SERVIDORES WEB
DE ALTAS PRESTACIONES

José David Torres de las Morenas

Práctica 3

NGINX

Lo primero que haremos es crear una nueva máquina virtual. No debemos instalar LAMP en esta tercera máquina ya que bloquearía el puerto 80.

Vemos si las dos máquinas creadas en las prácticas anteriores son accesibles desde la que acabamos de crear con ping:

```
root@ubuntu:/# ping 192.168.157.128
PING 192.168.157.128 (192.168.157.128) 56(84) bytes of data.
64 bytes from 192.168.157.128: icmp_seq=1 ttl=64 time=0.707 ms
64 bytes from 192.168.157.128: icmp_seq=2 ttl=64 time=1.01 ms
64 bytes from 192.168.157.128: icmp_seq=3 ttl=64 time=1.04 ms
64 bytes from 192.168.157.128: icmp_seq=4 ttl=64 time=0.872 ms
^C
--- 192.168.157.128 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3004ms
rtt min/avg/max/mdev = 0.707/0.909/1.046/0.135 ms
root@ubuntu:/# ping 192.168.157.129
PING 192.168.157.129 (192.168.157.129) 56(84) bytes of data.
64 bytes from 192.168.157.129: icmp_seq=1 ttl=64 time=0.511 ms
64 bytes from 192.168.157.129: icmp_seq=2 ttl=64 time=0.988 ms
64 bytes from 192.168.157.129: icmp_seq=3 ttl=64 time=0.999 ms
64 bytes from 192.168.157.129: icmp_seq=4 ttl=64 time=1.02 ms
^C
--- 192.168.157.129 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3004ms
rtt min/avg/max/mdev = 0.511/0.881/1.029/0.217 ms
root@ubuntu:/# _
```

Procedemos a la instalación de nginx:
Importamos la clave del repositorio para poder instalar nginx:

```
ubuntu@ubuntu:/tmp$ wget http://nginx.org/keys/nginx_signing.key
--2017-05-09 03:31:05--  http://nginx.org/keys/nginx_signing.key
Resolving nginx.org (nginx.org)... 206.251.255.63, 95.211.80.227, 2001:1af8:4060:a004:21::e3, ...
Connecting to nginx.org (nginx.org)|206.251.255.63|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1561 (1.5K) [text/plain]
Saving to: 'nginx_signing.key'

nginx_signing.key       100%[=================================>]   1.52K  --.-KB/s    in 0.005s

2017-05-09 03:31:05 (290 KB/s) - 'nginx_signing.key' saved [1561/1561]

ubuntu@ubuntu:/tmp$ rm -f /tmp/nginx_signing.key
```

Añadimos el repositorio al fichero:

```
root@ubuntu:/tmp# echo "deb http://nginx.org/packages/ubuntu lucid nginx" >> /etc/apt/sources.list
root@ubuntu:/tmp# echo "deb-src http://nginx.org/packages/ubuntu lucid nginx" >> /etc/apt/sources
root@ubuntu:/tmp# _
```

Y ya podemos instalar nginx con la siguiente instrucción :

apt-get install nginx

Ahora tenemos que configurarlo editamos el archivo default.conf con la siguiente instrucción:

nano /etc/nginx/conf.d/default.conf

Y sobreescribimos la configuración por la siguiente:

```
upstream apaches{
        server 192.168.157.128;
        server 192.168.157.129;
}

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 "";
        }
}
```

Reiniciamos el servicio: sudo service nginx restart

Tras esto si el balanceador de carga ha sido bien configurado hacemos curl de la máquina servidora e irá alternando entre el html de la maquina 1 y la máquina 2. Para dar un distinto peso solo debemos añadir a la línea del server el peso:

server 192.168.157.128 weight=2;
server 192.168.157.129 weight=1;

Reiniciamos y comprobamos que todo va bien con curl 192.168.157.134

HAPROXY

Instalamos con la siguiente instrucción:

sudo apt-get install haproxy

Tras la instalación debemos de cambiar la configuración de haproxy con esta instrucción:

sudo nano /etc/haproxy/haproxy.cfg

y establecemos la siguiente configuración:

```
global
        daemon
        maxconn 256

defaults
        mode    http
        timeout connection 4000
        timeout client 42000
        timeout server 43000
frontend http-in
        bind *:80
        default_backend servers
backend servers
        server m1 192.168.157.128:80 maxconn 32
        server m2 192.168.157.129:80 maxconn 32
```

Ya podemos probar la configuración al igual que con nginx, con curl a la dirección de la máquina que tiene instalado haproxy:

```
root@ubuntu:/etc/haproxy# sudo /usr/sbin/haproxy -f /etc/haproxy/haproxy.cfg
root@ubuntu:/etc/haproxy# curl 192.168.157.134
<!DOCTYPE html>
<html>

        <body>
                <p>Maquina 1</p>
        </body>
</html>
root@ubuntu:/etc/haproxy# curl 192.168.157.134
<!DOCTYPE html>
<html>
        <body>
                <p>Maquina 2</p>

        </body>
</html>
root@ubuntu:/etc/haproxy#
```

Para cambiar la carga de trabajo de cada servidor establecemos weight:

```
global
        daemon
        maxconn 256

defaults
        mode    http
        timeout connect 5000
        timeout client  50000
        timeout server  50000

frontend http-in
        bind *:80
        default_backend servers

backend servers
        server m1 192.168.157.128 maxconn 32 weight 20
        server m2 192.168.157.129 maxconn 32 weight 10
```

Sometemos a una alta carga al servidor balanceado con la siguiente instrucción:

ab –n 1000 –c 10 http://192.168.157.134/index.html

que hace mil peticiones concurrentemente de 10 en 10 y obtenemos la siguiente salida (Para utilizar ab debemos instalar la herramienta con la instrucción
sudo apt-get install apache2-utils):

```
Server Hostname:        192.168.157.134
Server Port:            80

Document Path:          /index.html
Document Length:        68 bytes

Concurrency Level:      10
Time taken for tests:   0.429 seconds
Complete requests:      1000
Failed requests:        0
Total transferred:      308000 bytes
HTML transferred:       68000 bytes
Requests per second:    2329.60 [#/sec] (mean)
Time per request:       4.293 [ms] (mean)
Time per request:       0.429 [ms] (mean, across all concurrent requests)
Transfer rate:          700.70 [Kbytes/sec] received

Connection Times (ms)
              min  mean[+/-sd] median   max
Connect:        0    0   1.7      0      23
Processing:     1    4   3.3      3      27
Waiting:        0    4   3.2      3      27
Total:          1    4   3.9      3      33

Percentage of the requests served within a certain time (ms)
  50%      3
  66%      5
  75%      6
  80%      7
  90%      9
  95%     10
  98%     13
  99%     24
 100%     33 (longest request)
root@ubuntu:/etc/haproxy# _
```

PARTE OPCIONAL

POUND

Instalamos pound con la siguiente instrucción:

sudo apt-get install pound

Una vez que acaba la instalación, debemos cambiar la configuración del archivo /etc/pound/pound.cfg :

sudo nano /etc/pound/pound.cfg

y añadimos la siguiente configuración:

```
ListenHTTP
        Address 192.168.157.134
        Port    80
End

Service
        BackEnd
                Address 192.168.157.128
                Port    80
        End
        BackEnd
                Address 192.168.157.129
                Port    80
        End
End
```

Guardamos la configuración de este archivo, y a continuación debemos cambiar el fichero /etc/default/pound y cambiamos el valor de startup a 1:

```
# Defaults for pound initscript
# sourced by /etc/init.d/pound
# installed at /etc/default/pound by the maintainer scripts

# prevent startup with default configuration
# set the below varible to 1 in order to allow pound to start
startup=1
```

Y por último lo probamos al igual que en ngin y haproxy: