Cortafuegos con `iptable`

Ismael Macareno Chouikh

2025-01-26

Índice

1. Materiales previos

- Máquina Virtual Debian 11
  - interfaz `enp0s3`
    - adaptador en modo puente
  - interfaz `enp0s8`
    - adaptador en modo red interna con la subred 10.0.X.0/24
  - Este equipo hace de *router* e implementa NAT
- Otros equipos clientes
  - configurar el adaptador de red en modo red interna

1.1. Configuración de red

Máquina	Dirección IP
Debian 11 server	10.0.1.10/24
Ubuntu 24.04 cliente	10.0.1.11/24
Windows 7 ultimate cliente	10.0.1.12/24

2. Cortafuegos con `IPTABLES`

Lo primero que tendremos que realizar es configurar todo lo realizado con la configuración de red de la máquina *server*.

Para configurar la red de nuestra MV Debian 11 lo que tendremos que hacer es modificar el fichero `/etc/network/interface` de la siguiente manera

```
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

source /etc/network/interfaces.d/*

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
allow-hotplug enp0s3
iface enp0s3 inet dhcp

allow-hotplug enp0s8
iface enp0s8 inet static
        address 10.0.1.10
        netmask 255.255.255.0
```

Luego, para que se aplique la configuración de red tendremos que ejecutar los siguientes comandos:

- `sudo systemctl restart networking.service`

- `sudo ifup enp0s3`
- `sudo ifup enp0s8`

Para que la máquina implemente NAT lo que tendremos que hacer será modificar el fichero `/etc/sysctl.conf` de la siguiente manera:

```
#####################################################################
# Functions previously found in netbase
#

# Uncomment the next two lines to enable Spoof protection (reverse-path filter)
# Turn on Source Address Verification in all interfaces to
# prevent some spoofing attacks
#net.ipv4.conf.default.rp_filter=1
#net.ipv4.conf.all.rp_filter=1

# Uncomment the next line to enable TCP/IP SYN cookies
# See http://lwn.net/Articles/277146/
# Note: This may impact IPv6 TCP sessions too
#net.ipv4.tcp_syncookies=1

# Uncomment the next line to enable packet forwarding for IPv4
net.ipv4.ip_forward=1
```

Y en las máquinas clientes lo único que tendremos que hacer será asegurar que su tarjeta de red este en modo **red interna** y tengan un IP ya sea por DHCP o estática de la misma red que la IP de la tarjeta de red `enp0s8` de nuestra máquina servidora.

2.1. Consultar el estado actual de las reglas implementadas en el cortafuegos

- `sudo iptables -L`
- `sudo iptables -t nat -L`

Si te falla iptables

Instalar con el comando sudo apt install iptables

```
root@debian11:~# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
root@debian11:~# iptables -t nat -L
Chain PREROUTING (policy ACCEPT)
target     prot opt source               destination
```

```
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination

Chain POSTROUTING (policy ACCEPT)
target     prot opt source               destination
```

2.2. Bloquear el `ping` al cortafuegos

- `sudo iptables -A INPUT -p icmp -j DROP`

Si añadimos esta regla al cortafuegos e intentamos hacer un `ping` desde la máquina real a la interfaz en modo puente no nos contestará el `ping`

```
root@debian11:~# tcpdump -i enp0s3 icmp
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on enp0s3, link-type EN10MB (Ethernet), snapshot length 262144 bytes
16:14:25.957059 IP 192.168.1.136 > 192.168.1.128: ICMP echo request, id 4, seq 55, length 64
16:14:26.981468 IP 192.168.1.136 > 192.168.1.128: ICMP echo request, id 4, seq 56, length 64
16:14:28.006011 IP 192.168.1.136 > 192.168.1.128: ICMP echo request, id 4, seq 57, length 64
16:14:29.029286 IP 192.168.1.136 > 192.168.1.128: ICMP echo request, id 4, seq 58, length 64
16:14:30.054089 IP 192.168.1.136 > 192.168.1.128: ICMP echo request, id 4, seq 59, length 64
```

En el bloque de arriba se puede ver como la MV debian 11 está recibiendo paquetes `icmp` desde una máquina con una dirección IP 192.168.1.136 que es mi máquina real.

En caso de que queramos que acepte los paquetes `icmp` de nuevo lo que tendremos que hacer será ejecutar el siguiente comando:

- `sudo iptables -F`

```
root@debian11:~# tcpdump -i enp0s3 icmp
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on enp0s3, link-type EN10MB (Ethernet), snapshot length 262144 bytes
16:16:21.669002 IP 192.168.1.136 > 192.168.1.128: ICMP echo request, id 4, seq 168, length 64
16:16:21.669058 IP 192.168.1.128 > 192.168.1.136: ICMP echo reply, id 4, seq 168, length 64
16:16:22.693951 IP 192.168.1.136 > 192.168.1.128: ICMP echo request, id 4, seq 169, length 64
16:16:22.694007 IP 192.168.1.128 > 192.168.1.136: ICMP echo reply, id 4, seq 169, length 64
16:16:23.717935 IP 192.168.1.136 > 192.168.1.128: ICMP echo request, id 4, seq 170, length 64
16:16:23.718005 IP 192.168.1.128 > 192.168.1.136: ICMP echo reply, id 4, seq 170, length 64
16:16:24.740902 IP 192.168.1.136 > 192.168.1.128: ICMP echo request, id 4, seq 171, length 64
16:16:24.740951 IP 192.168.1.128 > 192.168.1.136: ICMP echo reply, id 4, seq 171, length 64
```

En el bloque de arriba se puede apreciar como ya no es solo mi máquina real la que pregunta, si no que también responde la máquina virtual debian 11.

2.3. Bloquear el acceso al servidor web del equipo cortafuegos (se supone instalado `apache2`)

Antes de comenzar este apartado tendremos que comprobar si tenemos instalado el servidor web `apache2` o no para que en caso de que no este instalado poder instalarlo.

Para comprobar si está instalado podremos ejecutar el siguiente comando:

- `sudo systemctl status apache2.service`

Y en caso de que tengamos que instalarlo los comandos que tendríamos que ejecutar serían:

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

Para poder bloquear el acceso al servidor web del equipo cortafuegos tendremos que implementar la siguiente regla:

- `sudo iptables -A INPUT -p tcp --dport 80 -j DROP`

```
root@debian11:~# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
DROP       tcp  --  anywhere             anywhere             tcp dpt:http

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
```

Para comprobar que la regla se ha aplicado correctamente lo único que tendríamos que hacer sería ir a nuestra máquina real y acceder al servidor web de la MV poniendo en el navegador web `http://DIRECCIONIPMAQUINAVIRTUAL`. El resultado de esto sería que la página se queda pensando y al final no muestra nada.

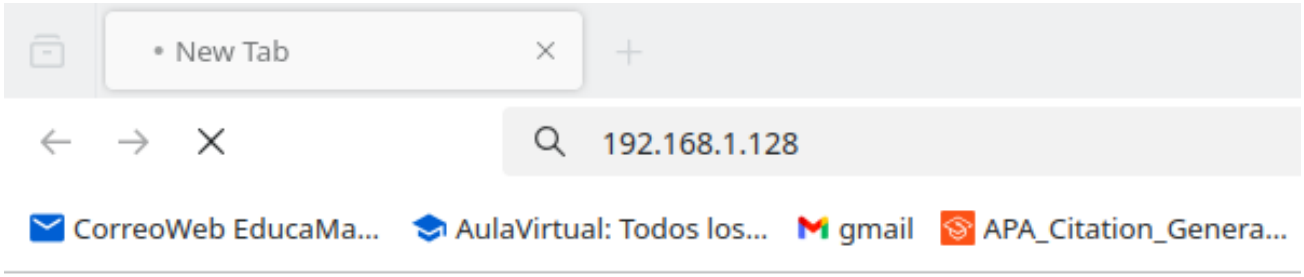

Figura 1: Macareno, Ismael (2025). Ejemplo de acceso denegado a servicio web [PNG]. Propia

En caso de que quisiésemos que se pudiese acceder al servidor web lo único que tendríamos que hacer sería ejecutar la siguiente regla:

- `sudo iptables -F`

2.4. Bloquear el acceso a internet desde cualquier equipo de la red interna

Para realizar este apartado las reglas que tendríamos que aplicar serían:

- `sudo iptables -A FORWARD -p tcp --dport 80 -j DROP`

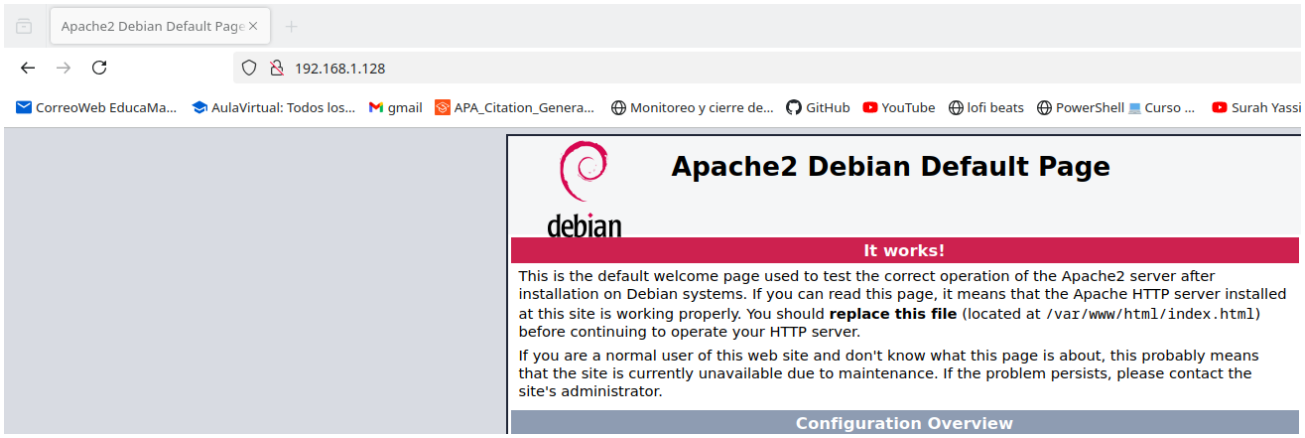

Figura 2: Macareno, Ismael (2025). Ejemplo de acceso permitido a servicio web [PNG]. Propia

- `sudo iptables -A FORWARD -p tcp --dport 443 -j DROP`

Para comprobar esta regla lo que tendríamos que hacer sería acceder a nuestra MV cliente y realizar las siguientes comprobaciones:

1. La MV cliente tiene como *gateway* la máquina que hace NAT
2. La tarjeta de red la tiene configurada en modo red interna (no tiene acceso a internet)

En mi caso voy a usar a modo de cliente una MV Ubuntu 24.04 LTS con la siguiente configuración de red

```
# This file is generated from information provided by the datasource.  Changes
# to it will not persist across an instance reboot.  To disable cloud-init's
# network configuration capabilities, write a file
# /etc/cloud/cloud.cfg.d/99-disable-network-config.cfg with the following:
# network: {config: disabled}
network:
    ethernets:
        enp0s3:
            dhcp4: no
            addresses:
              - 10.0.1.11/24
            routes:
              - to: default
                via: 10.0.1.10
            nameservers:
              addresses: [8.8.8.8, 8.8.4.4]
    version: 2
```

Listing 1: Macareno, Ismael (2025). Ejemplo de configuración para fichero `/etc/netplan/X.yaml/` [PNG]. Propia

CUIDADO

Los ficheros .yaml son de sintáxis muy estricta. Nada de tabulaciones, solo espacios

Una vez ya configurada nuestra MV cliente lo que tendríamos que hacer sería intentar hacer búsquedas usando el navegador web de la MV cliente.

2.5. Bloquear el acceso por ssh al equipo cortafuegos desde cualquier sitio

Para realizar este apartado la regla que necesitaremos será la siguiente:

- `sudo iptables -A INPUT -p tcp --dport 22 -j DROP`

Para hacer las comprobaciones lo que tendremos que hacer será en la MV que implementa NAT ejecutar el siguiente comando:

- `sudo tcpdump -i enp0s3 port 22 -w denySSH.pcap`

Mientras que nuestra máquina real intenta realizar una conexión SSH a la máquina servidora.

Luego podremos analizar el fichero `denySSH.pcap` usando el comando:

- `sudo tcpdump -r denySSH.pcap`

```
root@debian11:~# tcpdump -r denySSH.pcap
reading from file denySSH.pcap, link-type EN10MB (Ethernet), snapshot length 262144
16:08:34.814635 IP 192.168.1.136.42694 > 192.168.1.128.ssh: Flags [S], seq 401353934, win 64240, optio
16:08:35.839002 IP 192.168.1.136.42694 > 192.168.1.128.ssh: Flags [S], seq 401353934, win 64240, optio
16:08:36.861925 IP 192.168.1.136.42694 > 192.168.1.128.ssh: Flags [S], seq 401353934, win 64240, optio
16:08:37.886952 IP 192.168.1.136.42694 > 192.168.1.128.ssh: Flags [S], seq 401353934, win 64240, optio
16:08:38.909918 IP 192.168.1.136.42694 > 192.168.1.128.ssh: Flags [S], seq 401353934, win 64240, optio
16:08:39.933921 IP 192.168.1.136.42694 > 192.168.1.128.ssh: Flags [S], seq 401353934, win 64240, optio
16:08:41.981917 IP 192.168.1.136.42694 > 192.168.1.128.ssh: Flags [S], seq 401353934, win 64240, optio
16:08:46.013993 IP 192.168.1.136.42694 > 192.168.1.128.ssh: Flags [S], seq 401353934, win 64240, optio
```

2.6. Bloquear el acceso por ssh al equipo cortafuegos a todos los equipos de la red interna

Para este apartado la regla que tendremos que implementar es la siguiente:

- `sudo iptables -A INPUT -s 10.0.X.0/24 -p tcp --dport 22 -j DROP`

Para la comprobaciones sucede lo siguiente:

- Desde la máquina real nos podremos conectar mediante **ssh** gracias a la tarjeta de red configurada en modo puente.
- Desde la MV cliente Ubuntu 24.04 no nos podremos conectar mediante **ssh** ya que está va a la dirección IP 10.0.1.10/24 la cuál deniega.

```
root@debian11:~# tcpdump -i enp0s8 port 22
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on enp0s8, link-type EN10MB (Ethernet), snapshot length 262144 bytes
16:20:26.573334 IP 10.0.1.11.44044 > 10.0.1.10.ssh: Flags [S], seq 2960647050, win 64240, options [mss
16:20:27.627756 IP 10.0.1.11.44044 > 10.0.1.10.ssh: Flags [S], seq 2960647050, win 64240, options [mss
16:20:28.667773 IP 10.0.1.11.44044 > 10.0.1.10.ssh: Flags [S], seq 2960647050, win 64240, options [mss
16:20:29.692403 IP 10.0.1.11.44044 > 10.0.1.10.ssh: Flags [S], seq 2960647050, win 64240, options [mss
16:20:30.772064 IP 10.0.1.11.44044 > 10.0.1.10.ssh: Flags [S], seq 2960647050, win 64240, options [mss
16:20:31.849217 IP 10.0.1.11.44044 > 10.0.1.10.ssh: Flags [S], seq 2960647050, win 64240, options [mss
16:20:33.852031 IP 10.0.1.11.44044 > 10.0.1.10.ssh: Flags [S], seq 2960647050, win 64240, options [mss
16:20:37.883715 IP 10.0.1.11.44044 > 10.0.1.10.ssh: Flags [S], seq 2960647050, win 64240, options [mss
c^C
8 packets captured
```

```
8 packets received by filter
0 packets dropped by kernel
```

2.7. Establecer política restrictiva por defecto

Para este apartado tendremos que añadir a nuestras `iptables` las siguientes reglas:

- `sudo iptables -P INPUT DROP`
- `sudo iptables -P OUTPUT DROP`
- `sudo iptables -P FORWARD DROP`

Las comprobaciones en este apartado serían las mismas que hemos estado realizando hasta ahora:

- Hacer `ping` desde la máquina real y la virtual
- Conexión `ssh` desde la máquina real y la virtual
- Analizar en la máquina que implementa NAT mediante el comando `tcpdump`

```
16:27:04.562911 IP 192.168.1.136 > 192.168.1.128: ICMP echo request, id 5, seq 13, length 64
16:27:05.585477 IP 192.168.1.136 > 192.168.1.128: ICMP echo request, id 5, seq 14, length 64
16:27:06.609762 IP 192.168.1.136 > 192.168.1.128: ICMP echo request, id 5, seq 15, length 64
16:27:07.634828 IP 192.168.1.136 > 192.168.1.128: ICMP echo request, id 5, seq 16, length 64
16:27:08.657719 IP 192.168.1.136 > 192.168.1.128: ICMP echo request, id 5, seq 17, length 64
^C
5 packets captured
5 packets received by filter
0 packets dropped by kernel
root@debian11:~# tcpdump -i enp0s3 port 22
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on enp0s3, link-type EN10MB (Ethernet), snapshot length 262144 bytes
16:27:24.593497 IP 192.168.1.136.59426 > 192.168.1.128.ssh: Flags [S], seq 808198130, win 64240, opt
ions [mss 1460,sackOK,TS val 1138833410 ecr 0,nop,wscale 7], length 0
16:27:28.625180 IP 192.168.1.136.59426 > 192.168.1.128.ssh: Flags [S], seq 808198130, win 64240, opt
ions [mss 1460,sackOK,TS val 1138837442 ecr 0,nop,wscale 7], length 0
16:27:30.674372 IP 192.168.1.136.46208 > 192.168.1.128.ssh: Flags [P.], seq 2984386892:2984386976, a
ck 2754379429, win 636, options [nop,nop,TS val 1138839491 ecr 2365568024], length 84
^C
3 packets captured
3 packets received by filter
0 packets dropped by kernel
root@debian11:~# tcpdump -i enp0s8 port 22
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on enp0s8, link-type EN10MB (Ethernet), snapshot length 262144 bytes
16:27:53.363650 IP 10.0.1.11.53238 > 10.0.1.10.ssh: Flags [S], seq 2772910892, win 64240, options [m
ss 1460,sackOK,TS val 540083825 ecr 0,nop,wscale 7], length 0
16:27:57.415877 IP 10.0.1.11.53238 > 10.0.1.10.ssh: Flags [S], seq 2772910892, win 64240, options [m
ss 1460,sackOK,TS val 540087878 ecr 0,nop,wscale 7], length 0
16:28:05.859169 IP 10.0.1.11.53238 > 10.0.1.10.ssh: Flags [S], seq 2772910892, win 64240, options [m
ss 1460,sackOK,TS val 540096321 ecr 0,nop,wscale 7], length 0
^C
3 packets captured
3 packets received by filter
0 packets dropped by kernel
root@debian11:~#
[0] 0:bash*                                                    "debian11" 16:28 26-ene-25
```

Figura 3: Macareno, Ismael (2025). Comprobación mediante `tcpdump` en la máquina *server* [PNG]. Propia

AVISO

No eliminar estas reglas

2.8. Manteniendo la política restrictiva por defecto, permitir ahora cualquier conexión por la interfaz local

Las reglas a implementar ahora serían:

- `sudo iptables -A INPUT -i lo -j ACCEPT`
- `sudo iptables -A OUTPUT -o lo -j ACCEPT`

Para comprobar este apartado lo que tendríamos que hacer sería acceder a nuestro servidor web de manera local.

En caso de que nuestra máquina *server* no tuviese entorno gráfico (mi caso) podremos hacer la comprobación mediante el comando `wget` o `curl` de la siguiente manera:

- `wget http://localhost`

Figura 4: Macareno, Ismael (2025). Comprobación mediante `wget` en la máquina *server* [PNG]. Propia

AVISO

No eliminar estas reglas

2.9. Manteniendo la política restrictiva por defecto, permitir atravesar el cortafuegos desde/hacia la red interna de las peticiones web (http y https) y DNS (53/udp)

Las reglas a implementar para este apartado sería:

- `sudo iptables -A FORWARD -s 10.0.X.0/24 -p udp --dport 53 -j ACCEPT`
- `sudo iptables -A FORWARD -s 10.0.X.0/24 -p tcp --dport http -j ACCEPT`
- `sudo iptables -A FORWARD -s 10.0.X.0/24 -p tcp --dport https -j ACCEPT`
- `sudo iptables -A FORWARD -m state --state ESTABLISHED,RELATED -j ACCEPT`

```
root@debian11:~# iptables -A FORWARD -s 10.0.1.0/24 -p udp --dport 53 -j ACCEPT
root@debian11:~# iptables -A FORWARD -s 10.0.1.0/24 -p tcp --dport http -j ACCEPT
root@debian11:~# iptables -A FORWARD -s 10.0.1.0/24 -p tcp --dport https -j ACCEPT
root@debian11:~# iptables -A FORWARD -m state --state ESTABLISHED,RELATED -j ACCEPT
root@debian11:~# iptables -L
Chain INPUT (policy DROP)
target     prot opt source               destination
DROP       tcp  --  10.0.1.0/24         anywhere             tcp dpt:ssh
ACCEPT     all  --  anywhere            anywhere

Chain FORWARD (policy DROP)
target     prot opt source               destination
ACCEPT     udp  --  10.0.1.0/24         anywhere             udp dpt:domain
ACCEPT     tcp  --  10.0.1.0/24         anywhere             tcp dpt:http
ACCEPT     tcp  --  10.0.1.0/24         anywhere             tcp dpt:https
ACCEPT     all  --  anywhere            anywhere             state RELATED,ESTABLISHED

Chain OUTPUT (policy DROP)
target     prot opt source               destination
ACCEPT     all  --  anywhere            anywhere
```

2.10. Eliminar todas las reglas creadas hasta ahora

Esto sería mediante la instrucción:

- `sudo iptables -F`
- `sudo iptables -t nat -F`

2.11. Establecer política permisiva por defecto

Para este apartado las instrucciones necesarias serían:

- `sudo iptables -P INPUT ACCEPT`
- `sudo iptables -P OUTPUT ACCEPT`
- `sudo iptables -P FORWARD ACCEPT`

```
root@debian11:~# tcpdump -i enp0s3 icmp
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on enp0s3, link-type EN10MB (Ethernet), snapshot length 262144 bytes
16:53:00.407043 IP 192.168.1.136 > 192.168.1.128: ICMP echo request, id 8, seq 1, length 64
16:53:00.407113 IP 192.168.1.128 > 192.168.1.136: ICMP echo reply, id 8, seq 1, length 64
16:53:00.729563 IP 192.168.1.128 > Livebox: ICMP 192.168.1.128 udp port netbios-ns unreachable, length
16:53:01.407538 IP 192.168.1.136 > 192.168.1.128: ICMP echo request, id 8, seq 2, length 64
16:53:01.407613 IP 192.168.1.128 > 192.168.1.136: ICMP echo reply, id 8, seq 2, length 64
16:53:01.962431 IP 192.168.1.128 > Livebox: ICMP 192.168.1.128 udp port netbios-ns unreachable, length
^C
6 packets captured
11 packets received by filter
0 packets dropped by kernel
root@debian11:~# tcpdump -i enp0s3 port 22
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on enp0s3, link-type EN10MB (Ethernet), snapshot length 262144 bytes
16:53:07.751867 IP 192.168.1.128.ssh > 192.168.1.136.40812: Flags [P.], seq 3460243617:3460243813, ack
16:53:07.752471 IP 192.168.1.136.40812 > 192.168.1.128.ssh: Flags [.], ack 196, win 750, options [nop,
```

```
16:53:07.830747 IP 192.168.1.128.ssh > 192.168.1.136.40812: Flags [P.], seq 196:584, ack 1, win 501, o
16:53:07.831255 IP 192.168.1.136.40812 > 192.168.1.128.ssh: Flags [.], ack 584, win 749, options [nop,
16:53:07.922169 IP 192.168.1.128.ssh > 192.168.1.136.40812: Flags [P.], seq 584:940, ack 1, win 501, o
16:53:07.922785 IP 192.168.1.136.40812 > 192.168.1.128.ssh: Flags [.], ack 940, win 749, options [nop,
16:53:08.025901 IP 192.168.1.128.ssh > 192.168.1.136.40812: Flags [P.], seq 940:1296, ack 1, win 501,
16:53:08.026567 IP 192.168.1.136.40812 > 192.168.1.128.ssh: Flags [.], ack 1296, win 749, options [nop
16:53:08.129795 IP 192.168.1.128.ssh > 192.168.1.136.40812: Flags [P.], seq 1296:1652, ack 1, win 501,
16:53:08.130392 IP 192.168.1.136.40812 > 192.168.1.128.ssh: Flags [.], ack 1652, win 749, options [nop
16:53:08.233764 IP 192.168.1.128.ssh > 192.168.1.136.40812: Flags [P.], seq 1652:2008, ack 1, win 501,
16:53:08.234405 IP 192.168.1.136.40812 > 192.168.1.128.ssh: Flags [.], ack 2008, win 749, options [nop
16:53:08.338179 IP 192.168.1.128.ssh > 192.168.1.136.40812: Flags [P.], seq 2008:2364, ack 1, win 501,
16:53:08.338948 IP 192.168.1.136.40812 > 192.168.1.128.ssh: Flags [.], ack 2364, win 749, options [nop
^C
14 packets captured
18 packets received by filter
0 packets dropped by kernel
```

AVISO

No eliminar estas reglas

2.12. Bloquear lo que venga de la red interna y registrarlo en el `LOG`

Las reglas necesarias para este apartado serían:

- `sudo iptables -A INPUT -i enp0s8 -s 10.0.X.0/24 -j LOG --log-prefix ÏP_SPOOF: "`
- `sudo iptables -A INPUT -i enp0s8 -s 10.0.0.0/24 -j DROP`

Para las comprobaciones lo que tendríamos que hacer sería mirar los `logs` de la siguiente manera:

- `tail -f /var/log/syslog`

```
root@debian11:~# tail -f /var/log/syslog
Jan 26 17:01:01 debian11 kernel: [ 5976.136357] IP_SPOOF: IN=enp0s8 OUT= MAC=08:00:27:1d:af:7c:08:00:2
Jan 26 17:01:02 debian11 kernel: [ 5977.153905] IP_SPOOF: IN=enp0s8 OUT= MAC=08:00:27:1d:af:7c:08:00:2
Jan 26 17:01:03 debian11 kernel: [ 5978.166550] IP_SPOOF: IN=enp0s8 OUT= MAC=08:00:27:1d:af:7c:08:00:2
Jan 26 17:01:04 debian11 kernel: [ 5979.317392] IP_SPOOF: IN=enp0s8 OUT= MAC=08:00:27:1d:af:7c:08:00:2
Jan 26 17:01:05 debian11 kernel: [ 5980.342660] IP_SPOOF: IN=enp0s8 OUT= MAC=08:00:27:1d:af:7c:08:00:2
Jan 26 17:01:06 debian11 kernel: [ 5981.366015] IP_SPOOF: IN=enp0s8 OUT= MAC=08:00:27:1d:af:7c:08:00:2
Jan 26 17:01:07 debian11 kernel: [ 5982.390167] IP_SPOOF: IN=enp0s8 OUT= MAC=08:00:27:1d:af:7c:08:00:2
Jan 26 17:01:08 debian11 kernel: [ 5983.438492] IP_SPOOF: IN=enp0s8 OUT= MAC=08:00:27:1d:af:7c:08:00:2
Jan 26 17:01:09 debian11 kernel: [ 5984.566104] IP_SPOOF: IN=enp0s8 OUT= MAC=08:00:27:1d:af:7c:08:00:2
Jan 26 17:01:10 debian11 kernel: [ 5985.590261] IP_SPOOF: IN=enp0s8 OUT= MAC=08:00:27:1d:af:7c:08:00:2
Jan 26 17:01:11 debian11 kernel: [ 5986.649458] IP_SPOOF: IN=enp0s8 OUT= MAC=08:00:27:1d:af:7c:08:00:2
Jan 26 17:01:12 debian11 kernel: [ 5987.708359] IP_SPOOF: IN=enp0s8 OUT= MAC=08:00:27:1d:af:7c:08:00:2
Jan 26 17:01:13 debian11 kernel: [ 5988.734241] IP_SPOOF: IN=enp0s8 OUT= MAC=08:00:27:1d:af:7c:08:00:2
```

AVISO

Eliminar estas reglas

2.13. Redirigir el tráfico web que entra por la interfaz externa (enp0s3) al servidor web de la red interna

AVISO

En la máquina cliente tendrá que estar instalado el servidor apache

Las reglas necesarias para este apartado serían:

- `sudo iptables -t nat -A PREROUTING -i enp0s3 -p tcp --dport 80 -j DNAT --to 10.0.X.2:80`
- `sudo iptables -A FORWARD -i enp0s3 -o enp0s8 -d 10.0.X.2 -p tcp --dport http -m state --state NEW -j ACCEPT`

Desde el exterior se permite atravesar el cortafuegos para ir al puerto 80 del equipo con IP 10.0.1.11, siempre y cuando sea una petición (estado NEW). Y, por otro lado, el cortafuegos, antes de realizar el enrutamiento, redirecciona las peticiones web (tcp `--dport 80`) al puerto 80 de la máquina cliente

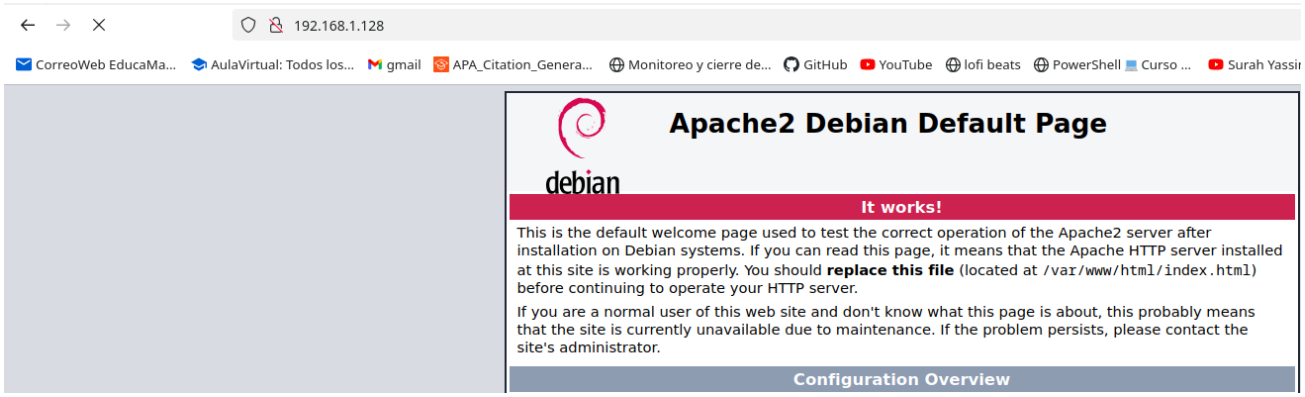

Figura 5: Macareno, Ismael (2025). Comprobación redirección de servidor web [PNG]. Propia

2.14. Aceptar los pings desde la máquina interna Windows y bloquear los pings desde la máquina interna Linux.

Las relgas necesarias para este apartado serían las siguientes:

- `sudo iptables -A INPUT -s 10.0.1.12 -p icmp --icmp-type echo-request -j ACCEPT`
- `sudo iptables -A INPUT -s 10.0.1.11 -p icmp --icmp-type echo-request -j DROP`

```
root@debian11:~# tcpdump -i enp0s8 icmp
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on enp0s8, link-type EN10MB (Ethernet), snapshot length 262144 bytes
17:29:37.769773 IP 10.0.1.12 > 10.0.1.10: ICMP echo request, id 1, seq 11, length 40
17:29:37.769823 IP 10.0.1.10 > 10.0.1.12: ICMP echo reply, id 1, seq 11, length 40
17:29:38.765836 IP 10.0.1.12 > 10.0.1.10: ICMP echo request, id 1, seq 12, length 40
17:29:38.765919 IP 10.0.1.10 > 10.0.1.12: ICMP echo reply, id 1, seq 12, length 40
17:29:39.765033 IP 10.0.1.12 > 10.0.1.10: ICMP echo request, id 1, seq 13, length 40
17:29:39.765069 IP 10.0.1.10 > 10.0.1.12: ICMP echo reply, id 1, seq 13, length 40
17:29:40.765475 IP 10.0.1.12 > 10.0.1.10: ICMP echo request, id 1, seq 14, length 40
17:29:40.765546 IP 10.0.1.10 > 10.0.1.12: ICMP echo reply, id 1, seq 14, length 40
17:29:43.944966 IP 10.0.1.11 > 10.0.1.10: ICMP echo request, id 8496, seq 1, length 64
17:29:44.975767 IP 10.0.1.11 > 10.0.1.10: ICMP echo request, id 8496, seq 2, length 64
```

```
^C
10 packets captured
10 packets received by filter
0 packets dropped by kernel
```

```
root@debian11:~# tcpdump -i enp0s8 icmp
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on enp0s8, link-type EN10MB (Ethernet), snapshot length 262144 bytes
17:30:24.401255 IP 10.0.1.11 > 10.0.1.10: ICMP echo request, id 8506, seq 4, length 64
17:30:25.473660 IP 10.0.1.11 > 10.0.1.10: ICMP echo request, id 8506, seq 5, length 64
17:30:26.507622 IP 10.0.1.11 > 10.0.1.10: ICMP echo request, id 8506, seq 6, length 64
17:30:27.537481 IP 10.0.1.11 > 10.0.1.10: ICMP echo request, id 8506, seq 7, length 64
17:30:28.559332 IP 10.0.1.11 > 10.0.1.10: ICMP echo request, id 8506, seq 8, length 64
17:30:29.622246 IP 10.0.1.11 > 10.0.1.10: ICMP echo request, id 8506, seq 9, length 64
17:30:30.669962 IP 10.0.1.11 > 10.0.1.10: ICMP echo request, id 8506, seq 10, length 64
17:30:31.705485 IP 10.0.1.11 > 10.0.1.10: ICMP echo request, id 8506, seq 11, length 64
17:30:32.722375 IP 10.0.1.11 > 10.0.1.10: ICMP echo request, id 8506, seq 12, length 64
^C
9 packets captured
9 packets received by filter
0 packets dropped by kernel
```

2.15. Aceptar los pings desde la máquina interna Linux y bloquear los pings desde la máquina interna Windows.

Las instrucciones necesarias serían las siguientes:

- `sudo iptables -A INPUT -s 10.0.1.11 -p icmp --icmp-type echo-request -j ACCEPT`
- `sudo iptables -A INPUT -s 10.0.1.12 -p icmp --icmp-type echo-request -j DROP`

```
root@debian11:~# tcpdump -i enp0s8 icmp
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on enp0s8, link-type EN10MB (Ethernet), snapshot length 262144 bytes
17:32:40.963625 IP 10.0.1.12 > 10.0.1.10: ICMP echo request, id 1, seq 15, length 40
17:32:45.735881 IP 10.0.1.12 > 10.0.1.10: ICMP echo request, id 1, seq 16, length 40
17:32:50.735809 IP 10.0.1.12 > 10.0.1.10: ICMP echo request, id 1, seq 17, length 40
17:32:55.736141 IP 10.0.1.12 > 10.0.1.10: ICMP echo request, id 1, seq 18, length 40
^C
4 packets captured
4 packets received by filter
0 packets dropped by kernel
```

```
root@debian11:~# tcpdump -i enp0s8 icmp
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on enp0s8, link-type EN10MB (Ethernet), snapshot length 262144 bytes
17:33:42.168698 IP 10.0.1.11 > 10.0.1.10: ICMP echo request, id 8546, seq 1, length 64
17:33:42.168842 IP 10.0.1.10 > 10.0.1.11: ICMP echo reply, id 8546, seq 1, length 64
```

```
17:33:43.174021 IP 10.0.1.11 > 10.0.1.10: ICMP echo request, id 8546, seq 2, length 64
17:33:43.174108 IP 10.0.1.10 > 10.0.1.11: ICMP echo reply, id 8546, seq 2, length 64
17:33:44.199597 IP 10.0.1.11 > 10.0.1.10: ICMP echo request, id 8546, seq 3, length 64
17:33:44.199668 IP 10.0.1.10 > 10.0.1.11: ICMP echo reply, id 8546, seq 3, length 64
17:33:45.222092 IP 10.0.1.11 > 10.0.1.10: ICMP echo request, id 8546, seq 4, length 64
17:33:45.222162 IP 10.0.1.10 > 10.0.1.11: ICMP echo reply, id 8546, seq 4, length 64
17:33:46.238390 IP 10.0.1.11 > 10.0.1.10: ICMP echo request, id 8546, seq 5, length 64
17:33:46.238461 IP 10.0.1.10 > 10.0.1.11: ICMP echo reply, id 8546, seq 5, length 64
^C
10 packets captured
10 packets received by filter
0 packets dropped by kernel
```