

# SERVIDORES WEB DE ALTAS PRESTACIONES

## PRÁCTICA 3: Balanceo de carga en un sitio web

### Autor

Miguel Ángel Pérez Díaz



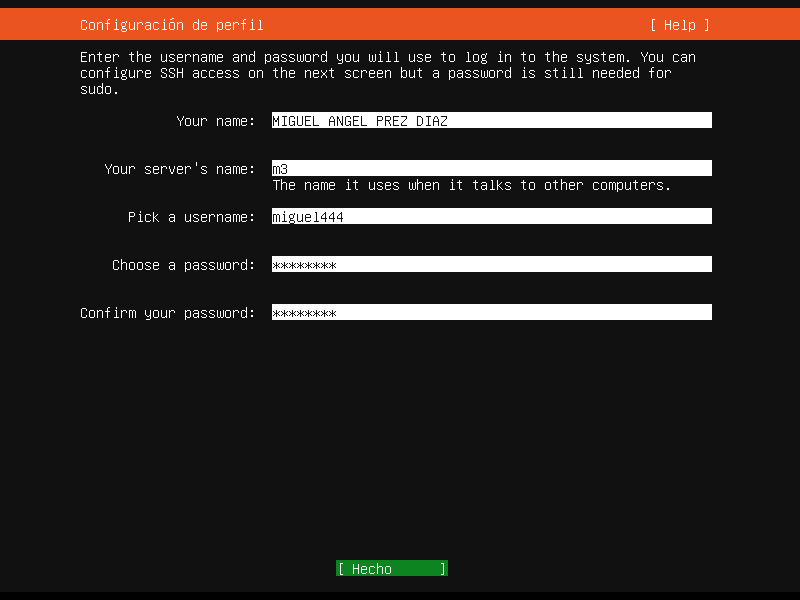
Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación

—

Granada,2020

1. ***Configurar una máquina e instalar el nginx como balanceador de carga.***

Siguiendo el esquema de la práctica propuesta utilizaremos M1 y M2 como granja web, una nueva máquina M3 como balanceador y M4 como máquina que realiza peticiones al balanceador. Inicialmente vamos a definir los elementos utilizados en esta práctica:

* Una nueva máquina virtual M3 con toda la configuración de red ya configurada.
* En mi caso en lugar de crear otra máquina M4 para realizar las peticiones, voy a trabajar con la máquina anfitrión utilizando la bash de Ubuntu desde Windows.
* Las máquinas virtuales M1 y M2 previamente creadas y configuradas en las prácticas anteriores.

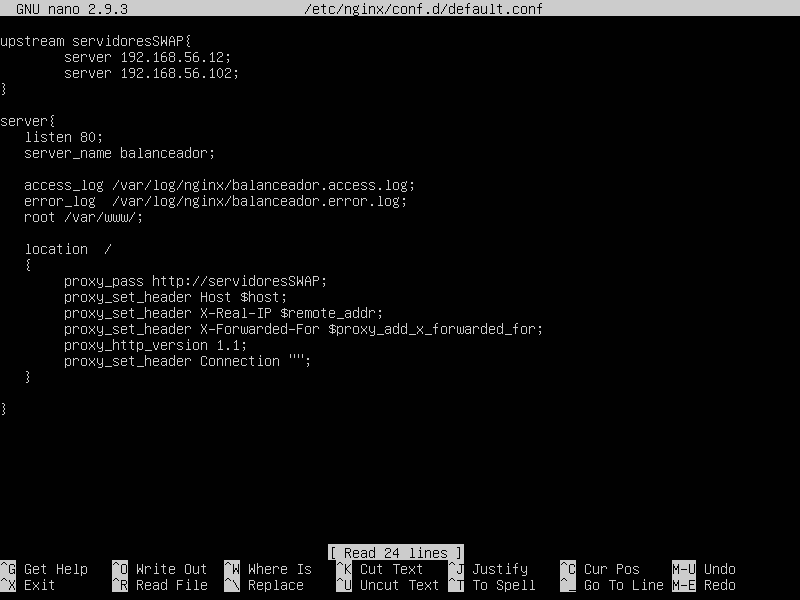
Una vez definidos los principales elementos de nuestra práctica, vamos a tratar de utilizar M3 como balanceador de carga de nuestra granja web, para ello debemos inicialmente instalar el servidor web ***nginx***:

* *sudo apt-get update && sudo apt-get dist-upgrade && sudo apt-get autoremove*
* *sudo apt-get install nginx*
* *sudo systemctl start nginx*

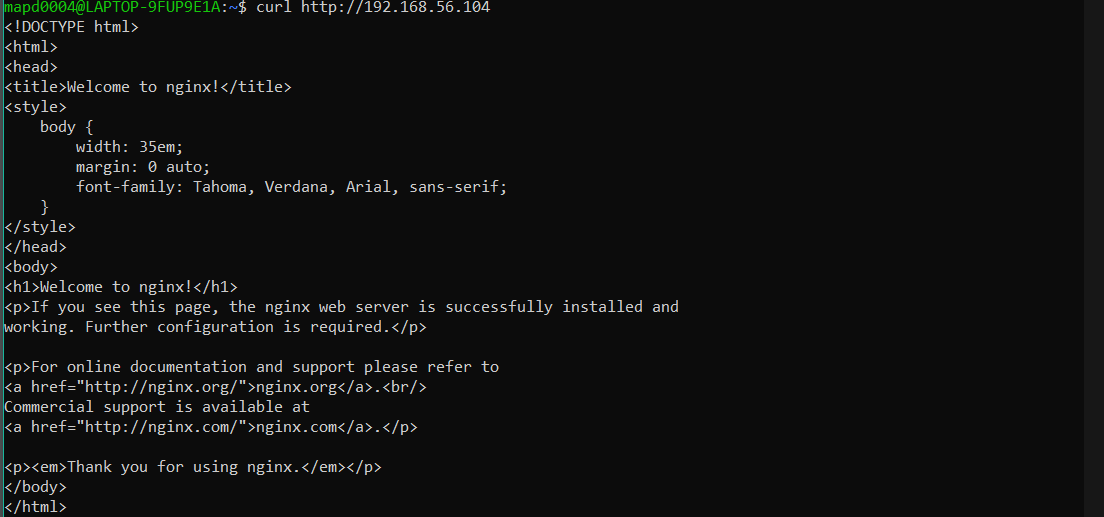
Una vez instalado nginx y activado el servicio, vamos a tratar de configurarlo para poder llevar a cabo el balanceo de la carga. En nuestro caso la configuración la vamos a hacer para definir balanceo por **round robin** (asignación por turnos) y por **ponderaciones o pesos** de los servidores.

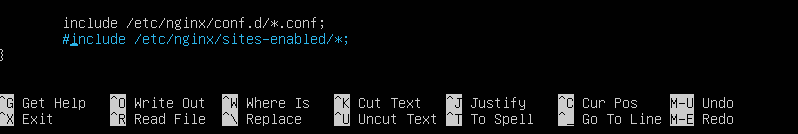
Para configurar nuestro balanceador debemos modificar, o en nuestro caso ha sido necesario crear, el fichero */etc/nginx/conf.d/default.conf.*

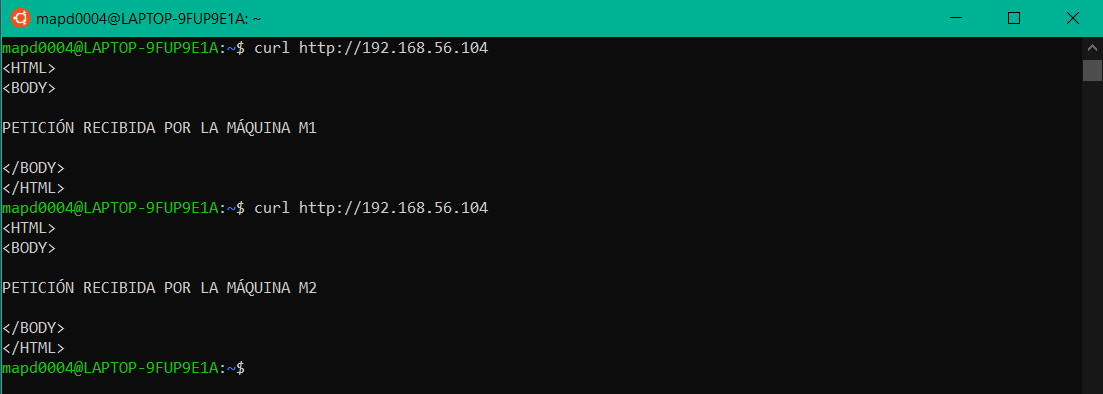
En el fichero indicaremos inicialmente que servidores formarán nuestra granja web, para ello indicaremos sus correspondientes IPs en la sección ‘upstream’ del fichero. Después en la sección ‘server’ indicaremos el puerto de escucha y el nombre del servidor, así como que la conexión entre nginx y los servidores finales sea HTTP 1.1 así como especificarle que debe eliminar la cabecera Connection (hacerla vacía) para evitar que se pase al servidor final la cabecera que indica el usuario.

Finalmente, el fichero quedaría de la siguiente forma:

En la configuración anterior hemos definido el balanceo por round robin en la que todos los servidores de la granja tienen la misma prioridad. Una vez configurado todo vamos a trata de probar nuestro balanceador de carga, para ello reiniciaremos el servicio con: *sudo service nginx restart* (comando que utilizaremos para reiniciar el servicio cada vez que hagamos un cambio en el archivo de configuración).

Una vez reiniciado el servicio, vamos a tratar de usar el comando cURL en nuestra máquina anfitriona M4 indicando la dirección IP de balanceador:

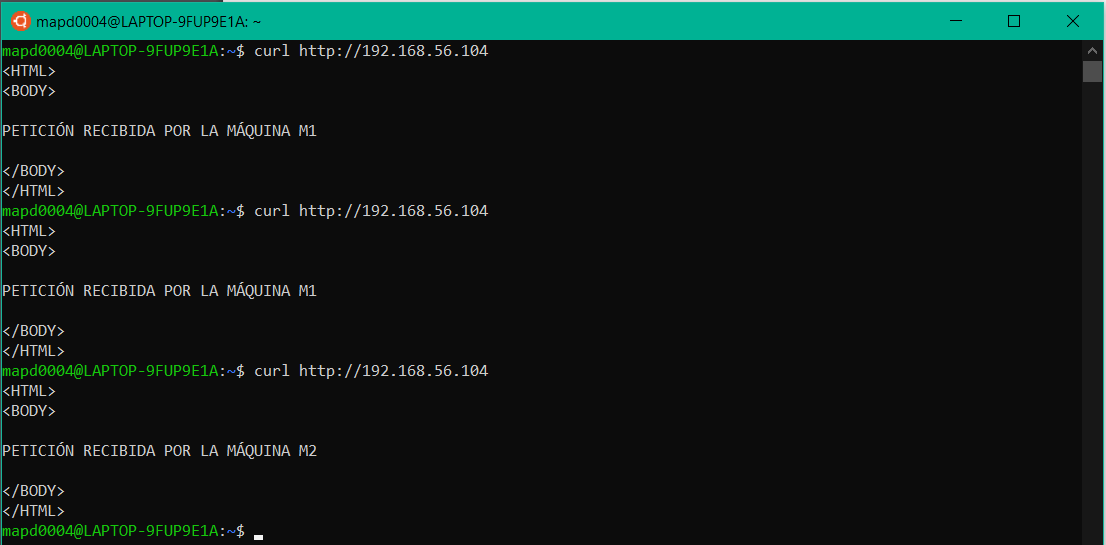
Podemos observar que no obtenemos una respuesta correcta de nuestro balanceador, esto se debe a que tenemos nginx configurado como servidor web y nos está devolviendo el archivo index.html de nginx y no de las máquinas que forman la granja. Para solucionar esto basta con comentar la línea: *include /etc/nginx/sites-enabled/\** del archivo */etc/nginx/nginx.conf.*

Una vez modificado el fichero, volvemos a reiniciar el servicio y volvemos a probar el comando cURL hacia la máquina balanceadora, y podemos ver como en este caso si se obtiene la respuesta esperada:

Como segunda parte de esta sección debemos realizar un balanceo por ponderaciones o pesos, para ello bastaría con modificar ligeramente el archivo de configuración creado anteriormente, especificando los pesos para cada uno de los servidores finales, dando en este caso el doble de capacidad a la máquina M1 sobre M2:

Podemos ver como se han definido los pesos con la variable ‘weight’.

Una vez realizado este cambio, volvemos a reiniciar el servicio y realizamos cURL desde la máquina anfitriona al balanceador:



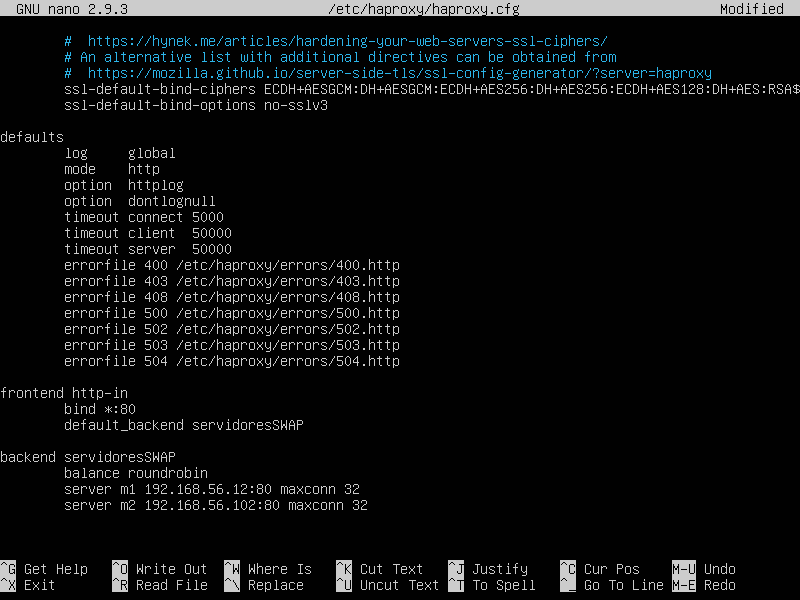
Podemos observar cómo se le da más prioridad a la máquina M1 recibiendo 2 peticiones de cada 3 enviadas al balanceador.

1. ***Configurar una máquina e instalar el haproxy como balanceador de carga.***

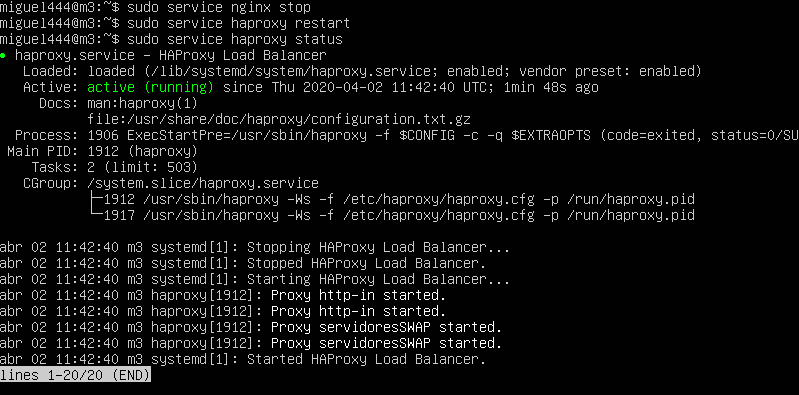
En esta nueva sección vamos a realizar las mismas tareas que en la sección anterior. Para ello inicialmente debemos instalar el balanceador haproxy usando el comando:

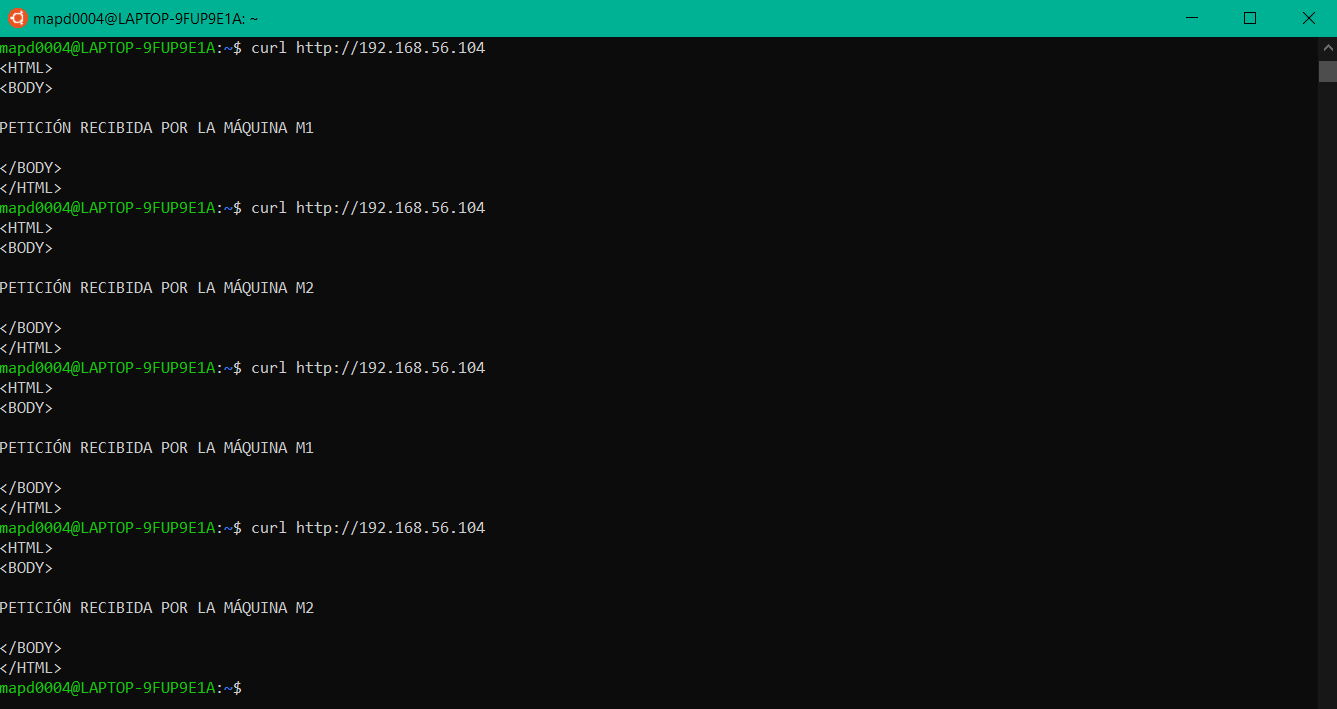
* *******Sudo apt-get install haproxy*

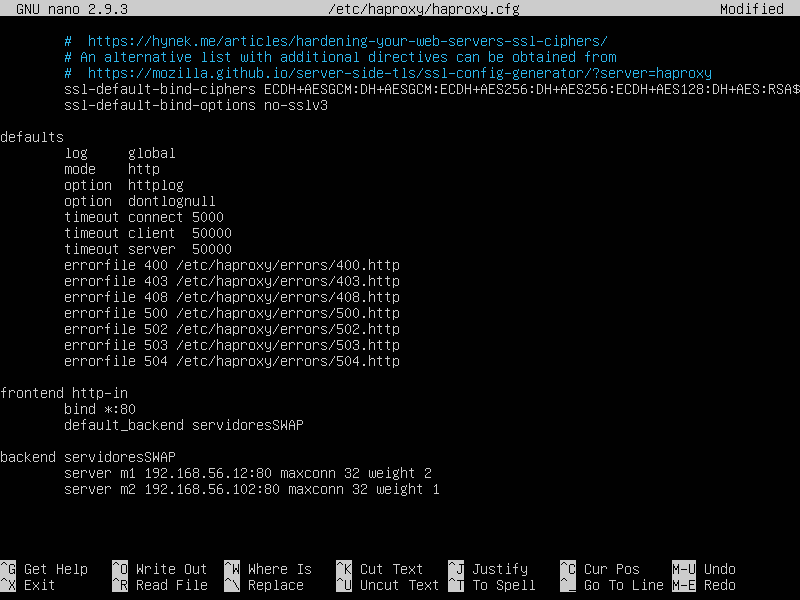
Una vez instalado el balanceador, debemos configurarlo para poder llevar a cabo la tarea. En este caso debemos modificar el fichero */etc/haproxy/haproxy.cfg* para indicarle cuales son nuestros balanceadores y que peticiones balancear.

Vamos a indicar que las peticiones las escuche por el puerto 80 y que las redirija a la dirección ip de la máquina correspondiente. Finalmente, el fichero de configuración quedaría de la siguiente manera:

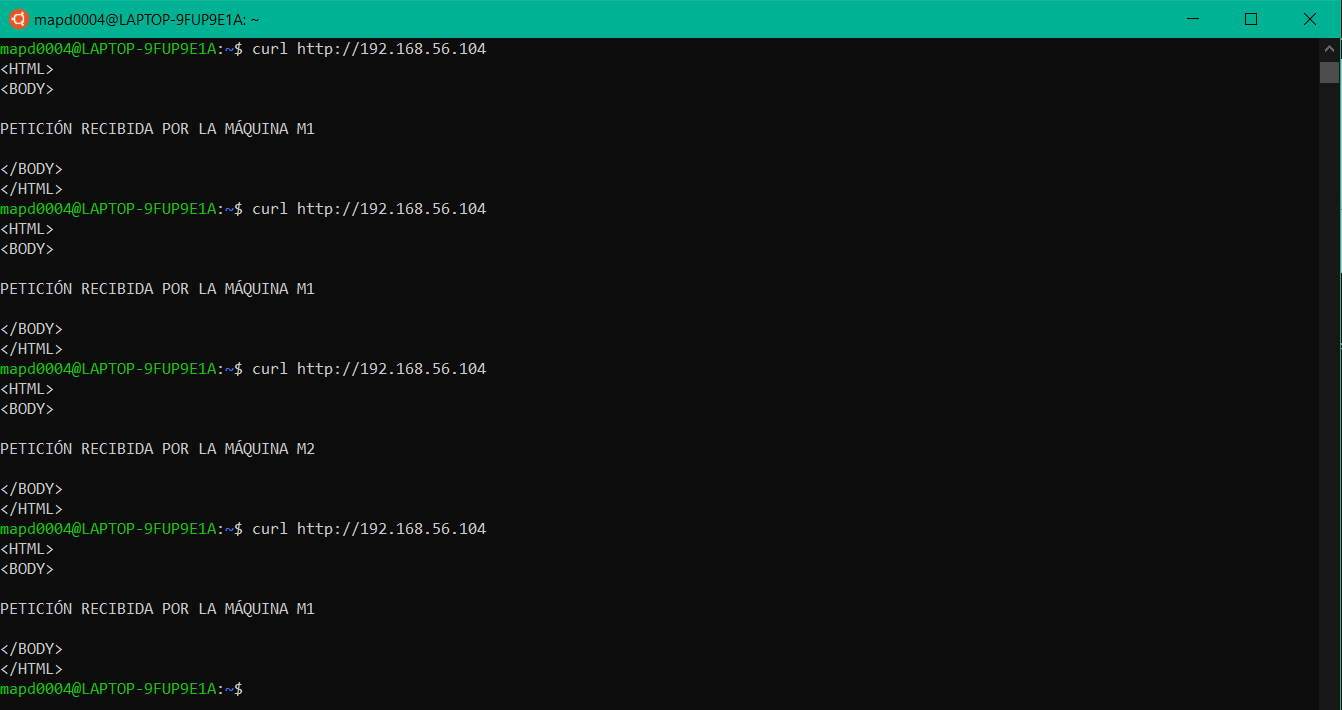
Al mismo tiempo se le ha indicado que realice un balanceo por round robin con la variable ‘balance roundrobin’ en la sección backend.

Antes de probar el funcionamiento del balanceador haproxy, se debe pausar el balanceador nginx instalado anteriormente ya que ambos escuchan por el puerto 80 y los dos no pueden estar funcionando. Y después reiniciaremos el servicio de haproxy para que empiece a funcionar:

Todo parece correcto y en funcionamiento por lo que procederemos a realizar las peticiones con el comando cURL desde la máquina anfitriona, obteniéndose así la siguiente respuesta:

Tras esto, vamos a configurar el balanceador para que realice el balanceo por ponderaciones o pesos, y al igual que antes bastará con una ligera modificación en el archivo de configuración indicando que la máquina M1 tenga el doble de capacidad que M2:

De la misma manera se ha utilizado la variable ‘weight’ para establecer los pesos de los diferentes servidores finales. Tras esto nuevamente debemos reiniciar el servicio.

Ahora veremos cómo al realizar varias peticiones el balanceador le da más prioridad a la máquina M1:

Una vez instalados, configurados y probados tanto ***nginx*** como ***haproxy***, vamos a proceder a realizar una comparación entre tiempos de servicio utilizando ambos utilizando la herramienta Apache Benchmark para simulaciones múltiples peticiones a nuestro balanceador.

1. ***Someter a la granja web a una alta carga, generada con la herramienta Apache Benchmark, teniendo primero nginx y después haproxy.***

Inicialmente en mi máquina anfitriona se ha tenido que instalar la herramienta, para ello se ha utilizado el comando: *sudo apt-get install apache2-utils.*

Una vez instalado vamos a realizar una simulación de prueba sobre el balanceador:

*ab -n 10000 -c 10* [*http://192.168.56.104/index.html*](http://192.168.56.104/index.html)

Realizando 10000 peticiones de manera concurrente con 10 hebras, por lo que se harán de 10 en 10 sobre la ip de la máquina balanceadora.

Para la comparación se ha utilizado la técnica de round robin para ambos balanceadores, obtenido así los siguientes resultados:

Para 10000 peticiones:

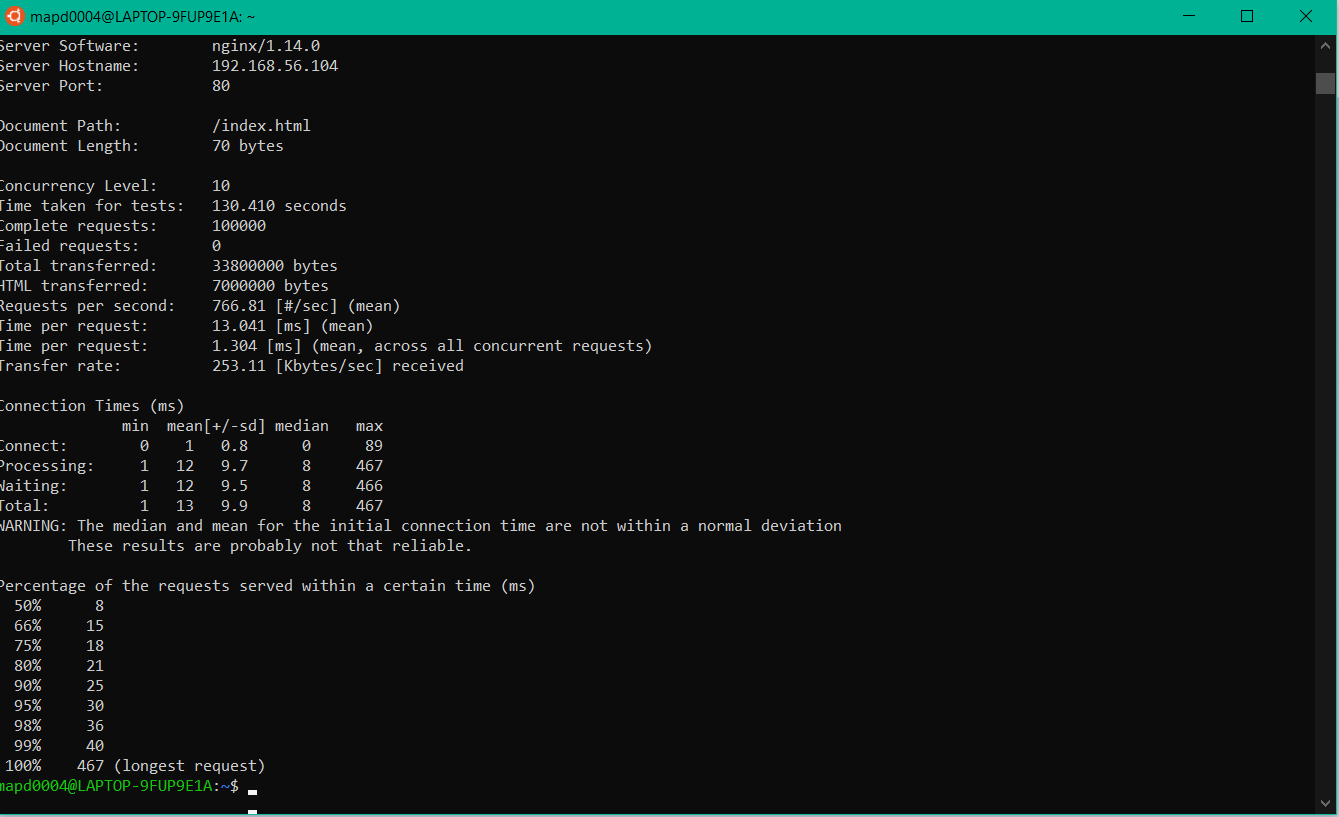
|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| TIPO | TIME(s) | REQ/S | TIME/REQ(ms) | TRANFER RATE(KB/s) | CONNECT(ms) | PROCESSING(ms) | WAITING(ms) | TOTAL(ms) |
| NGINX | 9.868 | 1013.38 | 9.868 | 334.49 | 0 | 9 | 9 | 10 |
| HAPROXY | 10.616 | 941.97 | 10.616 | 311.84 | 1 | 10 | 10 | 10 |

Para 100000 peticiones:

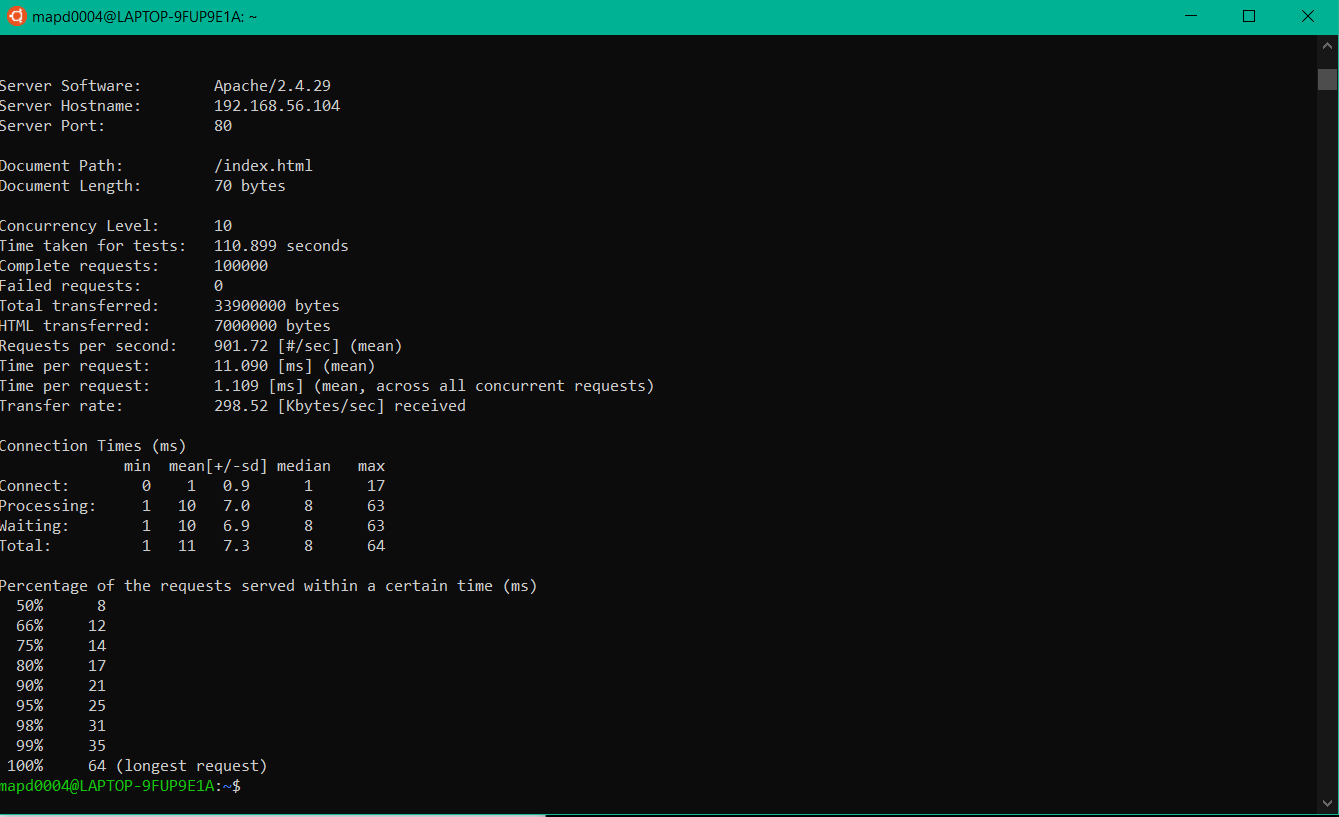
|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| TIPO | TIME(s) | REQ/S | TIME/REQ(ms) | TRANFER RATE(KB/s) | CONNECT(ms) | PROCESSING(ms) | WAITING(ms) | TOTAL(ms) |
| NGINX | 130.410 | 766.81 | 13.041 | 253.11 | 1 | 12 | 12 | 13 |
| HAPROXY | 110.899 | 901.72 | 11.090 | 298.52 | 1 | 10 | 10 | 11 |

Se puede observar como cuando el número de peticiones aumenta considerablemente se obtiene una respuesta más rápida por parte del balanceador ***haproxy*** como así indican los tiempos obtenidos.

Capturas obtenidas con la ejecución para cada balanceador:



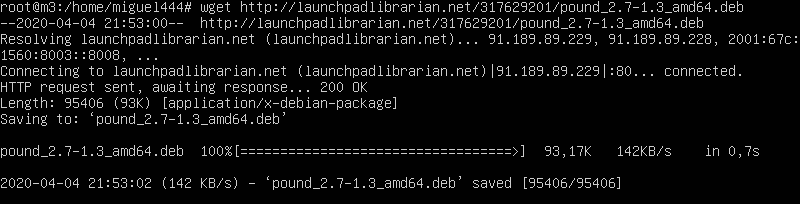
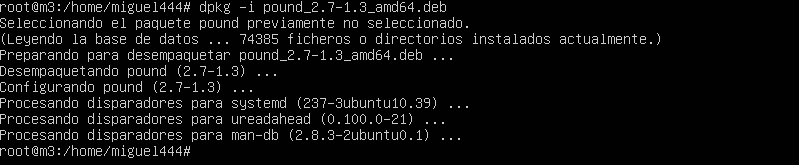
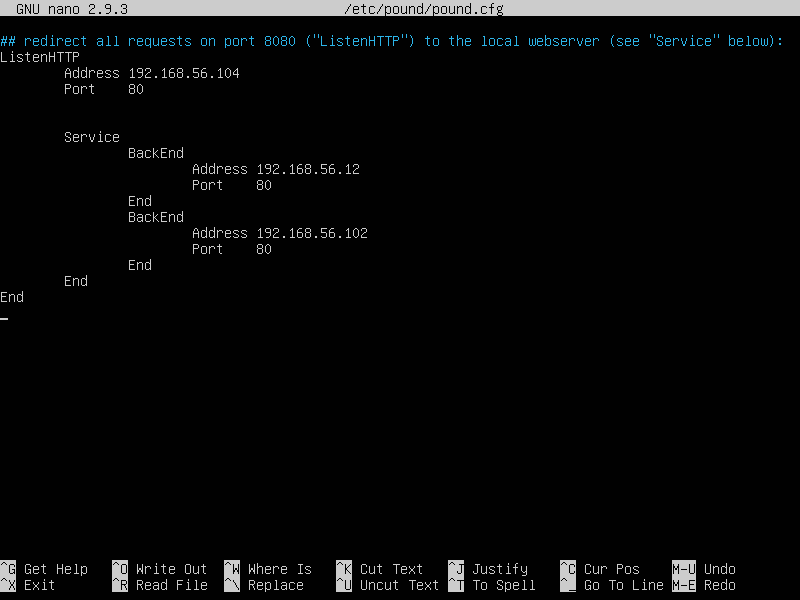
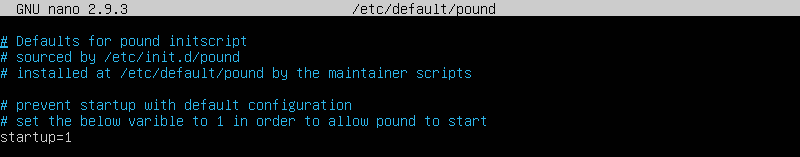
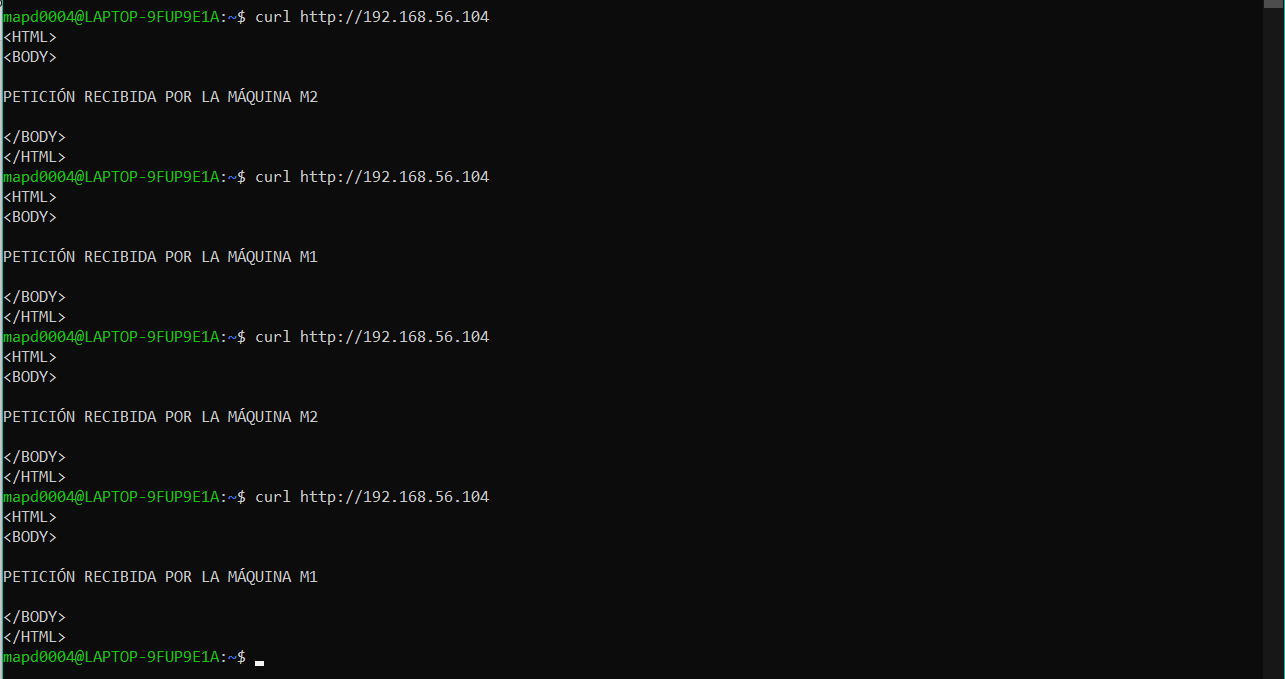
***NGNIX***

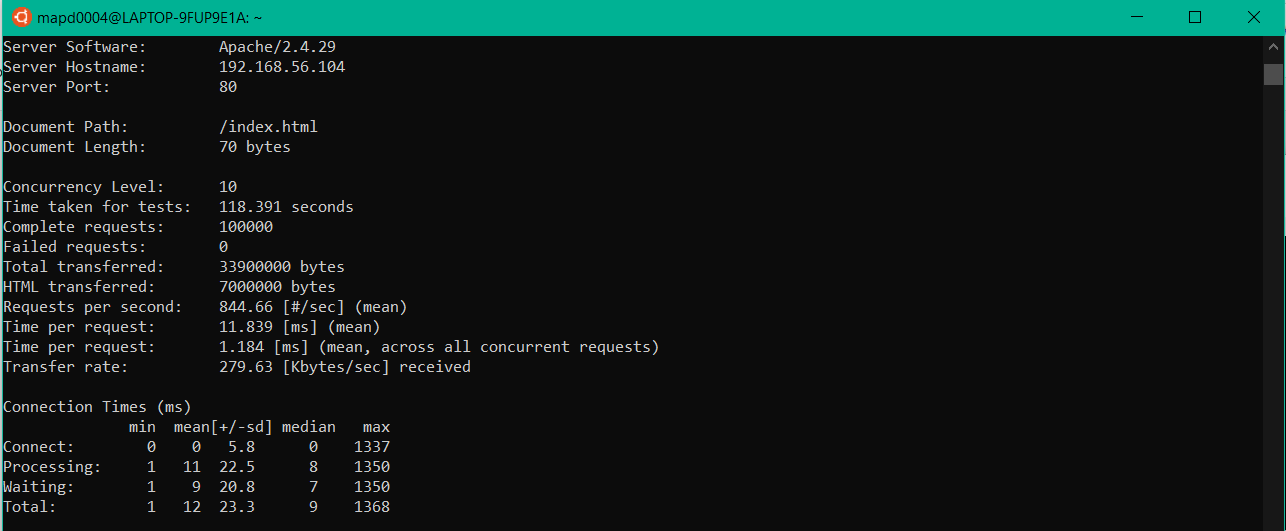


**HAPROXY**

* ***Se propone el uso de algún otro software de balanceo diferente a los dos explicados en este guion (por ejemplo, Pound)***

En este caso no se ha podido instalar el balanceador **Pound** de la misma forma que los demás con un simple *apt-get install*, sino que ha sido necesario descargar e instalar el balanceador a mano. Para la instalación y configuración se han seguido los siguientes pasos:

* Se ha descargado el fichero .deb correspondiente al balanceador con el comando: *wget* [*http://launchpadlibrarian.net/317629201/pound\_2.7-1.3\_amd64.deb*](http://launchpadlibrarian.net/317629201/pound_2.7-1.3_amd64.deb)
* Una vez tenemos descargado el paquete, vamos a tratar de instalarlo en nuestra máquina virtual. Para ello ejecutamos el siguiente comando: *dpkg -i pound\_2.7-1.3\_amd64.deb*
* Una vez finalizada la instalación del balanceador, vamos a proceder a configurarlo para poder redirigir las peticiones a las máquinas M1 y M2 que forman nuestra granja web. Para ello debemos modificar el fichero */etc/pound/pound.cfg* y añadir lo siguiente:
  + Address: IP de la máquina balanceadora
  + Creamos dos Backend para cada servidor final y especificamos sus correspondientes direcciones IP y el puerto de escucha.
* Una vez instalado y configurado el balanceador vamos a reiniciar el servicio, pero antes debemos modificar el fichero */etc/default/pound* y poner la variable ‘startup’ con valor 1 para que el servicio se ponga en marcha:
* Finalmente reiniciamos el servicio con *service pound restart y ya* podemos ver como nuestro balanceador distribuye las peticiones a los servidores correctamente (cabe destacar que previamente se han parado los servicios de nginx y haproxy para evitar problemas).

Para terminar, vamos a probar el desempeño del balanceador con la herramienta de Apache Benchmark y así obtener unas medidas de rendimiento al igual que se ha hecho con los balanceadores explicados anteriores en la práctica.