PRÁCTICA 3

Emilio Aparicio Benítez

Para la realización de esta práctica, vamos a necesitar l**as dos máquinas que usamos en la práctica anterior** junto con una **nueva máquina** que actuará como **balanceador** software de carga que no podrá tener instalada ningún software que se apodere del puerto 80.

PARTE CON NGINX

Instalé nginx en mi máquina 3:

```
sudo apt-get update && sudo apt-get dist-upgrade && sudo apt-get autoremove

sudo apt-get install nginx

sudo systemctl start nginx
```

Y a partir de ahí empecé a trabajar con nginx. Empecé modificando el contenido del archivo /etc/nginx/conf.d/default.conf con el siguiente comando:

sudo nano /etc/nginx/conf.d/default.conf y en él escribí:

```
upstream apaches {
        server 192.168.1.100;
        server 192.168.1.101;
}
server{
        listen 80;
        server_name balanceador;

        access_log /var/log/nginx/balanceador.access.log;
        error_log /var/log/nginx/balanceador.error.log;
        root /var/www/;

        location /
        {
                proxy_pass http://apaches;
                proxy_set_header Host $host;
                proxy_set_header X-Real-IP $remote_addr;
                proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
                proxy_http_version 1.1;
                proxy_set_header Connection "";
        }
}
```

Este es un ejemplo del algoritmo ***"Round-Robin"*** con la misma prioridad para todos los servidores.

Veamos en el siguiente ejemplo su ejecución:

```
swap1@swap1:/var/www$ curl 192.168.1.102
<h1> Soy Maquina 1</h1>
swap1@swap1:/var/www$ curl 192.168.1.102
<h1> Soy Maquina 2</h1>
swap1@swap1:/var/www$ curl 192.168.1.102
<h1> Soy Maquina 1</h1>
swap1@swap1:/var/www$ curl 192.168.1.102
<h1> Soy Maquina 2</h1>
swap1@swap1:/var/www$ curl 192.168.1.102
<h1> Soy Maquina 1</h1>
```

Posibles modificaciones del algoritmo:

Añadiendo un modificador "weight", al que le damos un valor numérico que indica la carga que le asignamos. Por defecto tiene el valor 1, por lo que si no modificamos ninguna máquina del grupo, todas recibirán la misma cantidad de carga:

```
upstream apaches {
        server 192.168.1.100 weight=1;
        server 192.168.1.101 weight=2;
}
```

Cuyo resultado sería:

```
swap1@swap1:/var/www$ curl 192.168.1.102
<h1> Soy Maquina 2</h1>
swap1@swap1:/var/www$ curl 192.168.1.102
<h1> Soy Maquina 1</h1>
swap1@swap1:/var/www$ curl 192.168.1.102
<h1> Soy Maquina 2</h1>
swap1@swap1:/var/www$ curl 192.168.1.102
<h1> Soy Maquina 2</h1>
swap1@swap1:/var/www$ curl 192.168.1.102
<h1> Soy Maquina 1</h1>
swap1@swap1:/var/www$ curl 192.168.1.102
<h1> Soy Maquina 2</h1>
swap1@swap1:/var/www$ curl 192.168.1.102
<h1> Soy Maquina 2</h1>
swap1@swap1:/var/www$ curl 192.168.1.102
<h1> Soy Maquina 1</h1>
swap1@swap1:/var/www$ curl 192.168.1.102
<h1> Soy Maquina 2</h1>
```

Podríamos hacer un balanceo por IP (todo el tráfico que venga de una IP se sirva durante toda la sesión por el mismo servidor final usando la directiva ip_hash al definir el upstream):

```
upstream apaches {
        ip_hash;
        server 192.168.1.100;
        server 192.168.1.101;
}
```

Como resultado:

```
swap1@swap1:/var/www$ curl 192.168.1.102
<h1> Soy Maquina 1</h1>
swap1@swap1:/var/www$ curl 192.168.1.102
<h1> Soy Maquina 1</h1>
swap1@swap1:/var/www$ curl 192.168.1.102
<h1> Soy Maquina 1</h1>
swap1@swap1:/var/www$ curl 192.168.1.102
<h1> Soy Maquina 1</h1>
swap1@swap1:/var/www$ curl 192.168.1.102
<h1> Soy Maquina 1</h1>
swap1@swap1:/var/www$ curl 192.168.1.102
<h1> Soy Maquina 1</h1>
swap1@swap1:/var/www$ curl 192.168.1.102
<h1> Soy Maquina 1</h1>
swap1@swap1:/var/www$ curl 192.168.1.102
<h1> Soy Maquina 1</h1>
swap1@swap1:/var/www$ curl 192.168.1.102
<h1> Soy Maquina 1</h1>
swap1@swap1:/var/www$ curl 192.168.1.102
<h1> Soy Maquina 1</h1>
```

Y por último el ejemplo usando la directiva keepalive, junto con el tiempo de mantenimiento de la conexión:

```
upstream apaches {
        server 192.168.1.100;
        server 192.168.1.101;
        keepalive 3;
}
```

Apache Benchmark

¿Cómo se instala?

```
swap1@swap1:~$ sudo apt-get install apache2-utils
```

Apache Benchmark con NGINX

```
Concurrency Level:      10
Time taken for tests:   0.675 seconds
Complete requests:      1000
Failed requests:        0
Write errors:           0
Total transferred:      297000 bytes
HTML transferred:       24000 bytes
Requests per second:    1482.14 [#/sec] (mean)
Time per request:       6.747 [ms] (mean)
Time per request:       0.675 [ms] (mean, across all concurrent requests)
Transfer rate:          429.88 [Kbytes/sec] received

Connection Times (ms)
              min  mean[+/-sd] median   max
Connect:        0    0   0.2      0       6
Processing:     1    2   4.9      1     102
Waiting:        1    2   4.9      1     102
Total:          1    2   5.0      2     103

Percentage of the requests served within a certain time (ms)
  50%      2
  66%      2
  75%      2
  80%      2
  90%      2
  95%      2
  98%      3
  99%      8
 100%    103 (longest request)
swap1@swap1:~$
```

PARTE CON HAPROXY

Primero tenemos que pausar nginx:

sudo service nginx stop

Luego procedemos a la instalación del haproxy y debemos modificar el archivo ***/etc/haproxy/haproxy.cfg*** ya que la configuración que trae por defecto no nos vale :

sudo apt-get install haproxy

```
global
        daemon
        maxconn 256
defaults
        mode http
        contimeout 4000
        clitimeout 42000
        srvtimeout 43000
frontend http-in
        bind *:80
        default_backend servers
backend servers
        server m1 192.168.1.100:80 maxconn 32
        server m2 192.168.1.101:80 maxconn 32
```

Lanzamos el servicio:

sudo /usr/sbin/haproxy -f /etc/haproxy/haproxy.cfg

Probamos su funcionamiento con curl 192,168,1,102 (ip del balanceador)

```
swap1@swap1:~$ curl 192.168.1.102/index.html
<h1> Soy Maquina 1</h1>
swap1@swap1:~$ curl 192.168.1.102/index.html
<h1> Soy Maquina 2</h1>
swap1@swap1:~$ curl 192.168.1.102/index.html
<h1> Soy Maquina 1</h1>
```

Apache Benchmark con HAPROXY

```
Failed requests:        0
Write errors:           0
Total transferred:      298000 bytes
HTML transferred:       24000 bytes
Requests per second:    2622.58 [#/sec] (mean)
Time per request:       3.813 [ms] (mean)
Time per request:       0.381 [ms] (mean, across all concurrent requests)
Transfer rate:          763.21 [Kbytes/sec] received

Connection Times (ms)
              min  mean[+/-sd] median   max
Connect:        0    1   0.4      0       3
Processing:     1    3   0.6      3       7
Waiting:        1    3   0.6      3       7
Total:          1    4   0.7      4       8
ERROR: The median and mean for the initial connection time are more than twice t
he standard
       deviation apart. These results are NOT reliable.

Percentage of the requests served within a certain time (ms)
  50%      4
  66%      4
  75%      4
  80%      4
  90%      5
  95%      5
  98%      5
  99%      6
 100%      8 (longest request)
swap1@swap1:~$ _
```