
Servidores Web de Altas Prestaciones.

Práctica 3

Balanceo de carga en un sitio web.

Ricardo Ruiz Fernández de Alba

25/05/2023



Índice

Introducción	2
Descripción de las tareas	2
Tarea 1. Balanceo de carga con NGINX y HAProxy.	3
Balanceo de carga con NGINX	3
Instalación de NGINX.	3
Configuración de NGINX como balanceador de carga	4
Ejemplo de funcionamiento	6
Repartir carga en función de pesos	7
Balanceo de carga con HAProxy	11
Tarea 2. Alta carga con Apache Benchmark	11
Tarea 3. Análisis Comparativo	11
Referencias	11

Introducción

En esta práctica, el objetivo es configurar las máquinas virtuales de forma que dos hagan de servidores web finales mientras que la tercera haga de balanceador de carga por software.

Descripción de las tareas

En esta práctica se llevarán a cabo las **tareas básicas**:

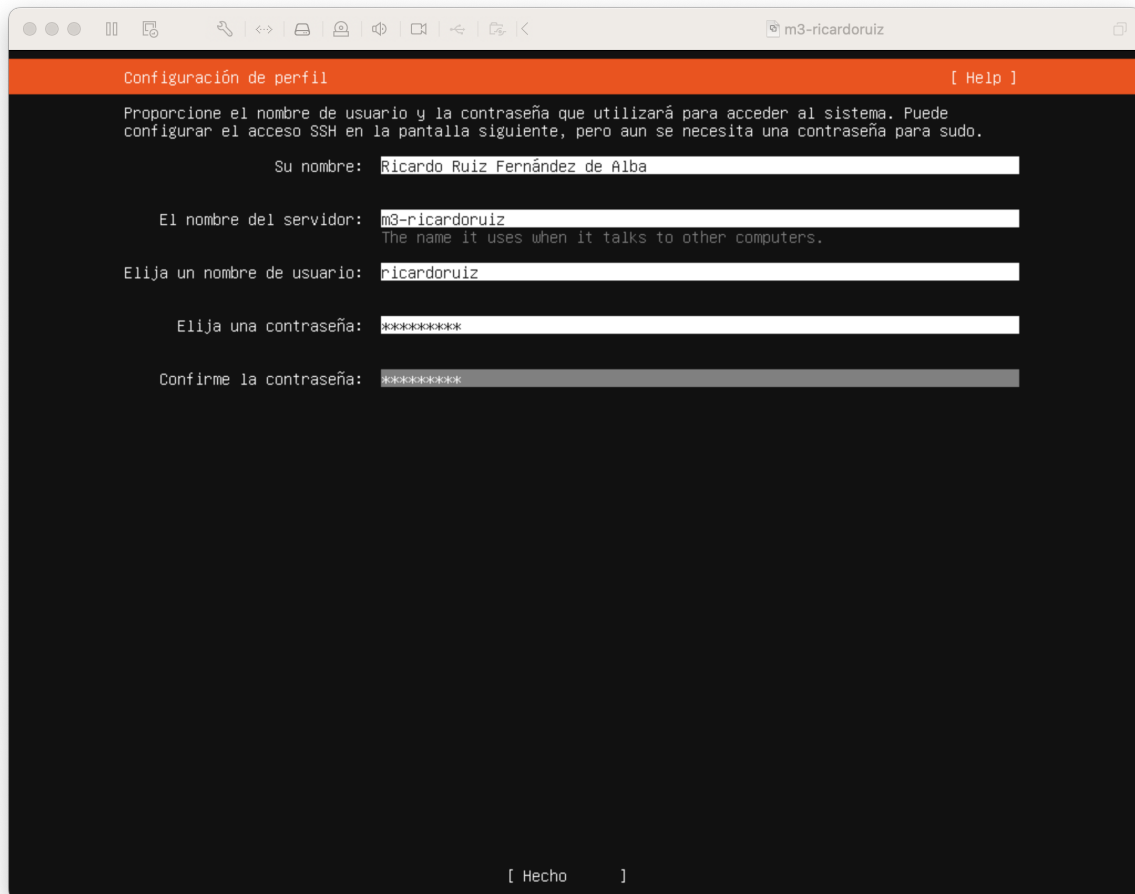
1. Configurar una máquina e instalar nginx y haproxy como balanceadores de carga con el algoritmo round-robin
2. Someter la granja web a una alta carga con la herramienta Apache Benchmark a través de M3, considerando 2 opciones:
 - a) nginx con round-robin
 - b) haproxy con round-robin
3. Realizar un análisis comparativo de los resultados considerando el número de peticiones por unidad de tiempo

Como **opciones avanzadas**:

1. Configurar nginx y haproxy como balanceadores de carga con ponderación, suponiendo que M1 tiene el doble de capacidad que M2.
2. Habilitar el módulo de estadísticas en HAproxy con varias opciones y analizarlo.
3. Instalar y configurar otros balanceadores de carga (Gobetween, Zevenet, Pound, etc.).
4. Someter la granja web a una alta carga con la herramienta Apache Benchmark considerando los distintos balanceadores instalados y configurados.
5. Realizar un análisis comparativo de los resultados considerando el número de peticiones por unidad de tiempo

Tarea 1. Balanceo de carga con NGINX y HAProxy.

Creemos una nueva máquina virtual llamada m3-ricardoruiz con Ubuntu Server 22.04 LTS, a la que añadiremos el usuario ricardoruiz con contraseña Swap12324.



Balanceo de carga con NGINX

Instalación de NGINX.

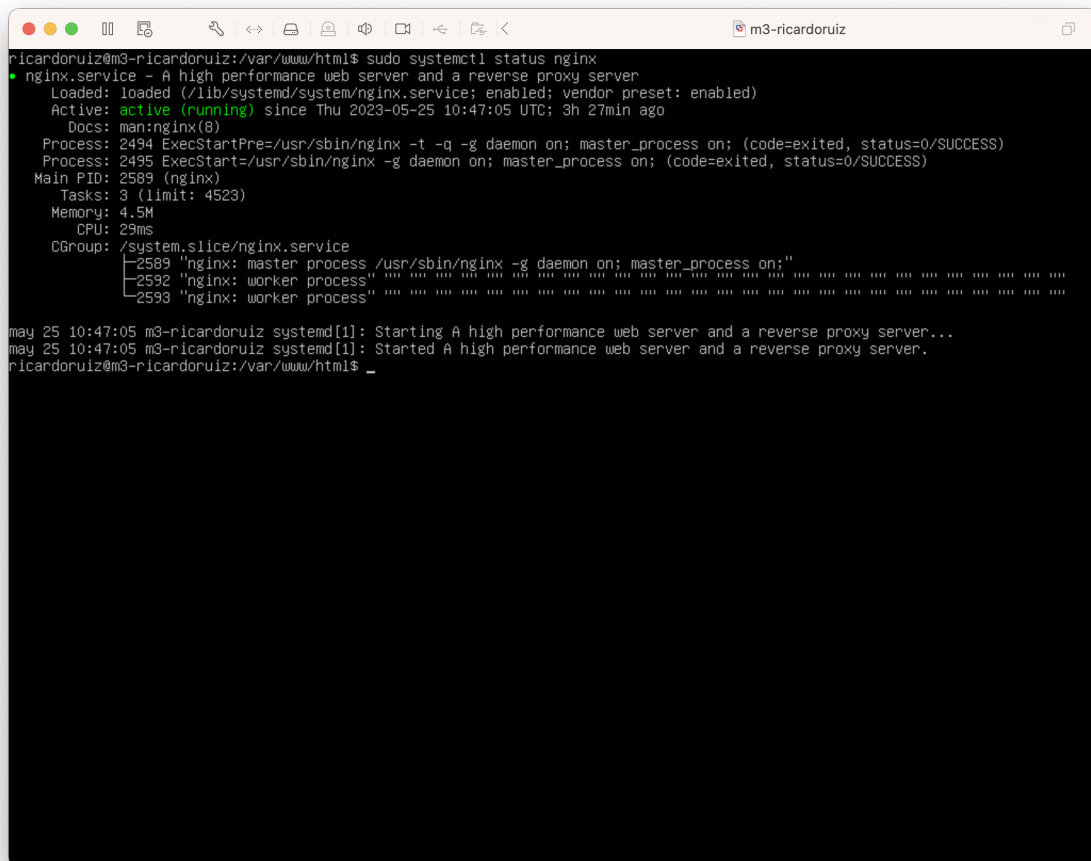
Seguiremos la guía de instalación de nginx para Ubuntu Server 22.04 de Digital Ocean.

```
1 ricardoruiz@m3-ricardoruiz $ sudo apt update
2 ricardoruiz@m3-ricardoruiz $ sudo apt install nginx
```

Antes de probar Nginx, es necesario configurar el firewall para permitir el acceso al servicio. Nginx se registra como un servicio en ufw durante la instalación, lo que facilita permitir el acceso a Nginx.

```
1 ricardoruiz@m3-ricardoruiz $ sudo ufw allow 'Nginx HTTP'
```

Comprobamos que nginx está activo con `sudo systemctl status nginx`:

A terminal window titled 'm3-ricardoruiz' showing the command 'sudo systemctl status nginx' and its output. The output indicates that the nginx.service is loaded and active (running) since Thursday, 2023-05-25 at 10:47:05 UTC. It shows the main PID as 2589 and lists three tasks: the master process and two worker processes. The terminal also shows log messages from systemd indicating the successful start of the service.

```
ricardoruiz@m3-ricardoruiz:/var/www/html$ sudo systemctl status nginx
• nginx.service - A high performance web server and a reverse proxy server
   Loaded: loaded (/lib/systemd/system/nginx.service; enabled; vendor preset: enabled)
   Active: active (running) since Thu 2023-05-25 10:47:05 UTC; 3h 27min ago
     Docs: man:nginx(8)
   Process: 2494 ExecStartPre=/usr/sbin/nginx -t -q -g daemon on; master_process on; (code=exited, status=0/SUCCESS)
   Process: 2495 ExecStart=/usr/sbin/nginx -g daemon on; master_process on; (code=exited, status=0/SUCCESS)
   Main PID: 2589 (nginx)
    Tasks: 3 (limit: 4523)
   Memory: 4.5M
      CPU: 29ms
   CGroup: /system.slice/nginx.service
           └─2589 "nginx: master process /usr/sbin/nginx -g daemon on; master_process on;"
             └─2592 "nginx: worker process"
               └─2593 "nginx: worker process"

may 25 10:47:05 m3-ricardoruiz systemd[1]: Starting A high performance web server and a reverse proxy server...
may 25 10:47:05 m3-ricardoruiz systemd[1]: Started A high performance web server and a reverse proxy server.
ricardoruiz@m3-ricardoruiz:/var/www/html$ _
```

Figura 1: Nginx

Configuración de NGINX como balanceador de carga

Debemos deshabilitar la configuración por defecto de nginx como servidor web para que actúe como balanceador.

Para ello, comentamos la línea

```
1 #include /etc/nginx/sites-enabled/*;
```

del fichero de configuración `/etc/nginx/nginx.conf`.

Creemos una nueva configuración en `/etc/nginx/conf.d/default.conf`

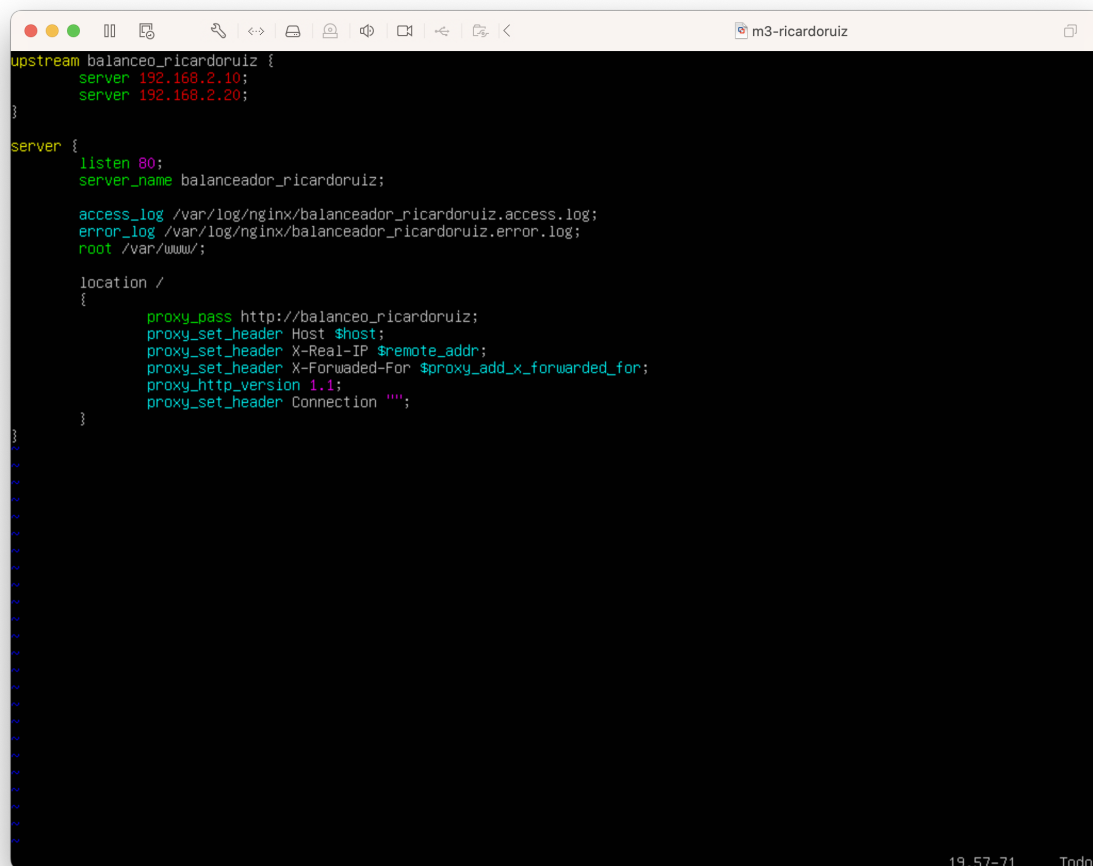
Para definir la granja web de servidores apache escribimos la sección upstream con la IP de las M1 y M2. Es importante que este al principio del archivo de configuración, fuera de la sección server.

```
1 upstream balanceo_ricardoruiz {
2     server 192.168.2.10;
3     server 192.168.2.20;
4 }
```

Debemos definir ahora la sección server para indicar a nginx que use el grupo definido anteriormente en upstream. Para que el proxy_pass funcione correctamente , debemos indicar una conexión de tipo HTTP 1.1 asi como eliminar la cabecera `Connection` para evitar que se pase al servidor final la cabecer que indica el usuario.

```
1 [..]
2 server {
3     listen 80;
4     server_name balanceador_ricardoruiz;
5     access_log /var/log/nginx/balanceador_ricardoruiz.access.log;
6     error_log /var/log/nginx/balanceador_ricardoruiz.error.log;
7     root /var/www/;
8     location / {
9         proxy_pass http://balanceo_ricardoruiz;
10        proxy_set_header Host $host;
11        proxy_set_header X-Real-IP $remote_addr;
12        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
13        proxy_http_version 1.1;
14        proxy_set_header Connection "";
15    }
16 }
```

Luego la configuración completa quedaría como sigue:



```
upstream balanceador_ricardoruiz {
    server 192.168.2.10;
    server 192.168.2.20;
}

server {
    listen 80;
    server_name balanceador_ricardoruiz;

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

    location /
    {
        proxy_pass http://balanceador_ricardoruiz;
        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 "";
    }
}
```

Ejemplo de funcionamiento

Configuramos la IP de la máquina m3-ricardoruiz como IP estática en el fichero `/etc/netplan/00-installer-config.yaml`, añadiendo un nuevo adaptador de red Host-Only.

```
1  network:
2    version: 2
3    renderer: networkd
4    ethernets:
5      ens160:
6        dhcp4: true
7        addresses:
8          - 192.168.1.30/24
9        routes:
10         - to: 0.0.0.0/0
11           via: 192.168.1.1
12         metric: 100
```

```
13     nameservers:  
14         addresses: [8.8.8.8, 8.8.4.4]  
15     ens256:  
16         dhpc4: false  
17         addresses:  
18             - 192.168.2.30/24
```

La IP accesible desde fuera a M3 es 172.16.21.133.

Podemos comprobar el funcionamiento del balanceador con

```
1 ricardoruiz@m3-ricardoruiz $ curl 172.16.21.133/swap.html
```

A terminal window titled 'eigenric — zsh — 80x24' on a macOS background. The prompt is 'eigenric@macbook ~ %'. The user has entered the command 'curl 172.16.21.133/swap.html'. The output is an HTML document with a body containing the text: 'Web de ejemplo de "ricardoruiz" para SWAP', 'Email: ricardoruiz@correo.ugr.es'. The prompt returns to 'eigenric@macbook ~ %'.

```
eigenric@macbook ~ % curl 172.16.21.133/swap.html  
<HTML>  
  <BODY>  
    Web de ejemplo de "ricardoruiz" para SWAP  
    Email: ricardoruiz@correo.ugr.es  
  </BODY>  
</HTML>  
eigenric@macbook ~ %
```

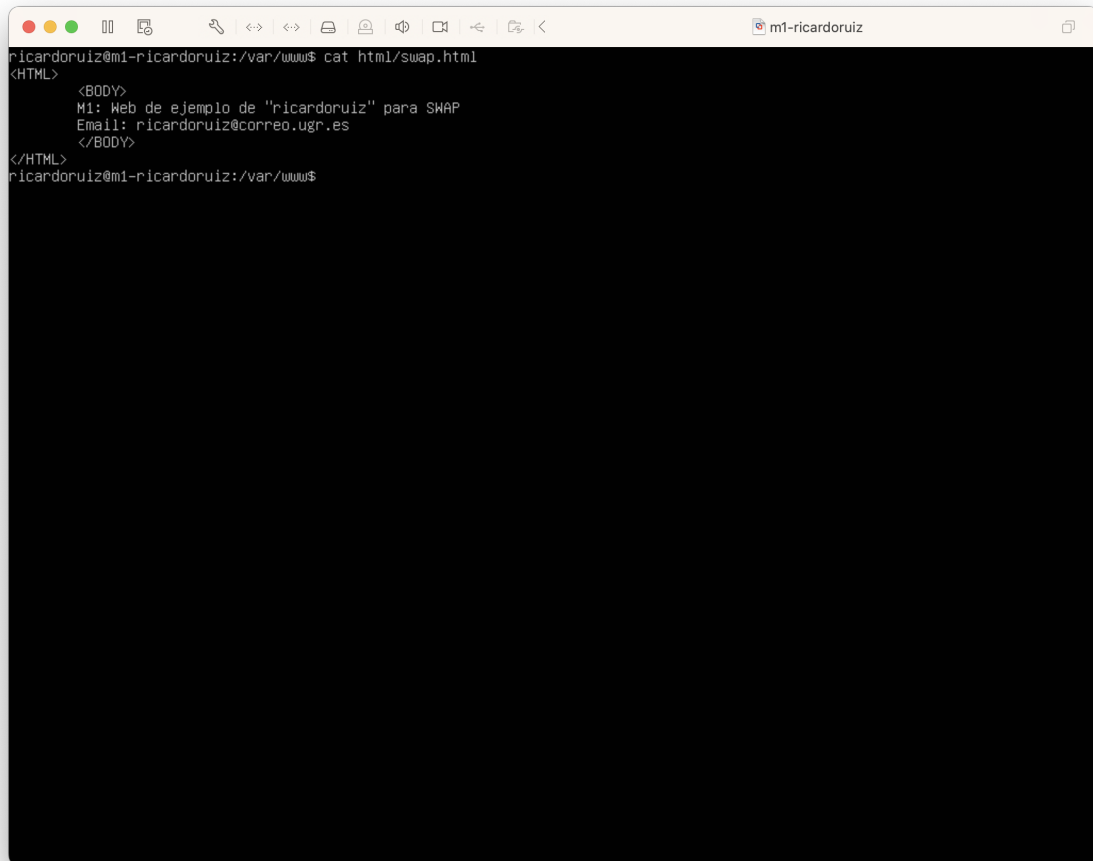
Repartir carga en función de pesos

En caso de saber que alguna de las máquinas finales es más potente, podemos modificar la definición del “upstream” para pasarle más tráfico que al resto. Para ello, asignamos un valor numero al modificador “weight”.

Por ejemplo, podemos hacer que cada tres peticiones que lleguen al balanceador, la máquina M2 atenderá dos y la máquina M1 atenderá una:

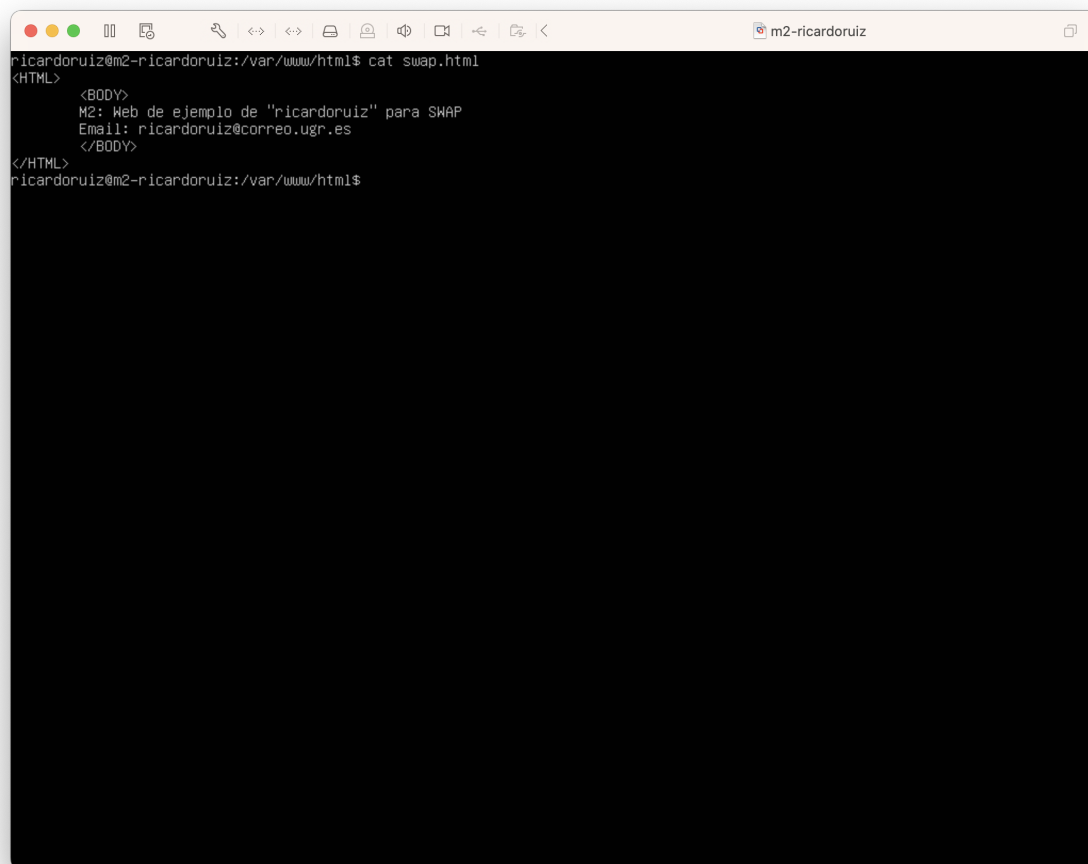

```
1 upstream balanceo_ricardoruiz {  
2     server 192.168.2.10 weight=1;  
3     server 192.168.2.20 weight=2;  
4 }
```

Para comprobarlo, modificamos `swap.html` las máquinas finales para identificarlas.

A terminal window titled 'm1-ricardoruiz' showing the command 'cat html/swap.html' and its output. The output is an HTML snippet with a body containing information about a web example for SWAP.

```
ricardoruiz@m1-ricardoruiz:/var/www$ cat html/swap.html  
<HTML>  
  <BODY>  
    M1: Web de ejemplo de "ricardoruiz" para SWAP  
    Email: ricardoruiz@correo.ugr.es  
  </BODY>  
</HTML>  
ricardoruiz@m1-ricardoruiz:/var/www$
```

Figura 2: swap.html en M1

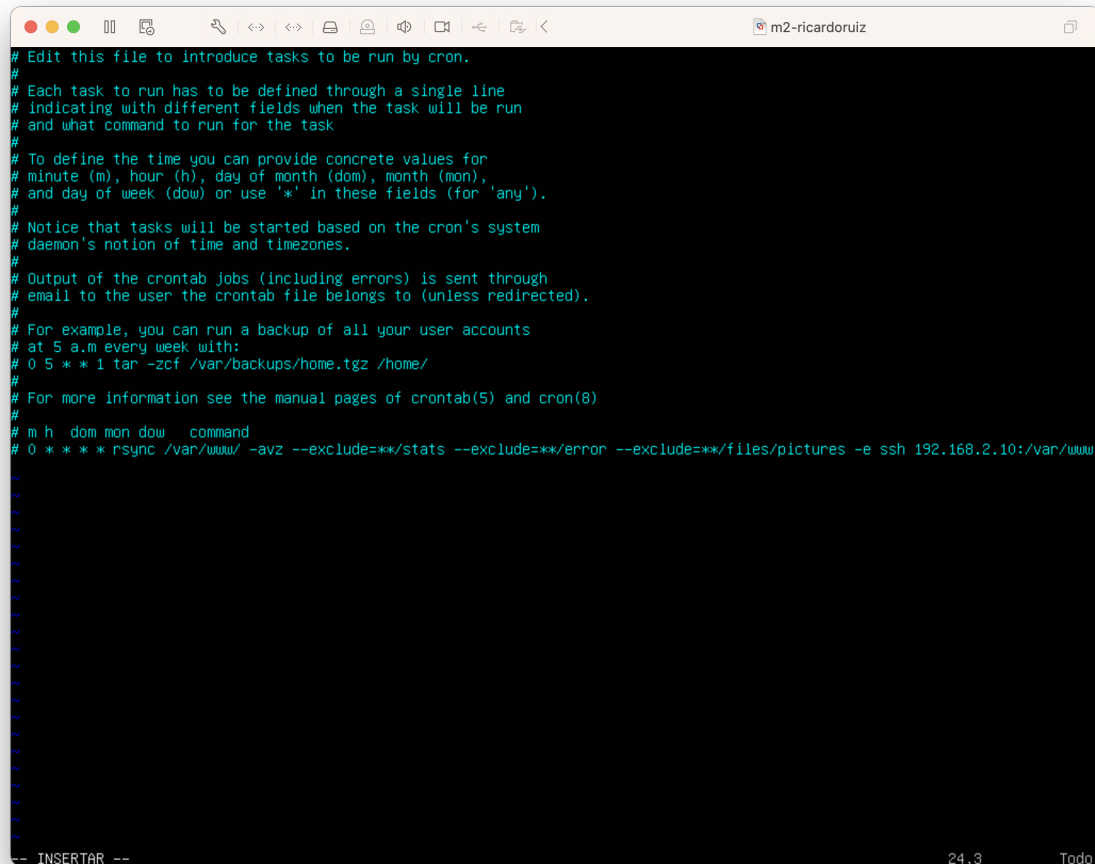
A terminal window titled 'm2-ricardoruiz' with a dark background. The prompt is 'ricardoruiz@m2-ricardoruiz:/var/www/html\$'. The command 'cat swap.html' has been executed, displaying the following HTML content:

```
<HTML>
  <BODY>
    M2: Web de ejemplo de "ricardoruiz" para SWAP
    Email: ricardoruiz@correo.ugr.es
  </BODY>
</HTML>
```

The prompt returns to 'ricardoruiz@m2-ricardoruiz:/var/www/html\$'.

Figura 3: swap.html en M2

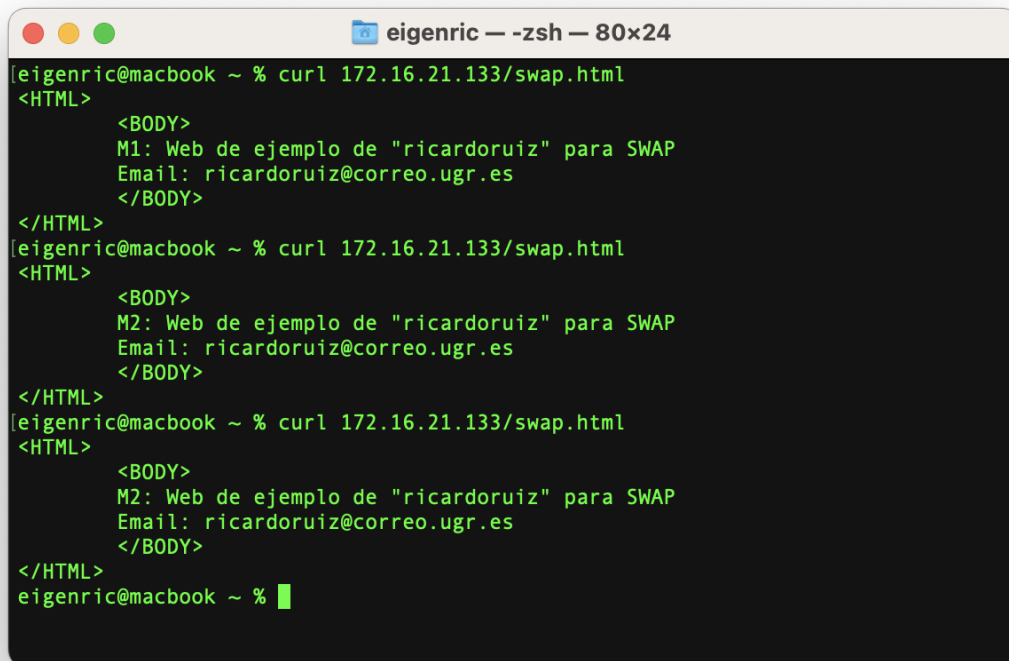
Desactivamos también la tarea cron de sincronización con rsync para evitar que se sobrescriban los cambios.



```
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h dom mon dow   command
# 0 * * * * rsync /var/www/ -avz --exclude=**/stats --exclude=**/error --exclude=**/files/pictures -e ssh 192.168.2.10:/var/www
-- INSERTAR --
```

Figura 4: Desactivación de la tarea cron

Realizamos tres peticiones y comprobamos que se sigue el **Algoritmo Round Robin**, acabando dos de ellas en M2:



```
eigenric — zsh — 80x24
[eigenric@macbook ~ % curl 172.16.21.133/swap.html
<HTML>
    <BODY>
        M1: Web de ejemplo de "ricardoruiz" para SWAP
        Email: ricardoruiz@correo.ugr.es
    </BODY>
</HTML>
[eigenric@macbook ~ % curl 172.16.21.133/swap.html
<HTML>
    <BODY>
        M2: Web de ejemplo de "ricardoruiz" para SWAP
        Email: ricardoruiz@correo.ugr.es
    </BODY>
</HTML>
[eigenric@macbook ~ % curl 172.16.21.133/swap.html
<HTML>
    <BODY>
        M2: Web de ejemplo de "ricardoruiz" para SWAP
        Email: ricardoruiz@correo.ugr.es
    </BODY>
</HTML>
eigenric@macbook ~ %
```

Balanceo de carga con HAProxy

Tarea 2. Alta carga con Apache Benchmark

Tarea 3. Análisis Comparativo

Referencias