**CẤU HÌNH WEB SERVER**

**Một vài câu lệnh cơ bản.**

* Sudo apt-get update -y
* Sudo apt-get upgrade
* Cài đặt nginx: sudo apt-get install nginx
* Kiểm tra trạng thái dịch vụ nginx: sudo systemctl status nginx
* Kiểm tra firewall: sudo ufw app list

Text

Description automatically generated

* Lúc này, tại **Error! Hyperlink reference not valid.**> ta sẽ được điều hướng đến trang web mặc định của nginx. Trang mặc định này là site html được lưu tại /var/www/html/:

Graphical user interface, text, application, email

Description automatically generated

* Thay trang này bằng 1 trang khác (index.html)

Graphical user interface, application

Description automatically generated

* **Một số lưu ý quan trọng!!!**
* Tất cả các file cấu hình của nginx trong /etc/nginx

Text

Description automatically generated

* File cấu hình **chính** cho nginx : /etc/nginx/nginx.conf -> include các file trong thư mục **site-enabled**
* Thư mục **site-available:** các site đã chuẩn bị trong máy, tuy nhiên người dùng sẽ không truy cập đến đó được, cần phải liên kết đến thư mục **site-enable**
* Nginx gồm các module. Các module này được điều khiển bởi các directive trong file cấu hình. Directive nói một các dễ hiểu là một instruction – 1 chỉ dẫn. Directives được chia thành 2 loại là: directive đơn giản và block directive.
  + 1 directive đơn giản gồm tên và tham số, kết thúc bằng ;. Vd: (trong file nginx.conf)

****

* + Một block sẽ được bọc trong {}. Vd:

**Graphical user interface, text

Description automatically generated**

* 1 tên domain nên tạo một file cấu hình riêng
* NGINX không có Virtual host, thay vào đó là [Server Blocks](https://www.nginx.com/resources/wiki/start/topics/examples/server_blocks/) sử dụng server\_name và nghe các chỉ thị để liên kết với các tcp sockets. Tất cả các file server block phải có định dạng là .conf và được lưu trong thư mục /etc/nginx/conf.d
* Các file ghi log (access.log và error.log ) tại /var/log/nginx



* Document root directory có thể đặt bất kì đâu!!! (là source website đó). một số vị trí thường được sử dụng là:
  + - /home/<user\_name>/<site\_name>
    - /var/www/<site\_name>
    - /var/www/html/<site\_name>
    - /opt/<site\_name>
    - /usr/share/nginx/html
* **Một số thao tác cần lưu ý**
* Trước khi thay đổi file cấu hình, cần sao lưu file lại: tức là copy file cấu hình vào thư mục /etc/nginx/conf/nginx.conf.backup
* Sao khi thay đổi cần restart lại service. Sudo systemctl restart nginx
* **Phân tích file nginx.conf tại /etc/nginx/nginx.conf, tương tự cho file conf khác**
* File bắt đầu bằng 3 directive nằm trong main context (là context chung nhất bao gồm tất cả các directive khác)
* user www-data;
* worker\_processes auto;
* pid /run/nginx.pid;
  + **user:** cho biết người dùng nào có quyền chạy các máy chủ nginx
    - www-data: user của Apache (user mặc định), hoặc đặt username (user của develeper) hoặc nobody
  + **worker-process:** định nghĩa số lượng worker process nginx sử dụng. **số lượng worker process nên đặt bằng số lượng core cpu.**  Với giá trị auto sẽ tự động liên kết giá trị này với số lượng core CPU. Ví dụ với các web server có sử dụng SSL, gzip thì ta nên đặt số worker\_process này lên cao hơn. Để xác định số **core CPU** ta thực hiện lệnh:

cat /proc/cpuinfo | grep processor

* + **pid**: xác định nơi Nginx sẽ ghi lại ID của master process
* **event context**: gồm các directive ở mức độ global và ảnh hưởng đến cách xử lý của nginx ở cấp độ chung. Trong file conf chỉ được định nghĩ 1 block event này thôi.
  + **worker\_connections:** cho biết số lượng connection mà mỗi worker\_process có thể xử lý. Để xem về mức giới hạn sử dụng của hệ thống bạn có thể dụng lệnh: **ulimit –n.** Gía trị này nên nhỏ hơn hoặc bằng giới hạn này: **max\_clients = worker\_connections\*worker\_processes**.
* **http context:** chứa tất cả các chỉ dẫn cần thiếu để xử lý quá trình kết nối **http hoặc https** khi cấu hình nginx như một web server hoặc reverse proxy. Một số lệnh trong đó như:
  + **include**: include một file khác vào trong cấu hình của nginx, giúp dễ đọc, đỡ lẫn lộn
  + **gzip:** giúp nén dữ liệu lại trước khi gửi đến client. Lợi ích là hạn chế số lượng bang thông sử dụng và tăng tốc độ chuyển dữ liệu
* **server context:** được khai báo trong **http context**, được khai báo nhiều lần. dựa vào yêu cầu của client mà nginx sẽ sử dụng thuật toán để lựa chọn server context nào được sử dụng. một số directive trong context này như:
  + **listen:** ip/port lắng nghe của server
  + **server\_name:** nếu có nhiều server match với listen ở trên thì nginx sẽ tiến hành phân tích cú pháp và chọn block server phù hợp.
  + **location context**: sau khi chọn được server context nào tiếp nhận request này thì nginx sẽ phân tích URL của request để tìm ra hướng sử lí của request. VD:

|  |
| --- |
| location = / {  index index.html;  } |

Cấu trúc chung :

|  |
| --- |
| **Location <optional\_modifier> <location\_match>{**  **… directives đây!**  **}** |

trong đó:

* + - optional\_modifier: kiểu so sánh để đối chiếu với location\_match
      * (none): tất cả các request có URL bắt đầu bằng location\_match sẽ được location block này sử lí.

vd: location / {

…………..

}

Thì các request có URL như /index.html, /abc, /xyz,… miễn bắt đầu bằng / thì đều được match với location này.

* + - * = : URL phải chính xác bằng location\_match

Vd: location = / {

…………..

}

Thì các request cho URL **bằng /** mới được sử lí, còn lại thì không.

* + - * ~: sử dụng regular expression cho URL
      * ~\* cũng là sử dụng regular expression nhưng cho phép pass cả chữ hoa và chữ thường.

\*\* Khi một location block được chọn để handle một request thì action sẽ nằm hoàn toàn trong location đó. Nó có thể chuyển hướng xử lí của request bằng các lệnh:

* **Index:** định nghĩa cái trang mặc định mà nginx sẽ phục vụ. có thể trỏ đến nhiều file và file đầu tiên được tìm thấy sẽ được sử dụng. nếu không tìm thấy trang nào => trả về 404

Vd: index index.html index.htm index.php

* **Try\_files:** cố gắng tìm file để điều hướng request, nếu không tồn tại file nào thì nhảy đến location được khai báo ở cuối hoặc URL cụ thể nào đó.

vd:

location / {

try\_files $uri $uri/ /index.html;

}

* Kiểm tra giá trị $uri (giá trị URL mà nginx hiện đang xử lí), nếu không tồn tại kiểm tra tiếp $uri/, nếu không tồn tại thì chuyển hướng đến index.html
* **Error\_page**: chuyển hướng request đến page 404 mình tự cấu hình nếu không tìm được file tham chiếu phù hợp.
* **Proxy\_pass:** sẽ thực hiện gửi request tới một proxy server cụ thể nào đó

**Ví dụ cụ thể và giải thích:**

Trong file cấu hình mình cấu hình một block server đơn giản như sau:

server {

    listen 192.168.1.1:80 default\_server;

    root /var/www/html;

    location / {

        index index.html index.htm index.nginx-debian.html;

    }

location ~ \.php {

        ….

    }

}

* Default\_server: thiết lập server mặc định cho bất kỳ request nào match với IP và port được chỉ định.
* Document root được đặt tại thư mục /var/www/html/
* Đầu tiên nginx sẽ kiểm tra IP address và port của request sau đó đến server\_name (nếu có, trong trường họp có nhiều server lắng nghe tại cùng 1 port)
* Tiếp theo sẽ tìm kiếm location phù hợp (không theo thứ tự liệt kê trong file conf). vd như conf ở trên, do location / sẽ match với tất cả các request nên nó sẽ được xếp cuối. Phần request được xử lí sẽ chỉ là một phần của URL, không bao gồm tham số. vd: <http://192.168.1.1/index.php?user=a> thì phần dùng để truy vấn là /index.php => sẽ match với location thứ 2. Sau đó nó sẽ map root vào với request thành /var/www/html/index.php và gửi về cho người dùng. Nếu không tồn tại thì trả về 404.
* **Cơ bản: thành một webserver trả về cho người dùng khi người dùng thực hiện request** [**http://ip:80/**](http://ip:80/) **sẽ cung cấp file index.html hoacjw (PHP)**

1. Cài đặt Nginx

ở đây là cài rồi!!!

* Đầu tiên trong **/etc/nginx/nginx.conf** comment hoặc xóa bỏ directive

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

tại dòng 62. các conf của mình sẽ chỉ nằm ở thư mục sites-availabled thui nên không cần cái này . trong file này đã include thư mục **sites-enabled (đã nói ở trang 2).** Muốn người dung truy cập đưdduoc đến server thì cần cấu hình 1 **server có sẵn trong thư mục sites-availabled** và liên kết nó tới thư mục site-enabled

1. Cài đặt mysql

sudo apt install mysql-server mysql-client -y

systemctl start mysql

systemctl enable mysql

1. Cài đặt PHP-FPM

sudo apt install php7.2 php7.2-fpm php7.2-cli php7.2-curl php7.2-mysql php7.2-curl php7.2-gd php7.2-mbstring php-pear -y

systemctl enable php7.2-fpm

sudo systemctl enable php7.2-fpm

1. Cấu hình nginx server

Do mạng ktx nên t test với locahost @@

sudo cp sites-available/default conf.d/default.conf

sudo rm -f sites-available/default

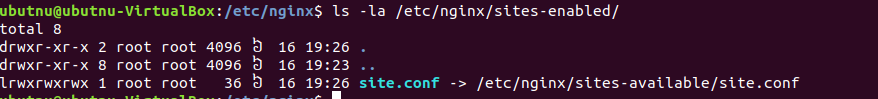
sudo rm /etc/nginx/sites-enabled/default

cd sites-available/

sudo gedit site.conf

|  |
| --- |
| server {  listen 127.0.0.1:80 default\_server;  root /var/www/html;  location / {  index index.html;  try\_files $uri $uri/ =404;  }  location ~ \.php$ {  include snippets/fastcgi-php.conf;  fastcgi\_pass unix:/run/php/php7.2-fpm.sock;  }  } |

sudo ln -s /etc/nginx/sites-available/site.conf /etc/nginx/sites-enabled/



Thực hiện kiểm tra:

sudo nginx -t

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

sudo systemctl restart nginx.service

kết quả:

Graphical user interface

Description automatically generated with low confidence

Graphical user interface, text

Description automatically generated

Tham khảo:

<http://nginx.org/en/docs/http/request_processing.html>

<https://dangkyhosting.com/huong-dan-cai-dat-nginx-cung-php-va-mysql-tren-ubuntu-18-04.html>

<https://github.com/banago/simple-php-website/blob/master/index.php>

**CẤU HÌNH REVERSE PROXY**

**Nội dung cấu hình:**

* Reverse cơ bản
* Đảm bảo dữ liệu đúng trong quá trình proxy, thay đổi header
* Dùng buffer để giải phóng cho Server web, tăng tốc độ, cả thiện trải nghiệm người dùng
* Proxy catching để giảm thời gian phản hồi

**Reverse\_proxy cho HTTP/HTTPS server**

Sử dụng proxy\_pass directive được để chuyển yêu cầu đến webserver nào đó. ở đây là trên cùng máy local của mình luôn @@

Vd:

Text, letter

Description automatically generated

Thì khi người dùng gửi request lên <http://127.0.0.1:80> => chuyển hướng đến google.com (đã check)

* fastcgi\_pass: reverse proxy cho máy chủ FastCGI.

Vd khác:

location /VD/

{

proxy\_pass http://127.0.0.1;

}. . .

Khi người dùn gửi request đến /VD/ thì nó sẽ chuyển hướng đến <http://127.0.0.1/VD>. Do mình triển khai trên cùng máy cùng port với server web nên chuyển hướng này nhìn y chan nhau @@

**Xử lí header trong quá trình chuyển hướng request từ proxy đến web server**

* Phải hiểu là request từ proxy -> server sẽ khác với request trực tiếp từ client-> server. Cụ thể là khi proxy nhận request từ client, nó sẽ xác định 2 giá trị trong phần header là **Host và Connection,** đồng thời xóa hết các phần khác trong header. Tiếp theo đặt giá trị **Host** thành **$proxy\_host** và đặt **Connection** thành **Close**
* giá trị Host là giá trị quan trọng nhất trong quá trình proxy. Một số giá trị có thể đặt như:
  + $proxy\_host: Đặt giá trị cho "Host" header là tên miền hoặc địa chỉ IP cùng với cổng trong proxy\_pass
  + $http\_host: Đặt "Host" header thành "Host" header của yêu cầu của client. Header được gửi bởi client luôn luôn hợp lệ và được Nginx coi như các biến. Biến này bắt đầu với tiền tố $http\_ , theo sau là tên header ở định dạng chữ viết thường , với các dấu "/" được thay bằng dấu "\_".
  + $host: Biến này tham chiêu đến tên host của yêu cầu gốc, "Host" header của yêu cầu từ client , hoặc tên server khớp với yêu cầu.

Nên sử dụng $host vì nó linh hoạt và thường cung cấp cho các upstream server một "Host" header chính xác.

Do đó chúng ta cần set các giá trị header này trước khi chuyển hướng nó đến server:

* Dùng directive proxy\_set\_header <trường> <giá trị>

**Dùng buffer để cải thiện tốc độ, trải nghiệm của người dùng**

* Vấn đề ở đây là mình cấu hình cho proxy xem giữa quá trình client -> server => giảm hiệu suất. Khi thực hiện proxy đến server web, tốc độ của 2 kết nối đó sẽ ảnh hưởng đến trãi nghiệm của nguoif dùng
* Có nhiều option cho việc cấu hình này, nhưng quan trong cần lưu ý là:
  + Proxy\_buffers <number> <size>

\*\*\*\*\*\*

Nội dung file config tính tới thời điểm này:

##################################################################################

#phân quyền cho user www-data

# có thể cấp lại

user www-data;

# tự động tính toán worker process cho phù hợp với CPU core

worker\_processes auto;

pid /run/nginx.pid;

events {

    #xác định số lượng connection cho mỗi worker

    worker\_connections 1024;

    #accept càng nhiều kết nối càng tốt

    multi\_accept on;

}

http {

    ################################################################

    #                         BASIC SETTING                        #

    ################################################################

    # sao chép data -> tăng tốc độ truyền data TCP

    sendfile on;

    # tiến hành gửi header

    tcp\_nopush on;

    #không lưu dữ liệu nhỏ vào buffer

    tcp\_nodelay on;

    types\_hash\_max\_size 2048;

    include /etc/nginx/mime.types;

    default\_type application/octet-stream;

    # request timed out, mặc định là 60s

    client\_body\_timeout 10;

    # nếu client ngừng tương tác, giải phóng bộ nhớ, mặc định là 60s

    send\_timeout 2;

    # set thời gian server sẽ đóng connection, mặc định là 75s

    keepalive\_timeout 30;

    # số lượng request từ client có thể thực hiện liên tục (cho môi trường test)

    keepalive\_requests 100000;

    # cho phép server đóng connection với client không còn được tương tác, nó sẽ giúp giải phóng bộ nhớ

    reset\_timedout\_connection on;

    ################################################################

    #                         SSL SETTING                          #

    ################################################################

    ssl\_protocols TLSv1 TLSv1.1 TLSv1.2; # Dropping SSLv3, ref: POODLE

    ssl\_prefer\_server\_ciphers on;

    ##

    # Gzip Settings

    ##

    gzip on;

    # gzip\_vary on;

    # gzip\_proxied any;

    # gzip\_comp\_level 6;

    # gzip\_buffers 16 8k;

    # gzip\_http\_version 1.1;

    # gzip\_types text/plain text/css application/json application/javascript text/xml application/xml application/xml+rss text/javascript;

# giới hạn số lượng connections với mỗi IP

        limit\_conn\_zone $binary\_remote\_addr zone=conn\_limit\_per\_ip:10m;

        # giới hạn số lượng reqest cho session

        limit\_req\_zone $binary\_remote\_addr zone=req\_limit\_per\_ip:10m rate=5r/s;

        # nếu như request body size lớn hơn buffer size, thì sau đó toàn bộ (hoặc một phần)request body được ghi vào file tạm

        client\_body\_buffer\_size  128k;

        # buffer size cho việc đọc client request header (cho môi trường test)

        client\_header\_buffer\_size 3m;

        # số lượng và buffer size tối đa cho các header lớn để đọc từ yêu cầu của client

        large\_client\_header\_buffers 4 256k;

        # đặt thời gian chờ cho client gửi request header (cho môi trương test)

        client\_header\_timeout 3m;

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

    # ghi log

     error\_log /var/log/nginx/error. crit; # chỉ ghi các lỗi critical

    include /etc/nginx/sites-enabled/\*;

}

Nội dung file site.conf:

server {

    listen 127.0.0.1:80 default\_server;

    root /var/www/html;

    #index index.html;

    limit\_conn conn\_limit\_per\_ip 10;

    limit\_req zone=req\_limit\_per\_ip burst=10 nodelay;

    #location mặc định, thường được xét cuối cùng

    location / {

        # đặt lại header cho quá trình proxy

        # đặt lại host chính bằng host trong header của request

        proxy\_set\_header HOST $host;

        # đăt trường X-Forwarded-Proto, http / https

        proxy\_set\_header X-Forwarded-Proto $scheme;

        # đặt  X-Real-IP là giá trị IP của client

        proxy\_set\_header X-Real-IP $remote\_addr;

        #  là một danh sách chứa địa chỉ IP của tất cả server mà client từng được ủy nhiệm

        proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;

        # chuyển hướng request đến webserver

        proxy\_pass http://127.0.1.1:80;

        #index index.html;

        try\_files $uri $uri/ =404;

    }

    #location match với các request với đuôi .php

    location ~ \.php$ {

        # đặt lại header cho quá trình proxy

        # đặt lại host chính bằng host trong header của request

        proxy\_set\_header HOST $host;

        # đăt trường X-Forwarded-Proto, http / https

        proxy\_set\_header X-Forwarded-Proto $scheme;

        # đặt  X-Real-IP là giá trị IP của client

        proxy\_set\_header X-Real-IP $remote\_addr;

        #  là một danh sách chứa địa chỉ IP của tất cả server mà client từng được ủy nhiệm

        proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;

        #file cấu hình mặc định trên nginx

        include snippets/fastcgi-php.conf;

        # reverse proxy cho server fastcgi cho PHP

        fastcgi\_pass unix:/run/php/php7.2-fpm.sock;

    }

}

server{

    listen 127.0.1.1:80;

    root /var/www/html;

    location / {

        index index.html;

    }

}

**Nhớ backup dữ liệu**

sudo cp nginx.conf conf.d/

sudo cp sites-available/site.conf conf.d/

kiểm tra config: sudo /etc/init.d/nginx configtest => OK

CẤU HÌNH HTTPS, SSL CHO NGINX

Vấn

1. Giờ mình cần 3 máy:1 máy reverse proxy, 1 máy master (đóng vai trò server chính) và backup server (thay cho master khi master bị chết)

* Reverse proxy => điều hướng đến master server (bằng địa chỉ ảo): 192.168.167.112
* Master: máy mình làm ở trên!!! Set host only IP thật: 192.168.167.113
* Tạo một máy backup (export máy master ra xài luôn cho lẹ). IP thật: 192.168.167.111

=> đổi cấu hình tại file site.conf

Tại master:

listen 192.168.167.112:80 default\_server;

sudo rm /etc/nginx/sites-enabled/site.conf

sudo ln -s /etc/nginx/sites-available/site.conf /etc/nginx/sites-enabled/

sudo nginx -t

sudo systemctl restart nginx.service

tương tự cho backup nhe!!!

Kiểm tra:

A picture containing graphical user interface

Description automatically generated

Graphical user interface, text, application

Description automatically generated

1. Chỉnh nội dung file index.html để test cho rõ !!
2. Kiểm tra lại xem nginx chạy đúng trên 2 máy chưa

Từ từ nhaaaa

* Trên máy master:

sudo apt-get install keepalived

TEST

* Bước 1 : 1 reverse -> 1 webserver xem nó có trả về khong
  + Reverse proxy: 192.168.0.45
  + Server 1: 192.168.0.46
  + Server 2: 192.168.0.47

Tren server 2:

cd /etc/nginx/sites-available/

sudo gedit site.conf

|  |
| --- |
| server {  listen 192.168.0.47:80 default\_server;  root /var/www/html;  #location mặc định, thường được xét cuối cùng  location / {    index index.html;  }  #location match với các request với đuôi .php  location ~ \.php$ {  #file cấu hình mặc định trên nginx  include snippets/fastcgi-php.conf;  # reverse proxy cho server fastcgi cho PHP  fastcgi\_pass unix:/run/php/php7.2-fpm.sock;  }  } |

sudo rm /etc/nginx/sites-enabled/site.conf

sudo ln -s /etc/nginx/sites-available/site.conf /etc/nginx/sites-enabled/

sudo nginx -t

sudo systemctl restart nginx.service

web1: <http://192.168.130.135/>

web2: <http://192.168.130.134/>

web3: 192.168.130.137

proxy: 192.168.130.136

proxy2: 192.168.130.138

MỘT SỐ BIẾN MÔI TRƯỜNG TRONG NGINX

<https://viblo.asia/p/tim-hieu-va-huong-dan-setup-web-server-nginx-cac-bien-moi-truong-va-bo-tu-cua-module-http-trong-nginx-WrJeYAJVeVO>