Đề tài: Tìm hiểu và triển khai HAProxy

1. Danh sách thành viên & Công việc

Họ & tên	MSSV	Công việc	Tiến độ
Nguyễn Văn Sơn	175A071416	- Cài đặt demo HAProxy - Hướng dẫn sử dụng	70%
Vũ Xuân Tùng	175A071301	- Tìm hiểu tài tiệu về HAProxy - HAProxy có chức năng cụ thể gì - Cách thức hoạt động của HAproxy	100%

2.Nội dung nghiên cứu

a.HAproxy là gì???

HAProxy viết tắt của High Availability Proxy, là công cụ mã nguồn mở nổi tiếng ứng dụng cho giải pháp cân bằng tải TCP/HTTP cũng như giải pháp máy chủ Proxy (Proxy Server). HAProxy có thể chạy trên các mỗi trường Linux, Solaris, FreeBSD. Công dụng phổ biến nhất của HAProxy là cải thiện hiệu năng, tăng độ tin cậy của hệ thống máy chủ bằng cách phân phối khối lượng công việc trên nhiều máy chủ (như Web, App, cơ sở dữ liệu). HAProxy hiện đã và đang được sử dụng bởi nhiều website lớn như GoDaddy, GitHub, Bitbucket, Stack Overflow, Reddit, Speedtest.net, Twitter và trong nhiều sản phẩm cung cấp bởi Amazon Web Service.

b. HAproxy có chức năng cụ thể là Chức năng chủ yếu là giữ đảm bảo hiệu năng, độ tin cậy cho các ứng dụng TCP/HTTP

Cân bằng tải là một phương pháp phân phối khối lượng truy cập trên nhiều máy chủ nhằm tối ưu hóa tài nguyên hiện có đồng thời tối đa hóa thông lượng, giảm thời gian đáp ứng và tránh tình trạng quá tải cho một máy chủ.

HAProxy (High Availability Proxy) là một giải pháp mã nguồn mở về cân bằng tải có thể dùng cho nhiều dịch vụ chạy trên nền TCP (Layer 4), phù hợp với việc cân bằng tải với giao thức HTTP giúp ổn định phiên kết nối và các tiến trình Layer 7.

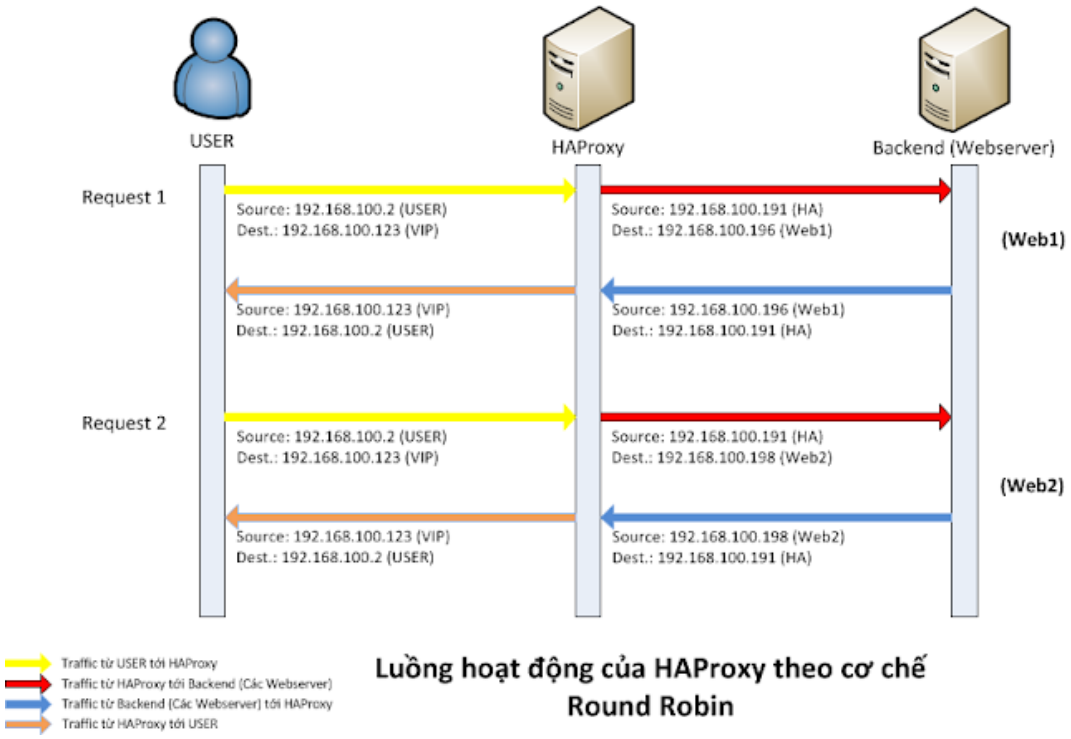
- Cân bằng tải ở Layer 4 chỉ thích hợp cho việc bạn có các webserver có cùng một ứng dụng.
- Cân bằng tải ở Layer 7 có thể phân tải cho các ứng dụng trên một webserver có nhiều ứng dụng cùng domain.

Một số lợi ích khi sử dụng phương pháp cân bằng tải:

- Tăng khả năng đáp ứng, tránh tình trạng quá tải
- Tăng độ tin cậy và tính dự phòng cao
- Tăng tính bảo mật cho hệ thống

c. Cách thức hoạt động của HAproxy

- **Bước 1**: Request từ phía USER đến VIP của HAProxy
- **Bước 2**: Request từ phía USER được HAProxy tiếp nhận và chuyển tới các Webserver
- **Bước 3**: Các Webserver xử lý và response lại HAProxy
- **Bước 4**: HAProxy tiếp nhận các response và gửi lại cho USER bằng VIP



Luồng hoạt động của HAProxy theo cơ chế Round Robin

d. Hướng dẫn cài đặt HaProxy trên Ubuntu

Bước 1: Cài đặt và cấu hình HAproxy.

Các bạn nhập câu lệnh sau vào cửa sổ terminal

```
sudo apt install -y haproxy

sudo apt update
```

Bước 2 : Cấu hình HAproxy.

Các bạn nhập câu lệnh:

```
sudo nano /etc/haproxy/haproxy.cfg
```

Sau đó nhập vào dòng cuối cùng của file haproxy.cfg. Ở đây, <server name> bạn có thể tuỳ chọn đặt, và <private IP> cho những IP mà bạn muốn truyền tới. Bạn có thể xem IP ở **UpCloud Control Panel** và **Private network** tab dưới **Network** menu.

```
frontend http_front
 bind *:80
 stats uri /haproxy?stats
 default_backend http_back

backend http_back
 balance roundrobin
 server <server1 name> <private IP 1>:80 check
 server <server2 name> <private IP 2>:80 check
```

Ở đây là dành cho Load balancer 4 lớp. nó sẽ truy cập vào cổng số 80.

stats URI /haproxy?stats cho phép trang thống kê tại địa chỉ được chỉ định.

Lưu ý với khai báo balance roundrobin, các bạn có thể thay đổi thuật toán loadbalancing nếu thích.
Các bạn có thể tham khảo thuật toán khác tại đây:
https://upcloud.com/community/tutorials/haproxy-load-balancer-ubuntu/

Bước 3: Reset lại Haproxy

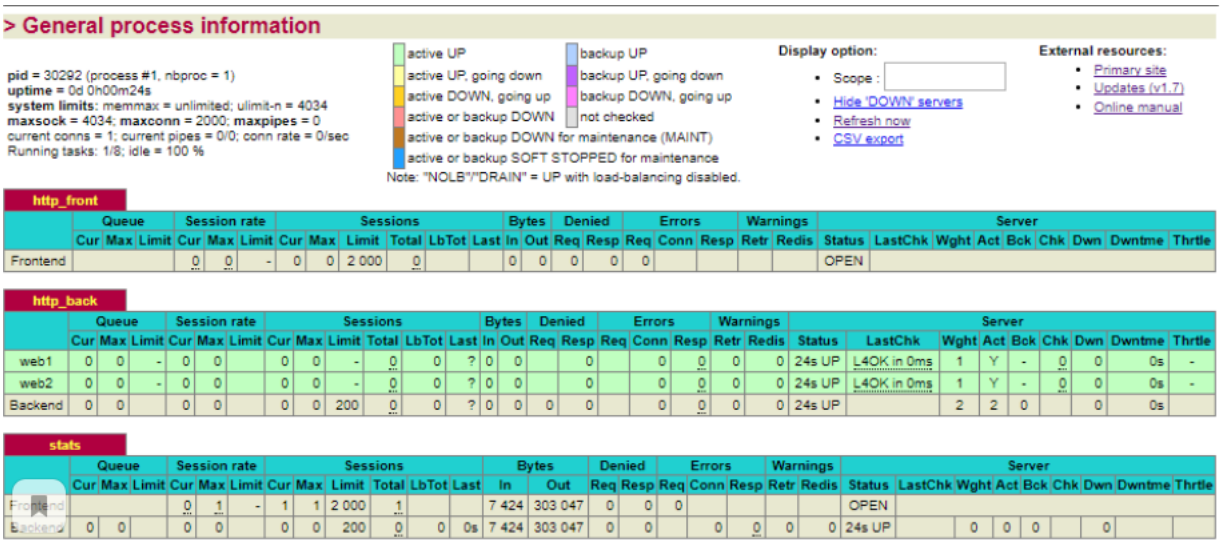
```
sudo systemctl restart haproxy
```

Bước 4: Kiểm tra

Khi bạn tải trang thống kê và tất cả các máy chủ của bạn được liệt kê màu xanh, cấu hình của bạn đã thành công!

```
http://<load balancer public IP>/haproxy?stats
```

Như thế này nhé



e. Hướng dẫn sử dụng HAProxy cho load balancing ứng dụng

Các thuật ngữ trong HAProxy:

Access Control List (ACL)

Trong cân bằng tải, ACL được dùng để kiểm tra điều kiện và thực hiện một hành động (ví dụ như lựa chọn một server hay chặn một request) dựa trên kết quả của việc kiểm tra đó. Việc sử dụng ACL cho phép tạo một môi trường có khả năng chuyển tiếp các request một cách linh hoạt dựa trên các yếu tố khác nhau mà người dùng có thể tùy chỉnh một cách dễ dàng.

Ví dụ của một ACL:

```
acl url_blog       src        /something
```

ACL này sử dụng cho các request có chứa /**something** (vd như: **localhost/something/1/)**và các request này được redirect đến src.

Backend

Backend là một tập các server mà HAProxy có thể forward các request tới. Backend được cấu hình trong mục **backend** trong file configuration của HAProxy. Ở dạng đơn giản nhất, một backend có thể được cài đặt bằng cách

- Đặt ra thuật toán để căn bằng tải (round-robin, least-connection,...)
- Một danh sách các máy chủ và port của chúng có thể dùng để nhận request từ HAProxy

Một backend có thể chứa một hoặc nhiều server, về cơ bản thì càng nhiều server trong một backend thì khả năng chịu tải và performance của hệ thống càng tăng lên. Tính tin cậy của hệ thống cũng tăng lên theo phương pháp này vì khi một server bị ngắt khỏi proxy, các server có thể chịu tải thay được cho nó.

Frontend

Frontend được dùng để định nghĩa cách mà các request được điều hướng cho backend. Frontend được định nghĩa trong mục **fontend** của HAProxy configuration. Các cấu hình cho frontend gồm:

- Một bộ địa chỉ IP và port (ví dụ: 10.0.0.1:8080, *:443,...)
- Các ACL do người dùng định nghĩa
- Backend được dùng để nhận request

Một ví dụ cấu hình của frontend trong file configuration của HAProxy:

```
frontend web

  bind 0.0.0.0

  default_backend web-backend
```

```
frontend forum
  bind 0.0.0.0:8080
  default_backend forum
```

Các loại cân bằng tải

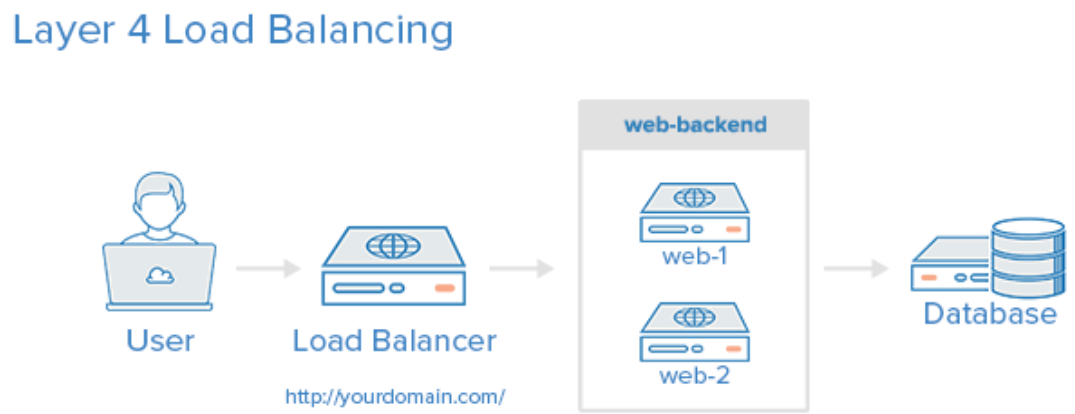
1.Không có cân bằng tải

Đây là một dạng đơn giản nhất cho một ứng dụng web, thường được sử dụng cho môi trường dev, test khi số lượng người dùng ít và không có hoặc không cần đảm bảo tính tin cậy của ứng dụng.

Với mô hình này, người dùng sẽ kết nối trực tiếp với web server tại **yourdomain.com** và không có cân bằng tải nào được sử dụng. Nếu webserver gặp trục trặc, người dùng sẽ không thể kết nối đến ứng dụng web được nữa.

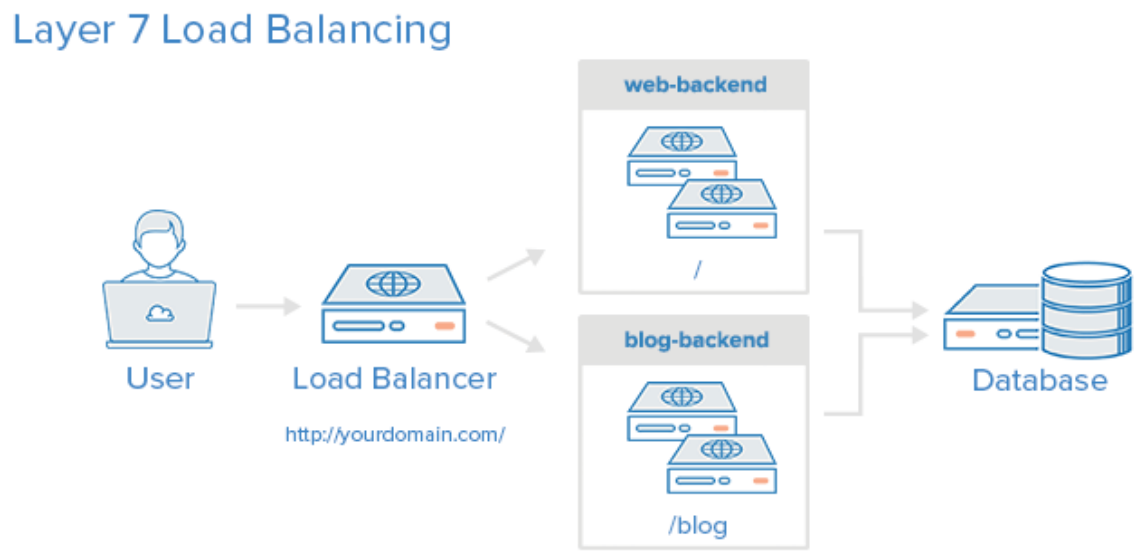
2.Cân bằng tải tại tầng 4 (tầng truyền vận)

Cách đơn giản nhất để có thể cân bằng tải tới nhiều server là sử dụng cân bằng tải trên tầng 4. Theo hướng này thì các request sẽ được điều hướng dựa trên khoảng địa chỉ IP và cổng (ví dụ một request tới địa chỉ **http:://www.example.com/something** sẽ được điều hướng tới backend được dùng để điều hướng cho domain **example.com** với cổng 80)



3.Cân bằng tải tại tầng 7 (tầng ứng dụng)

Cân bằng tải tại tầng 7 là cách cân bằng tải phức tạp nhất và cũng là cách cân bằng tải có nhiều tùy biến nhất. Sử dụng cân bằng tải tại tầng 7, ta có thể điều hướng request dựa trên nội dụng của request đó. Với kiểu câng bằng tải này, nhiều backend có thể được sử dụng cho dùng một domain và port.



Ví dụ, một người dùng request tới **example.com/something**, request đó sẽ được điều hướng đến một backend chuyên dụng cho **something.**

```
frontend web
    bind *:80
    mode http

    acl something_url   path    /something
    use_backend something-server if something_url

    default_backend web-backend
```

Các thuật toán cân bằng tải

Các thuật toán cân bằng tải được sử dụng gồm

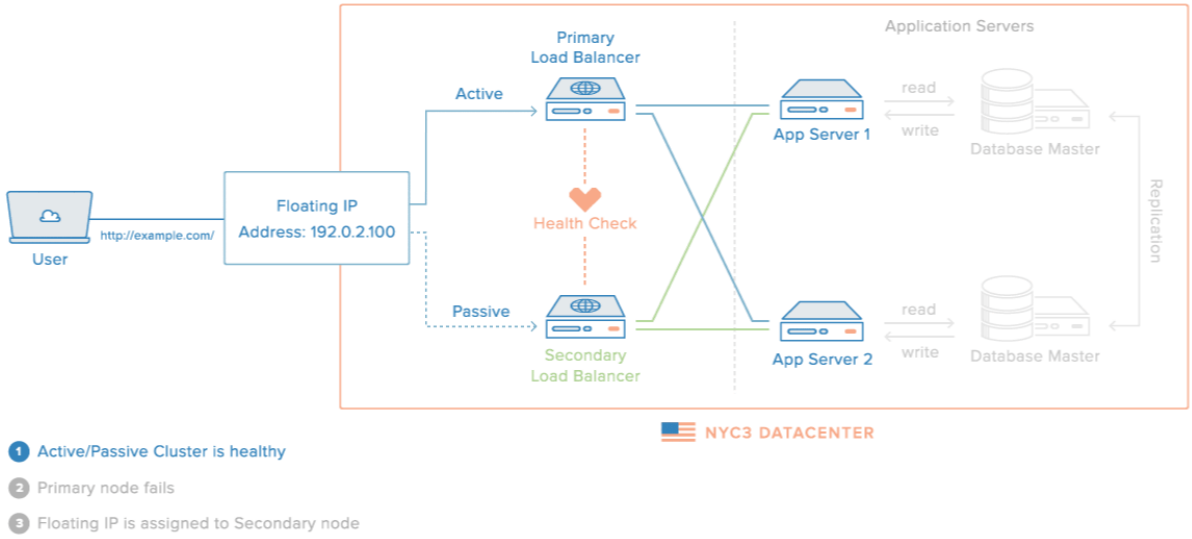
- roundrobin: các request sẽ được chuyển đến server theo lượt. Đây là thuật toán mặc định được sử dụng cho HAProxy
- leastconn: các request sẽ được chuyển đến server nào có ít kết nối đến nó nhất

- source: các request được chuyển đến server bằng các hash của IP người dùng. Phương pháp này giúp người dùng đảm bảo luôn kết nối tới một server

Tính sẵn sàng cao

Cân bằng tải trên tầng 4 và tầng 7 được miêu tả ở trên đều được sử dụng để cân bằng tải để chuyển các request tới các backend server. Tuy nhiên, chính cân bằng tải của bạn là một điểm lỗi (single point of failure), vì nếu cân bằng tải gặp sự cố nhưng các server đều chạy bình thường thì người dùng cũng không thể kết nối đến được ứng dụng web trong khi server vẫn đang chạy bình thường.

Một cài đặt cấu hình cao sẽ không có một điểm lỗi nào. Nó ngăn chặn việc nếu một máy chủ không hoạt động làm ảnh hưởng tới toàn bộ hệ thống.



Với mô hình có 2 cân bằng tải như trên, nếu một trong 2 cân bằng tải không hoạt động, ứng dụng vẫn có thể hoạt động bình thường. Hay nếu một trong hai server không hoạt động, server còn lại sẽ có thể chịu tải thay cho nó.