctl enable firewalld

* Cấu hình firewall và restart lại dịch vụ:
* sudo firewall-cmd --permanent --zone=public --add-service=http
* sudo firewall-cmd --permanent --zone=public --add-service=https
* sudo firewall-cmd --reload
* Truy cập vào địa chỉ để kiểm tra :

[](https://github.com/hocchudong/ghichep-nginx/blob/master/images/centos_install.png)

* Dùng trình soạn thảo vi mở file cấu hình /etc/nginx/nginx.conf

vi /etc/nginx/nginx.conf

* Tại block server sửa lại thông số như sau :
* server {
* listen 80 default\_server;
* listen [::]:80 default\_server;
* server\_name \_;
* proxy\_redirect off;
* proxy\_set\_header X-Real-IP $remote\_addr;
* proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;
* proxy\_set\_header Host $http\_host;
* location / {
* proxy\_pass http://10.10.20.10/;
* }
* }

# trong đó 10.10.20.10 là địa chỉ của apache

* Restart lại nginx :

systemctl restart nginx

### 2. Cài đặt trên node apache .

* Cài đặt apache :

yum install httpd httpd-devel -y

* Dùng trình soạn thảo vi mở file /etc/httpd/conf/httpd.conf

vi /etc/httpd/conf/httpd.conf

* Tìm đến dòng 196 và sửa lại thành :

LogFormat "\"%{X-Forwarded-For}i\" %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\"" combined

* Restart lại apache :

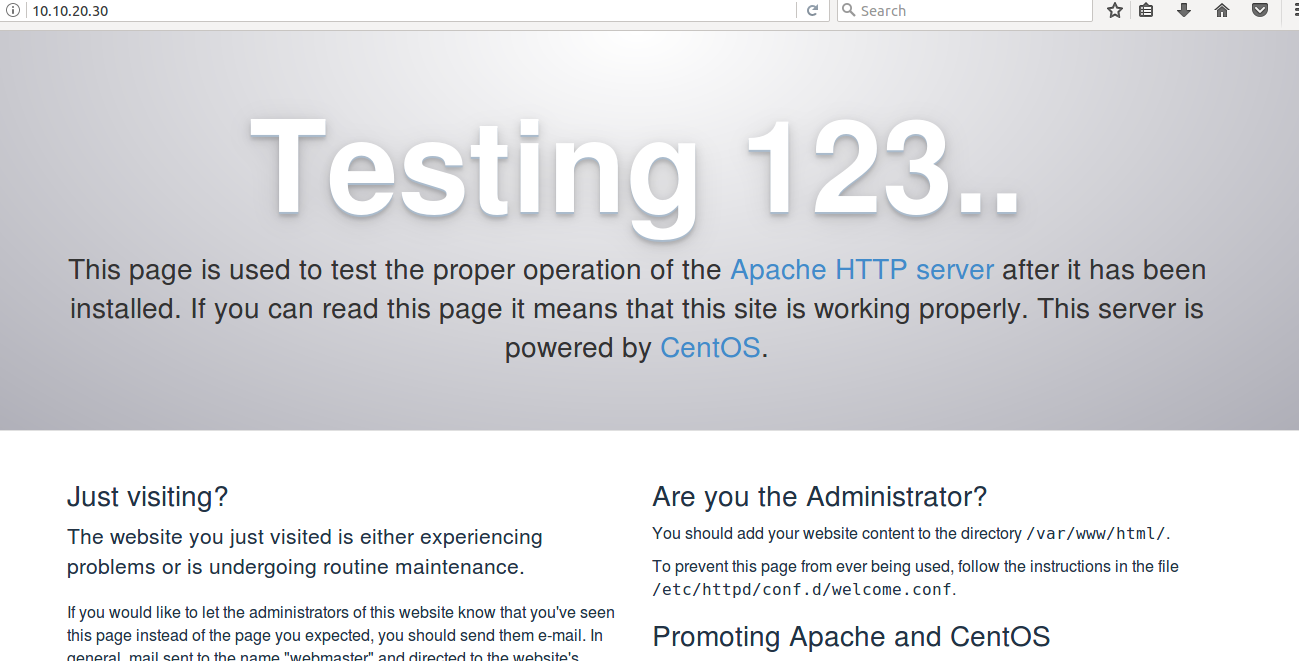
systemctl restart httpd

### 3. Kiểm tra .

* Truy cập vào địa chỉ nginx :

[http://10.10.20.30](http://10.10.20.30/)

Kết quả nhận được :

[](https://github.com/hocchudong/ghichep-nginx/blob/master/images/proxy.png)

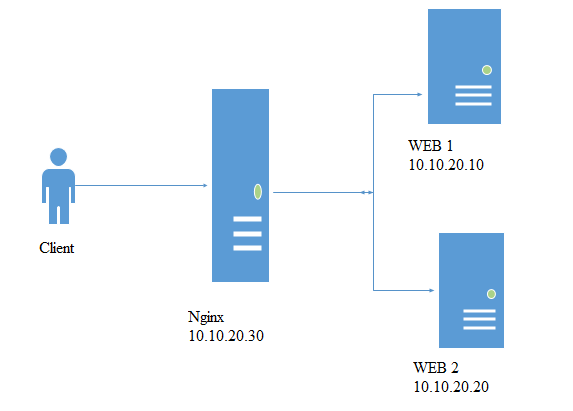
# Cấu hình nginx loadbalancing .

======================================

* [1.  Mô hình](https://github.com/hocchudong/ghichep-nginx/blob/master/docs/nginx-loadbalncing.md#1)
* [2. Một số giải pháp](https://github.com/hocchudong/ghichep-nginx/blob/master/docs/nginx-loadbalncing.md#2)
  + [2.1. Weight load balancing](https://github.com/hocchudong/ghichep-nginx/blob/master/docs/nginx-loadbalncing.md#2.1)
  + [2.2. Round robin](https://github.com/hocchudong/ghichep-nginx/blob/master/docs/nginx-loadbalncing.md#2.2)
  + [2.3. Least connection](https://github.com/hocchudong/ghichep-nginx/blob/master/docs/nginx-loadbalncing.md#2.3)
  + [2.4. Health check](https://github.com/hocchudong/ghichep-nginx/blob/master/docs/nginx-loadbalncing.md#2.4)
  + [2.5. Kết hợp các thuật toán](https://github.com/hocchudong/ghichep-nginx/blob/master/docs/nginx-loadbalncing.md#2.5)

======================================

## 1. Mô hình.

[](https://github.com/hocchudong/ghichep-nginx/blob/master/images/loadbalancing.png)

## 2. Một số giải pháp.

* Cân bằng tải là một kỹ thuật thường dùng để tối ưu hóa việc sử dụng tài nguyên , tối đa hóa thông lượng , giảm độ trễ về đảm bảo tính chịu lỗi.
* Chúng ta có thể sử dụng nginx như là một bộ cân bằng tải để phân phối lưu lượng truy cập đến các máy chủ nhằm mục đích cải thiện hiệu năng , khả năng mở rộng và độ tin cậy của các ứng dụng web với nginx.
* Có rất nhiều thuật toán được xây dựng cho việc cân bằng tải, mỗi thuật toán đều có những ưu nhược điểm khác nhau, trong mỗi trường hợp sẽ có được tác dụng riêng, chúng ta cũng có thể kết hợp các thuật toán với nhau để giúp cho hệ thống của chúng ta hoạt động được tốt hơn. Tùy vào cơ sở hạ tầng và mục đích sử dụng thì chúng ta sẽ lựa chọn thuật toán phù hợp với hệ thống . Sau đây là một số thuật toán cân bằng tải.

### 2.1. Weighted load balancing.

* Đây là một thuật toán quan trọng trong loadbalancing, khi sử dụng thuật toán này sẽ giúp chúng ta giải quyết đươc bài toán phân chia các server xử lý. Vói mặc định của nginx sử dụng thuật toán round-robin thì các request sẽ được chuyển luân phiên đến các server để xử lý, tuy nhiên đối với Weighted load balancing thì chúng ta sẽ phân ra được khối lượng xử lý giữa các server.
* Ví dụ chúng ta có 2 server dùng để load balancing muốn cứ 5 request đến thì 4 dành cho server 1, 1 dành cho server 2 hay các trường hợp tương tự thì weighted load balancing là sự lựa chọn hợp lý.
* Dưới đây là cách cấu hình chi tiết.

#### 2.1.1. Trên các node apache :

* Cài đặt apache :

yum install httpd httpd-devel

* Khởi động apache :

systemctl start httpd

* Truy cập thư mục /var/www/html

cd /var/www/html

* Tạo file index.html

vi index.html

* Thêm nội dung vào file index.html :

################### WEB 1 (2) #################

#### 2.1.2. Trên node nginx.

* Thêm repo nginx:

yum install epel-release

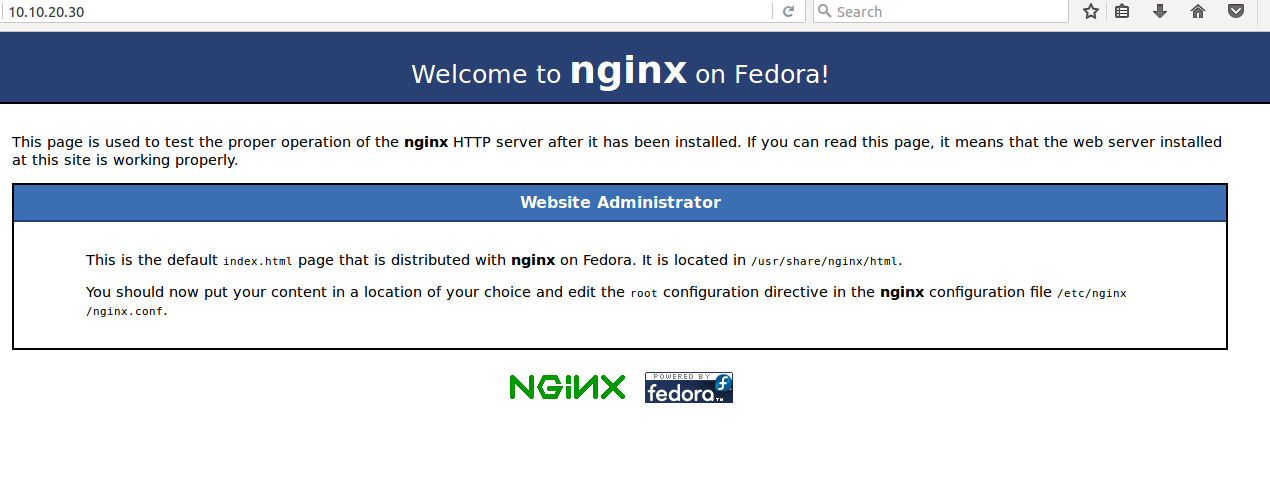
* Cài đặt nginx :
* yum install nginx
* Khởi động nginx :

systemctl start nginx

* Khởi động firewall-cmd :
* systemctl start firewalld

systemctl enable firewalld

* Cấu hình firewall và restart lại dịch vụ:
* sudo firewall-cmd --permanent --zone=public --add-service=http
* sudo firewall-cmd --permanent --zone=public --add-service=https
* sudo firewall-cmd --reload
* Truy cập vào địa chỉ để kiểm tra :

[](https://github.com/hocchudong/ghichep-nginx/blob/master/images/centos_install.png)

* Dùng trình soạn thảo vi mở file /etc/nginx/nginx.conf

vi /etc/nginx/nginx.conf

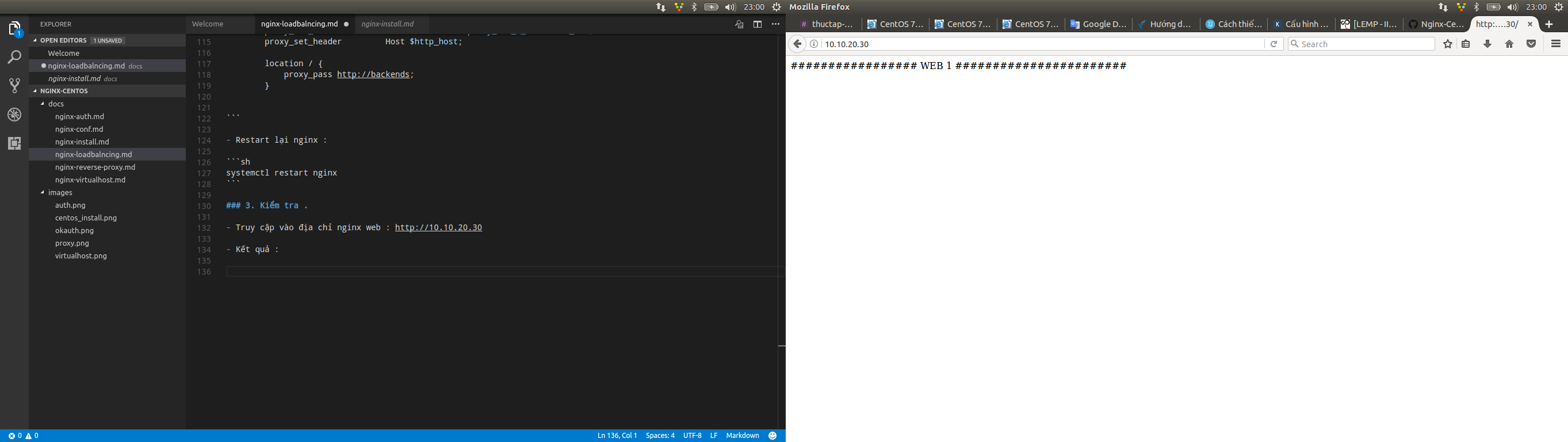
Sửa lại cấu hình như sau :

* Tại block http thêm các cấu hình :
* http {
* upstream backends {
* server 10.10.20.10:80 weight=3;
* server 10.10.20.20:80 weight=2;
* }
* # Cấu hình trên có nghĩa là cứ 5 request gửi tới server sẽ có 3 request vào web 1 và 2 request vào web 2.
* Tại block server thêm hoặc sửa các cấu hình thành như sau :
* server {
* listen 80 default\_server;
* listen [::]:80 default\_server;
* server\_name \_;
* proxy\_redirect off;
* proxy\_set\_header X-Real-IP $remote\_addr;
* proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;
* proxy\_set\_header Host $http\_host;
* location / {
* proxy\_pass http://backends;
* }
* Restart lại nginx :

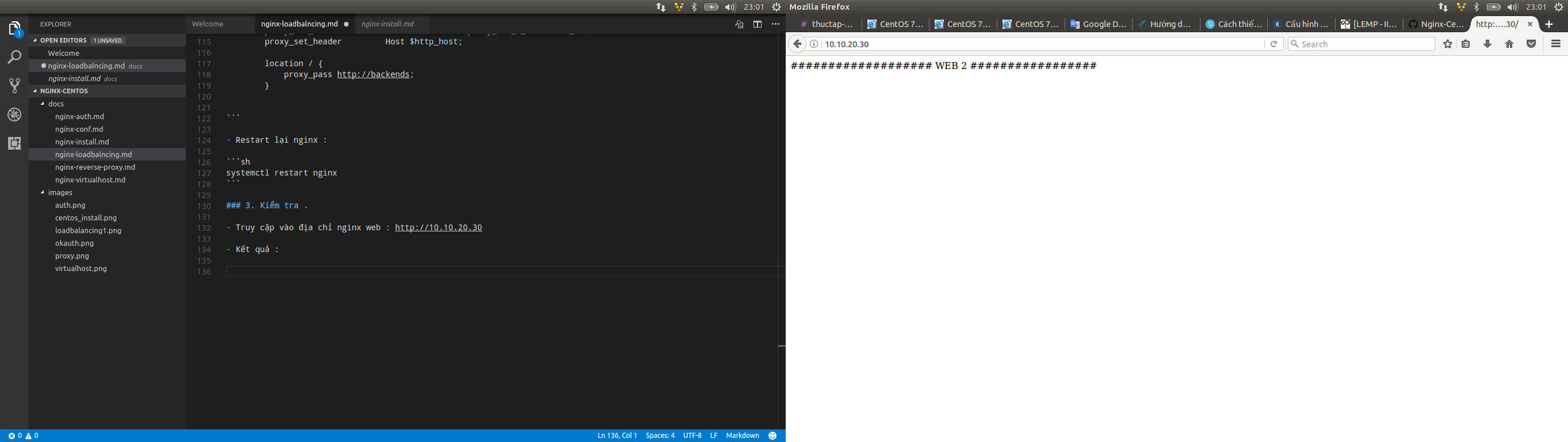
systemctl restart nginx

#### 2.1.3. Kiểm tra .

* Truy cập vào địa chỉ nginx web : [http://10.10.20.30](http://10.10.20.30/)
* Kết quả :

[](https://github.com/hocchudong/ghichep-nginx/blob/master/images/loadbalancing1.png)

Sau 3 request :

[](https://github.com/hocchudong/ghichep-nginx/blob/master/images/loadbalancing2.png)

### 2.2. Round Robin.

* Round Robin là thuật toán mặc định của nginx khi chúng ta không có cấu hình gì thêm trong block http .
* Đặc ddieeerm của thuật toán này là các request sẽ được luân chuyển liên tục giữa các server 1:1 , điều này sẽ làm giải tải cho các hệ thống có lượng request lớn.

#### 2.2.1. Cấu hình tiết.

* Dùng trình soạn thảo vi mở file /etc/nginx/nginx.conf

vi /etc/nginx/nginx.conf

* tại block http sửa lại như sau :
* http {
* upstream backends {
* server 10.10.20.10:80;
* server 10.10.20.20:80;

}

### 2.3. Least connection.

* Đây là thuật toán nâng cấp của round robin và weighted load balancing, thuật toán này sẽ giúp tối ưu hóa cân bằng tải cho hệ thống.
* Đặc điểm của thuật toán này là sẽ chuyển request đến cho server đang xử lý it hơn làm việc, thích hợp đối với các hệ thống mà có các session duy trì trong thời gian dài, tránh được trường hợp các session duy trì quá lâu mà các request được chuyển luân phiên theo quy tắc định sẵn , dễ bị down 1 server nào đó do xr lý qúa khả năng của nó.

#### Cấu hình .

vi /etc/nginx/nginx.conf

* tại block http sửa lại như sau :
* http {
* upstream backends {
* least\_conn;
* server 10.10.20.10:80;
* server 10.10.20.20:80;

}

### 2.4. Health check.

* Thuật toán này xác định máy chủ sẵn sàng xử lý request để gửi request đến server , điều này tránh được việc phải loại bỏ thủ công một máy chủ không sẵn sàng xử lý.
* Các hoạt động của thuật toán này là nó sẽ gửi một kết nối TCP đến máy chủ , nếu như máy chủ đó lắng nghe trên địa chỉ và port đã cấu hình thì nó mới gửi request đến cho server xử lý.
* Tuy nhiên health check vẫn có lúc kiể tra xem máy chủ có sẵn sàng hay không, đối với các máy chủ cơ sở dữ liệu thì health check không thể làm điều này.

#### Cấu hình .

vi /etc/nginx/nginx.conf

* tại block http sửa lại như sau :
* http {
* upstream backends {
* server 10.10.20.10:80;
* server 10.10.20.20:80 max\_fails=3 fail\_timeout=5s;
* server 10.10.20.10:80;

}

### 2.5. Load balancing kết hợp thuật toán.

* Các thuật toán không bao giờ có thể hữu dụng trong tất cả các trường hợp,việc lựa chọn thuật toán dựa trên cơ sở hạ tầng chúng ta có cũng như mục đích sử dụng, để có thể tối ưu hóa hơn trong việc cân bằng tải thông thường chúng ta sẽ kết hợp các thuật toán lại với nhau để có thể đưa ra được giải pháp cân bằng tải hợp lý nhất cho hệ thống. Sau đây là một số giải pháp kết hợp.

#### 2.5.1. Kết hợp least balancing và weight load balancing.

* Thuật toán least load balancing giúp hệ thống có thể lựa chọn server đang xử lý ít hơn để gửi request cho server đó xử lý . Ngoài ra nó còn có thể tự loại bỏ server bị lỗi trong vòng xử lý của nó. Tuy nhiên least load balancing chỉ hữu hiệu khi chúng ta có 2 server có cùng cấu hìn. Giả sử chúng ta có 2 server , server1 có cấu hình mạnh ghấp 2 lần server2 thì chúng ta dùng least load balancing thì đến một thời điểm nào đó con server2 rất dễ bị tèo. Do đó để tránh trường hợp này chúng ta có một giải pháp có thể giảm thiểu khả năng tèo của con thứ 2 đó là kết hợp thêm thuật tóan weighted load balacing :
* Chi tiết cấu hình trong trường hợp trên như sau :
* http {
* upstream backends {
* least\_conn;
* server 10.10.20.10:80 weight=2;
* server 10.10.20.20:80 weight=1;

}

* Tham khảo thêm một số giải pháp tại [đây](https://www.nginx.com/blog/load-balancing-with-nginx-plus-part2/?_ga=1.129223048.568102583.1493302924)