

Examen 29/05/2019

Jesús Rodríguez Heras

29 de mayo de 2019

1. Configuración inicial

Para crear el fichero vagrant usamos el siguiente comando:

```
vagrant init debian/jessie64
```

A continuación añadimos las cuatro máquinas virtuales con sus correspondientes IPs:

- VM1: 192.168.2.11
- VM2: 192.168.2.12
- VM3: 192.168.2.13
- VM4: 192.168.2.14

El vagrantfile quedaría de la siguiente forma:

```
1 Vagrant.configure("2") do |config|
2   config.vm.box = "debian/jessie64"
3
4   config.vm.define :vm1 do |vm1|
5     vm1.vm.box="debian/jessie64"
6     vm1.vm.hostname="VM1"
7     vm1.vm.network "private_network", ip: "192.168.2.11"
8   end
9
10  config.vm.define :vm2 do |vm2|
11    vm2.vm.box="debian/jessie64"
12    vm2.vm.hostname="VM2"
13    vm2.vm.network "private_network", ip: "192.168.2.12"
14  end
15
16  config.vm.define :vm3 do |vm3|
17    vm3.vm.box="debian/jessie64"
18    vm3.vm.hostname="VM3"
19    vm3.vm.network "private_network", ip: "192.168.2.13"
20  end
21
22  config.vm.define :vm4 do |vm4|
23    vm4.vm.box="debian/jessie64"
24    vm4.vm.hostname="VM4"
25    vm4.vm.network "private_network", ip: "192.168.2.14"
26  end
27 end
```

2. Cortafuegos

Para configurar el cortafuegos para SSH (puerto 22), HTTP (puerto 80) y HTTPS (puerto 443) usaremos las siguientes reglas de IPTables en este orden:

1. `sudo iptables -A INPUT -p tcp -dport 22 -j ACCEPT`
2. `sudo iptables -A INPUT -p tcp -dport 80 -j ACCEPT`
3. `sudo iptables -A INPUT -p tcp -dport 443 -j ACCEPT`
4. `sudo iptables -A INPUT -m state --state ESTABLISHED -j ACCEPT`
5. `sudo iptables -A INPUT -j DROP`

Usamos dichas reglas para permitir las conexiones desde los puertos 22, 80 y 443 y denegar el resto (política por defecto).

Luego, para comprobar que los puertos están abiertos, nos dirigiremos a otra máquina virtual, por ejemplo, vm3, e instalaremos curl con:

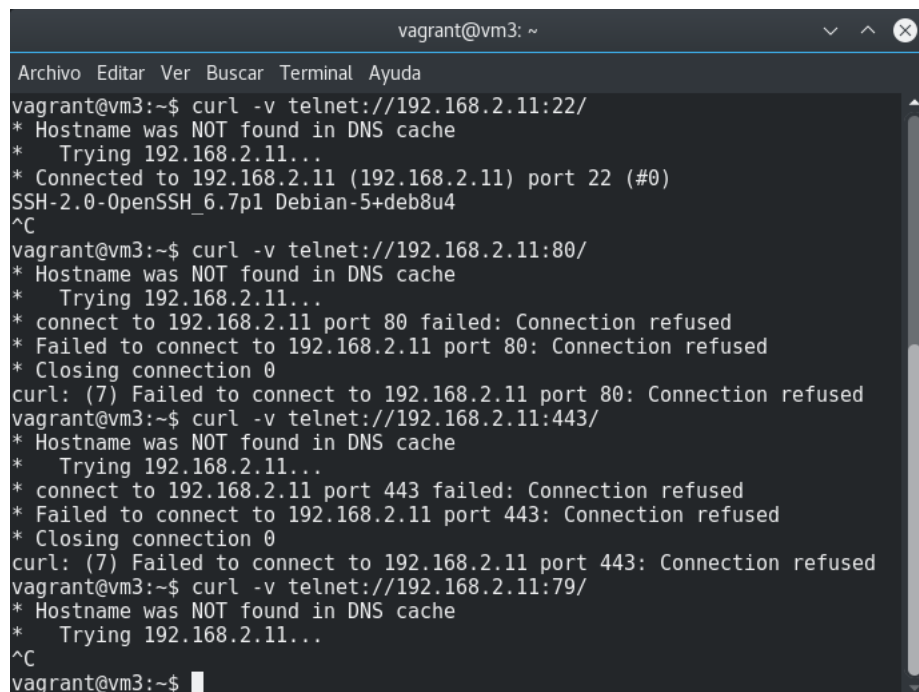
```
sudo apt-get install curl
```

Y comprobaremos que los puertos están abiertos con:

```
curl -v telnet://192.168.2.11:22/  
curl -v telnet://192.168.2.11:80/  
curl -v telnet://192.168.2.11:443/  
curl -v telnet://192.168.2.11:79/
```

También comprobamos el puerto 79 para ver la diferencia, ya que el 79 ha de estar cerrado. Por lo que se quedará esperando una respuesta del servidor que nunca llegará, al revés que con los otros puertos que llegamos al servidor pero no hay ningún servicio en esos puertos.

En esta imagen tenemos el resultado del curl:



```
vagrant@vm3: ~  
Archivo Editar Ver Buscar Terminal Ayuda  
vagrant@vm3:~$ curl -v telnet://192.168.2.11:22/  
* Hostname was NOT found in DNS cache  
*   Trying 192.168.2.11...  
* Connected to 192.168.2.11 (192.168.2.11) port 22 (#0)  
SSH-2.0-OpenSSH_6.7p1 Debian-5+deb8u4  
^C  
vagrant@vm3:~$ curl -v telnet://192.168.2.11:80/  
* Hostname was NOT found in DNS cache  
*   Trying 192.168.2.11...  
* connect to 192.168.2.11 port 80 failed: Connection refused  
* Failed to connect to 192.168.2.11 port 80: Connection refused  
* Closing connection 0  
curl: (7) Failed to connect to 192.168.2.11 port 80: Connection refused  
vagrant@vm3:~$ curl -v telnet://192.168.2.11:443/  
* Hostname was NOT found in DNS cache  
*   Trying 192.168.2.11...  
* connect to 192.168.2.11 port 443 failed: Connection refused  
* Failed to connect to 192.168.2.11 port 443: Connection refused  
* Closing connection 0  
curl: (7) Failed to connect to 192.168.2.11 port 443: Connection refused  
vagrant@vm3:~$ curl -v telnet://192.168.2.11:79/  
* Hostname was NOT found in DNS cache  
*   Trying 192.168.2.11...  
^C  
vagrant@vm3:~$
```

Figura 1: Resultados del curl.

3. Apache

3.1. Instalación de Apache

Instalamos Apache en el servidor principal con:

```
sudo apt-get update
sudo apt-get install apache2
```

3.2. Instalación de PHP

Para instalar PHP usaremos el siguiente comando:

```
sudo apt-get install php5-common libapache2-mod-php5
```

3.3. Instalación de Webmin

Para instalar Webmin tendremos que acceder a la página de descargas de webmin para Debian:
<http://www.webmin.com/deb.html>.

Nos dirigimos al apartado *Using the webmin APT repository*. En el archivo `/etc/apt/sources.list` de la máquina virtual, pegamos lo siguiente:

```
deb https://download.webmin.com/download/repository sarge contrib
```

A continuación, instalamos `apt-transport-https`, actualizamos los paquetes e instalamos Webmin:

```
sudo apt-get install apt-transport-https
sudo apt-get update
sudo apt-get install webmin
```

3.4. Crear los virtualhost

Para el acceso a webmin y al servidor, previamente he configurado el fichero `/etc/hosts` de mi ordenador para que la dirección 192.168.2.11 (el servidor inicial) tenga el alias `www.jrh.ai`.

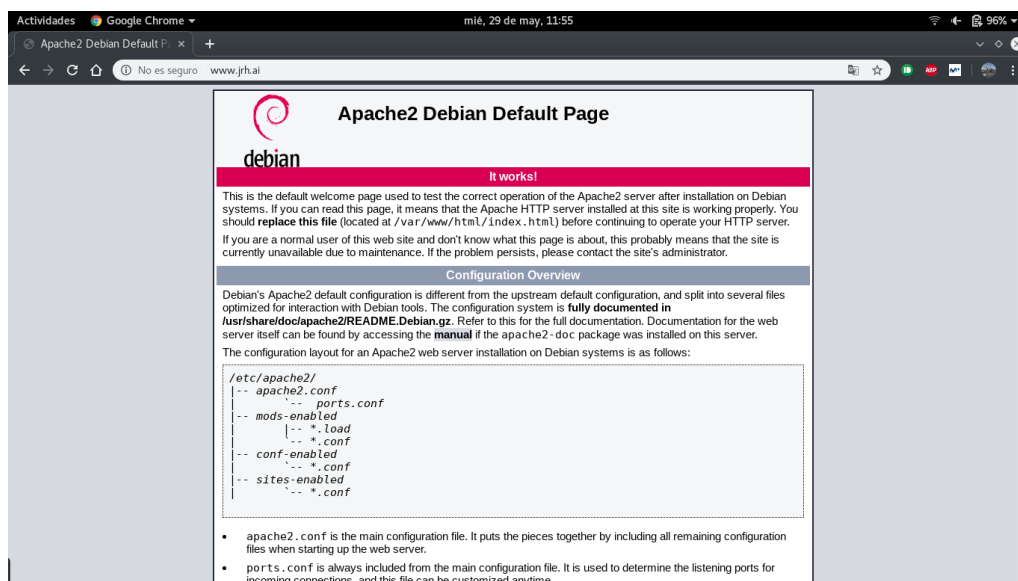


Figura 2: Página principal de apache.

Para entrar en webmin iremos a la dirección `https://jrh.ai:10000` y entraremos con usuario `vagrant` y contraseña `vagrant`.

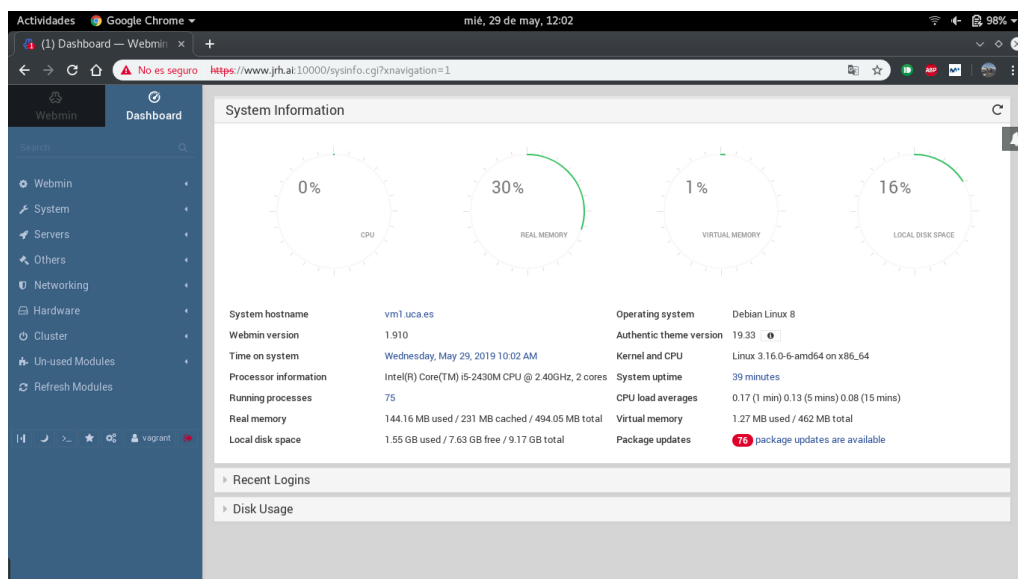


Figura 3: Página principal de Webmin.

Ahora tenemos que configurar los virtualhosts. Para ello, tenemos que crear el directorio `jrh` dentro de `/var/www/` y, dentro de `/var/www/jrh/` creamos los directorios `test` y `admin`.

Luego, vamos a la pestaña “Apache webserver” de webmin y le damos a “Create virtual host”. Creamos los hosts virtuales con los “Document root” y “Server name” y nos debería quedar algo tal que así:

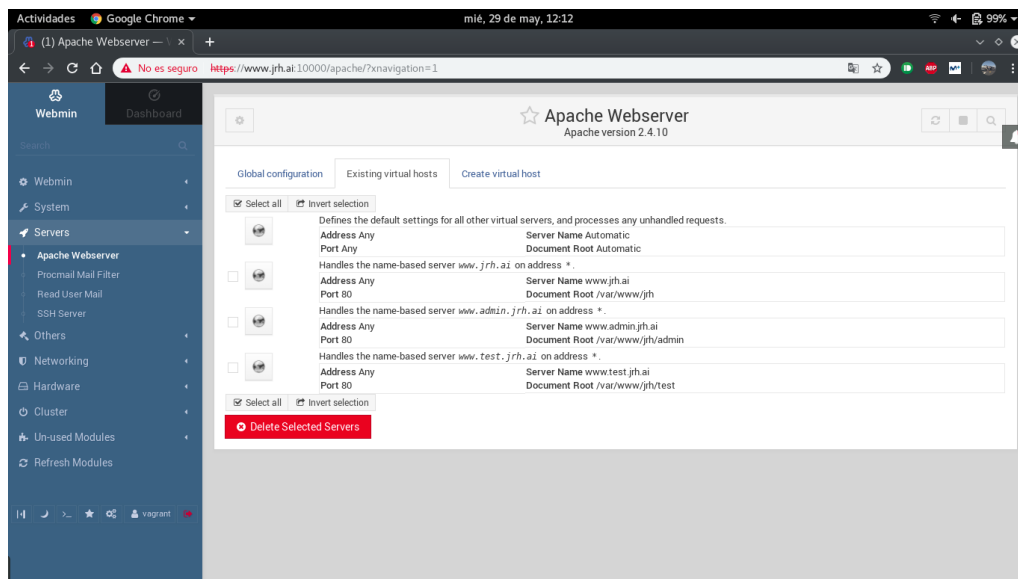


Figura 4: Configuración de virtual hosts.

En el virtual host principal `www.jrh.ai` crearemos el directorio `src` que podrá ser indexado. Para ello, creamos el directorio `/var/www/jrh/src` y, luego, entramos en el virtual host para añadir el directorio. Introducimos el path del directorio y le damos a “Create”.

A continuación, nos dirigimos en webmin a ese directorio y en “Document options” marcamos la casilla de “Selected below” y la de “Generate directory indexes” y guardamos.

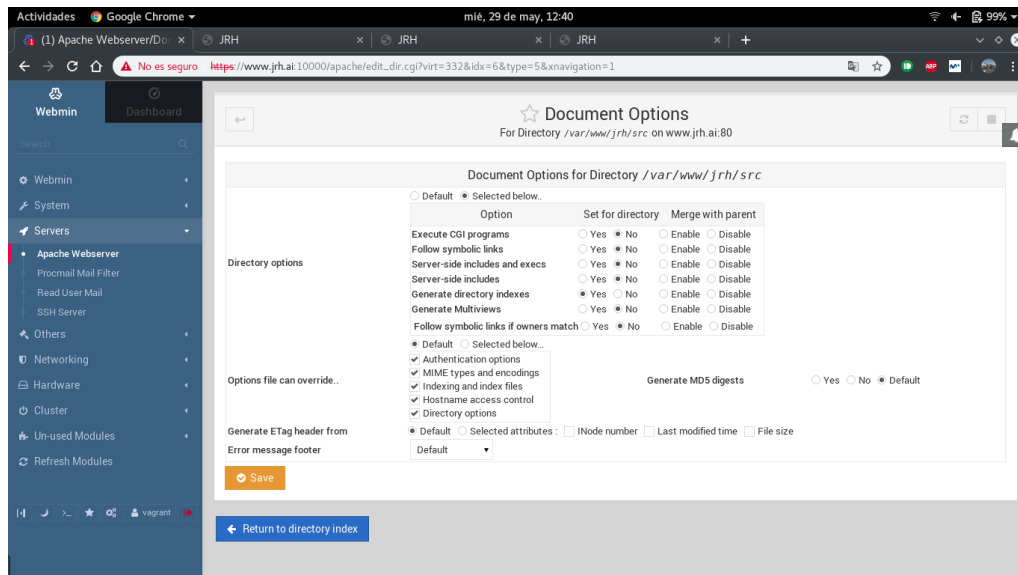


Figura 5: Directorio src indexado.

Para comprobar que funciona creamos algunos archivos en el directorio `/var/www/jrh/src/` y entramos mediante el navegador como en la siguiente imagen:



Figura 6: Directorio src.

3.5. Crear una pequeña página web en cada uno de los virtualhsts con tildes y demás caracteres especiales

Para que podamos escribir tildes y caracteres especiales nos dirigimos a “Global configuration” y luego a “Edit config files” (en Webmin) y añadimos al final:

```
adddefaultcharset utf-8
```

Para ver que funcionan las tildes, crearemos el fichero `index.html` dentro del directorio `/var/www/jrh/` que contendrá el siguiente código:

```
1 <html>
2 <head>
3   <title>JRH</title>
4 </head>
5 <body>
6   <p>Página de inicio de JRH.</p>
7 </body>
8 </html>
```

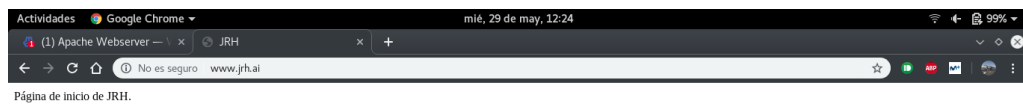


Figura 7: Página de inicio de JRH.

Los directorios `admin` y `test` tendrán también un código similar en su `index.html` y lo podremos ver accediendo a ellos como en la siguiente imagen:

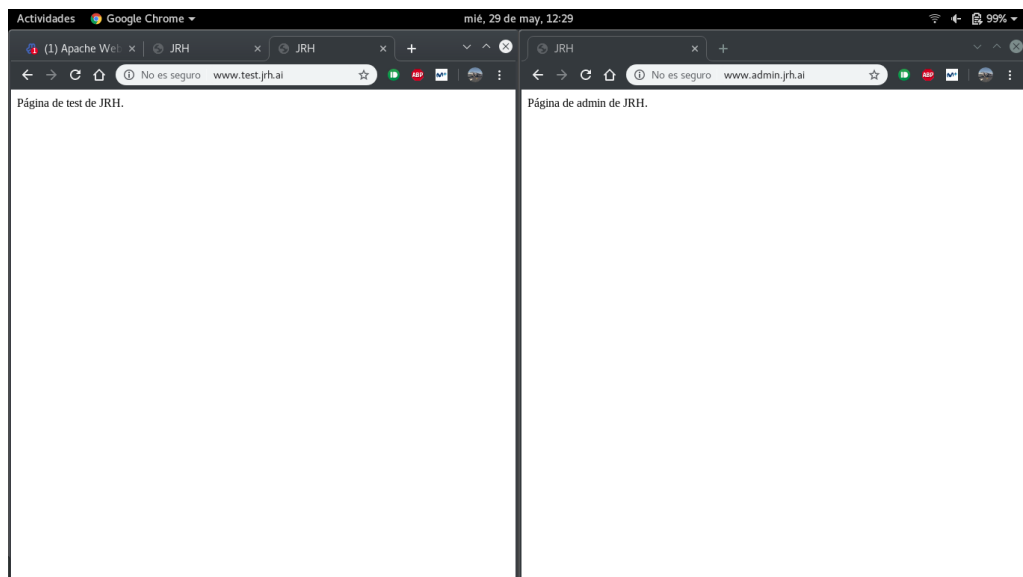


Figura 8: Subdominios test y admin.

3.6. Establecer la redirección

Ahora creamos el fichero `prodList.php` en el directorio `/var/www/jrh/` que contendrá el siguiente código en PHP:

```

1  <?php
2  echo 'Página del producto ' . htmlspecialchars($_GET["id"]);
3  ?>

```


El resultado habitual sería el siguiente (accediendo a `www.jrh.ai/prodList.php?id=002`):



Figura 9: prodList sin redirección.

Ahora debemos hacer la redirección para que se pueda usar un enlace más amigable.

Para ello, habilitamos el módulo “rewrite” en “Configure apache modules” de “Global configuration”.

Luego, añadimos la siguiente línea en el “Edit config files” de “Global configuration”:

```
RewriteRule "^catalogo/(.*)" "/prodList.php?id=$1"
```

Tiene un acento circunflejo delante pero se me pone arriba de la barra en latex (no se por qué).

Para ver el resultado con la redirección entramos en `www.jrh.ai/catalogo/002`:



Figura 10: Catalogo con redirección.

Ahora expondré los ficheros de directivas:

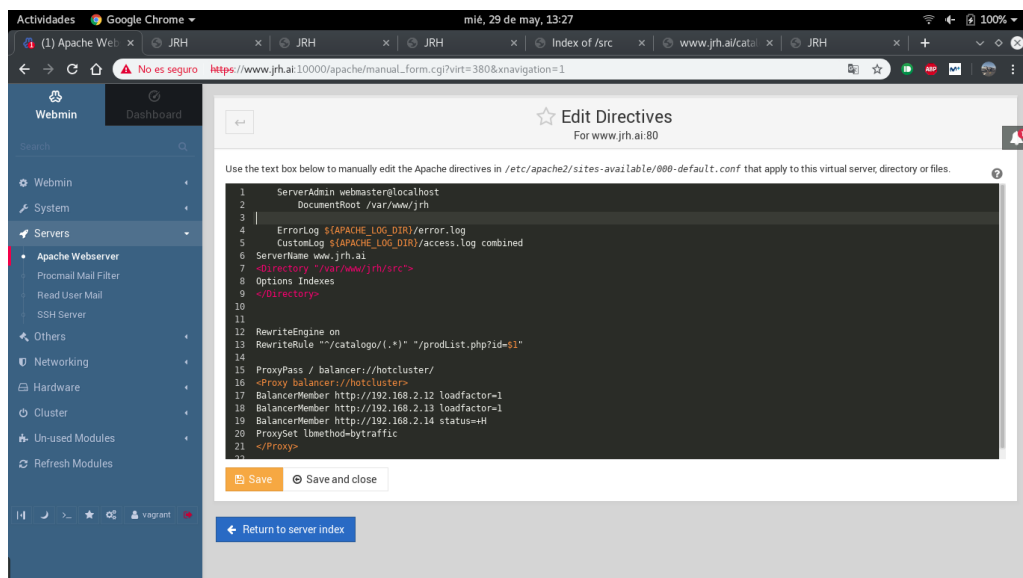


Figura 11: Directivas jrh.

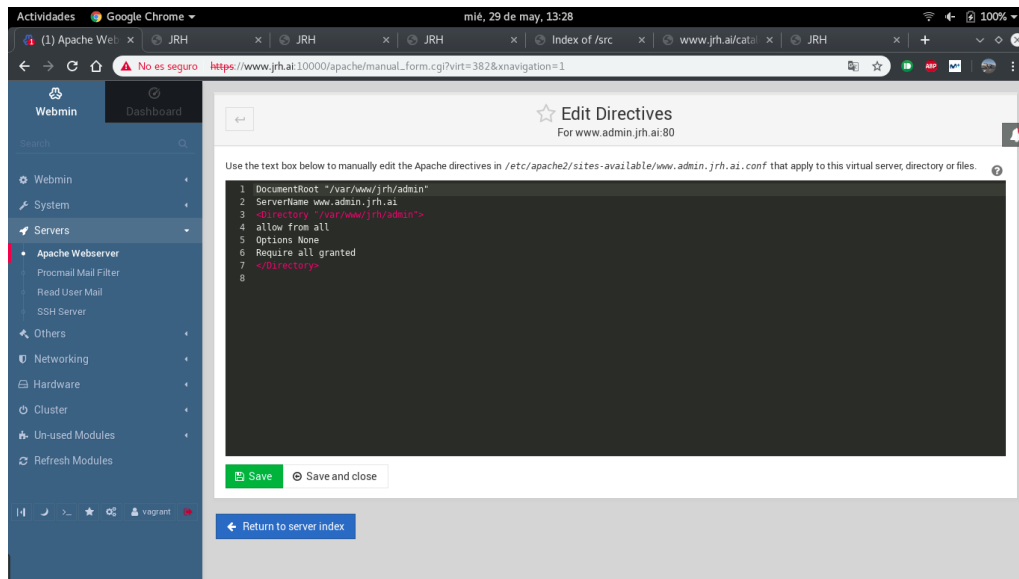


Figura 12: Directivas admin.

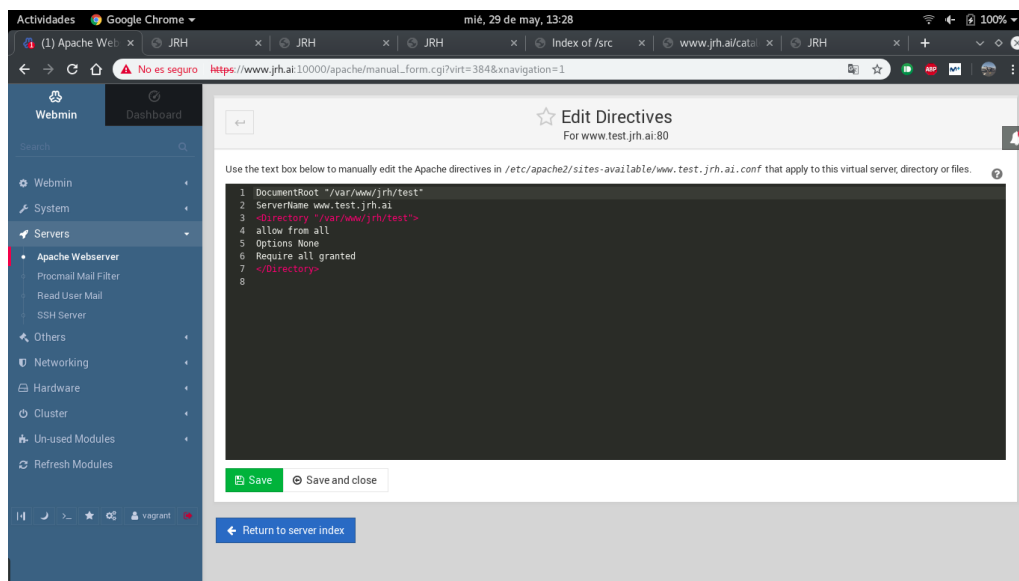


Figura 13: Directivas test.

4. Configuración avanzada de Apache

Como en este examen no se ha configurado el servidor DNS, le pondremos directamente la dirección IP de los otros nodos.

Para hacer el balanceo de carga, debemos tener instalado Apache en todos los nodos con:

```
sudo apt-get install apache2
```

Para comprobarlo modificaremos el fichero `index.html` de la siguiente forma:

```

1 <html>
2 <head>
3 <title>JRH</title>
4 </head>
5 <body>
6 <p>Página de inicio de JRH en nodo 1.</p>
7 </body>
8 </html>

```

Para los nodos 2 y 3 estará modificado también.

Ahora, nos dirigimos a webmin y habilitamos los siguientes módulos de Apache:

- lbmethod_bybusyness
- lbmethod_byrequests
- lbmethod_bytraffic
- lbmethod_heartbeat
- proxy_balancer
- proxy_html
- proxy_http

Luego, nos dirigimos al host virtual por defecto de la máquina que actúa como balanceador y, en “Edit directives” ponemos lo siguiente:

```

ProxyPass / balancer://hotcluster/ <Proxy balancer://hotcluster>BalancerMember
http://192.168.2.12 loadfactor=1 BalancerMember http://192.168.2.13
loadfactor=1 BalancerMember http://192.168.2.14 status=+H ProxySet
lbmethod=bytraffic </Proxy>

```

Para comprobar que funciona, solo debemos entrar dos veces en la página para que podamos ver lo siguiente:

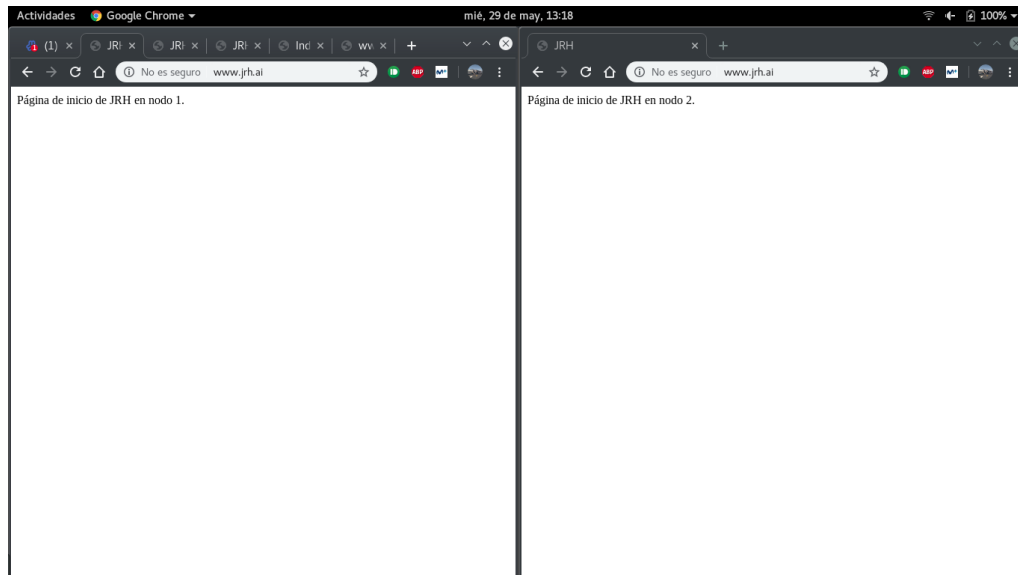


Figura 14: Balanceo de carga.