MANUAL DE USUARIO

CONFIGURACIÓN DE SERVIDOR NAT CON IPTABLES

(LINUX)

## IPTABLES

IPTABLES es el componente más conocido del proyecto netfilter y es una herramienta que funciona en el espacio de usuario y que permite definir reglas para el filtrado y la modificación de paquetes TCP/IP que pasen por cualquiera de las interfaces de red de un equipo.

## NAT

NAT son las siglas del inglés network address translation o traducción de direcciones de red y es un mecanismo que se usa ampliamente, fundamentalmente porque permite compartir una dirección IP pública por muchos equipos y esto es imprescindible en muchas situaciones por la escasez de direcciones IPv4.

Existen diferentes tipos de NAT, dependiendo de si se cambia la dirección IP origen o la dirección IP destino del paquete que abre la conexión, incluso existe una extensión de NAT que permite modificar el puerto origen o destino. Estos tipos de variantes de NAT reciben diferentes nombres dependiendo de la implementación, aquí se presenta los nombres más utilizados cuando se implementa NAT con iptables.

### Source NAT

Este tipo de NAT es en el que se cambia la dirección IP de origen, es la situación más utilizada cuando estamos utilizando una dirección IP privada (RFC 1918) en una red local y establecemos una conexión con un equipo de Internet. Un equipo de la red (normalmente la puerta de enlace) se encarga de cambiar la dirección IP privada origen por la dirección IP pública, para que el equipo de Internet pueda contestar. Los pasos que se seguirían serían algo como:

* Un equipo de una red local con una dirección IP privada (supongamos 192.168.3.14) quiere solicitar una página web (puerto 80/tcp) del equipo de Internet (Ej. www.google.com).
* Realiza una consulta DNS y obtiene que el equipo que aloja dicha página tiene la dirección IP 76.74.254.126
* Consulta su tabla de encaminamiento y como no está en la misma red que el servidor web de google, envía el paquete con la solicitud de la página al equipo que es su destino por defecto (puerta de enlace o gateway), que supongamos tiene la dirección 192.168.3.254.
* El gateway, que en este caso debe actuar como dispositivo de NAT, recibe el paquete y comprueba la dirección IP destino, como no es la suya, lo envía a su propio destino por defecto (gateway) que ya será una dirección IP pública.
* Antes de que el paquete salga por la interfaz de red externa, se le cambia la dirección IP origen (192.168.3.14) por la dirección IP pública (supongamos que fuese 80.58.1.14) y se guarda la petición en lo que se denomina tablas de NAT (anotando también el puerto origen, supongamos que fuese el 5015/tcp).
* El paquete viaja por Internet saltando de router a router hasta que llega a su destino
* El equipo 76.74.254.126 recibe una petición desde la dirección 80.58.1.14 y la contesta, por lo que el paquete de vuelta llevará ahora dirección IP origen 76.74.254.126, dirección IP destino 80.58.1.14, puerto origen 80/tcp y puerto destino 5015/tcp.
* La contestación del servidor web de google.com llega a la interfaz externa del dispositivo de NAT, que consulta las tablas de NAT y comprueba (gracias al puerto origen) que corresponde con una petición realizada desde el equipo 192.168.3.14, por lo que modifica la dirección IP destino por ésta y se lo envía directamente.

### IP masquerading

Este tipo de NAT normalmente es sinónimo de SNAT, pero iptables distingue dos casos:

* SNAT: Cuando la dirección IP pública que sustituye a la IP origen es estática (SNAT también significa Static NAT).
* MASQUERADE: Cuando la dirección IP pública que sustituye a la IP origen es dinámica, caso bastante habitual en conexiones a Internet domésticas.

### Destination NAT o port forwarding

Este tipo de NAT se utiliza cuando tenemos algún servidor en una máquina detrás del dispositivo de NAT. En este caso será un equipo externo el que inicie la conexión, ya que solicitará un determinado servicio y el dispositivo de NAT, en este caso, debe modificar la dirección IP destino. Veamos paso a paso cuál sería la situación.

* Un equipo cualquiera de Internet, con dirección IP pública 150.212.23.6 desea conectarse por ssh (22/tcp) al equipo estudio.mired.com
* Realiza una consulta DNS y obtiene como respuesta que estudio.mired.com tiene la dirección IP 85.136.14.7
* Establece la conexión (supongamos puerto origen 23014/tcp) con el equipo 85.136.14.7, que resulta ser un dispositivo de NAT que no tiene ningún servicio ssh escuchando en el puerto 22/tcp, pero que tiene una regla de DNAT para que todo lo que llegue a ese puerto se lo envíe a un equipo de su red local (supongamos que fuese el 10.0.0.2), por lo que cambia la dirección IP destino (85.136.14.7) por la 10.0.0.2 y lo registra en sus tablas de NAT.
* Al equipo 10.0.0.2 llega un solicitud al puerto 22/tcp y la respuesta tiene las siguientes características: IP origen 10.0.0.2, puerto origen 22/tcp, IP destino 150.212.23.6 y puerto destino 23014/tcp.
* El dispositivo de NAT cambia ahora la dirección IP origen por su dirección IP pública (85.136.14.7) y el paquete llega de vuelta a su destino.

### PAT (Port Address translation)

Modifica específicamente el puerto (origen o destino) en lugar de la dirección IP. Por ejemplo si queremos reenviar todas las peticiones web que lleguen al puerto 80/tcp al mismo equipo pero al puerto 8080/tcp

### Instalacion

Iptables ya hasta instalado en la mayor parte de las distribuciones de GNU/LINUX, en caso contrario se puede instalar usando la siguiente linea de comando en la consola de linux:

sudo apt install iptables (para distribuciones basadas en debian)

### Tablas y cadenas

Hay tres tablas definidas en iptables que son filter, nat y mangle, que se encargan de diferentes tipos de procesos, aquí sólo trataremos de la tabla nat, por lo que cada vez que escribamos una instrucción comenzaremos con iptables -t nat, ademas se debe tener en cuenta que para realizar cualquier modificacion se debe estar en modo “super usuario”, esto se puede tener con el comando “sudo su” e introduciendo la contraseña de usuario.

La tabla nat está formada por tres cadenas:

* PREROUTING: Permite modificar paquetes entrantes antes de que se tome una decisión de enrutamiento.
* OUTPUT: Permite modificar paquetes generados por el propio equipo después de enrutarlos
* POSTROUTING: Permite modificar paquetes justo antes de que salgan del equipo.

Para cadena se especifican reglas, para las que es fundamental el orden, ya que cuando un paquete encuentra una regla que lo define, aplica esa regla y no lee las siguientes.

## Parámetros generales

### Listar reglas

Utilizamos el parámetro -L (normalmente se acompaña de -n para que los resultados se muestren de forma numérica y evitar consultas DNS)  
  
# iptables -t nat -L -n  
  
Chain PREROUTING (policy ACCEPT)  
target prot opt source destination  
  
Chain POSTROUTING (policy ACCEPT)  
target prot opt source destination  
  
Chain OUTPUT (policy ACCEPT)  
target prot opt source destination  
  
Que nos muestra las tres cadenas de la tabla nat y que en este momento no hay ninguna regla aplicada.

### Verbose

Para una salida más completa de iptables utilizamos el parámetro -v:  
  
# iptables -t nat -L PREROUTING -n -v  
  
Chain PREROUTING (policy ACCEPT 21 packets, 4133 bytes)  
pkts bytes target prot opt in out source destination  
  
que nos informa de los paquetes y bytes que “atraviesan” una cadena y en caso de que hubiese reglas, se contarían los paquetes y bytes a los que se ha aplicado cada una.

### Borrar contadores

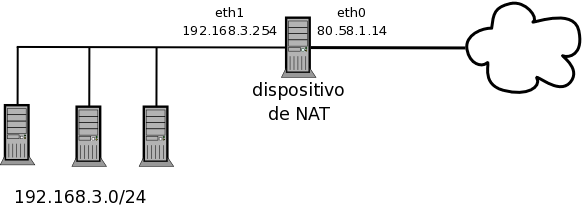
Si queremos poner a cero los contadores de paquete que se aplican en las cadenas de una tabla:  
  
iptables -t nat -Z

### Borrar todas las reglas de una cadena

Para borrar todas las reglas de una cadena se escribe:  
  
iptables -t nat -F OUTPUT  
  
Se puede no especificar ninguna cadena, con lo que se borran todas las reglas de todas las cadenas de una tabla:  
  
iptables -t nat -F  
  
Antes de empezar a utilizar iptables, una norma elemental es escribir:  
  
iptables -t nat -F  
iptables -t nat -Z  
  
Que borra todas las reglas anteriores y pone los contadores a cero.

## Ejemplo

Supongamos que tenemos una situación como la de la imagen:



Esquema de NAT

Vamos a ver los pasos que habría que dar para que todos los equipos de la red local tuviesen acceso a Internet y se pudiesen alojar servicios en cualquiera de ellos.

### Activación del bit de forward

En principio un equipo con GNU/Linux no permite que pasen paquetes de una interfaz de red a otra, para que se permita esto y por tanto pueda funcionar el equipo como router, o más concretamente en este caso como dispositivo de NAT, hay que activar (dar valor 1) lo que se denomina bit de forward:  
  
echo 1 > /proc/sys/net/ipv4/ip\_forward  
  
Esta activación se borra cuando se apaga el equipo, ya que el directorio /proc está en memoria. Para que dicha activación permanezca lo habitual es definirla en el fichero /etc/sysctl.conf, asegurándonos de que exista una línea como:  
  
net.ipv4.ip\_forward=1

## POSTROUTING

Todos los equipos de la red 192.168.3.0/24 están interconectados entre sí, pero en principio no tienen acceso a Internet puesto que sus direcciones IP son privadas y por tanto no son accesibles desde Internet (ningún equipo contestaría a sus peticiones). El equipo que tiene dos interfaces de red sí tiene acceso a Internet ya que la interfaz de red eth0 tiene una dirección IP pública, además pertenece a la red 192.168.3.0/24 ya que está conectado a través de la interfaz de red eth1 con dirección IP 192.168.3.254.

El equipo con dos interfaces de red puede funcionar como dispositivo de NAT (source NAT), aceptando paquetes provenientes del resto de equipos de la red 192.168.3.0/24 que entren por eth1 con destino a cualquier equipo de Internet. Tal como se describe en el ejemplo inicial de SNAT, el dispositivo de NAT debe cambiar la dirección IP origen, pero esto se hace justo antes de enviar el paquete a Internet y por tanto habrá que definirlo en la cadena POSTROUTING.

### Source NAT (estático) con iptables

La regla que hay que poner para que se haga SNAT de todos los equipos de la red 192.168.3.0/24 es tan simple como:  
  
iptables -t nat -A POSTROUTING -s 192.168.3.0/24 -o eth0 -j SNAT --to 80.58.1.14  
  
Explicación de los parámetros:

* -A POSTROUTING: Añade (Add) una regla a la cadena POSTROUTING
* -s 192.168.3.0/24: Se aplica a los paquetes que tengan como dirección origen (source) la 192.168.3.0/24
* -o eth0: Se aplica a los paquetes que salgan (out-interface) por eth0
* -j SNAT --to 80.58.1.14 (--to aquí es equivalente a --to-source): Cambia la dirección de origen por la 80.58.1.14

### Source NAT (dinámico) con iptables

Podríamos tener un caso similar al anterior, pero en el que la dirección IP pública del equipo que se conecta a Internet fuese dinámica, por lo que no la sabríamos a priori y no sería posible definirla en una regla como la anterior. En ese caso la regla de iptables a utilizar sería:  
  
iptables -t nat -A POSTROUTING -s 192.168.3.0/24 -o eth0 -j MASQUERADE  
  
Donde el único cambio se refiere a la acción (parámetro -j), en este caso es MASQUERADE, que cambia la dirección origen por la que tenga la interfaz de salida (eth0).

MASQUERADE podría funcionar también si la dirección IP de eth0 fuese estática, pero en ese caso se recomienda utilizar SNAT(estático).

## PREROUTING

(Todo lo que se explica en este punto no se hace con la seguridad en mente, sino simplemente para explicar algunas cosas que se pueden hacer con PREROUTING. En una implementación real, esto tendría que ir combinado con un cortafuegos y un esquema de red diferente).

Para realizar una conexión cliente-servidor entre dos equipos de Internet hay que especificar completamente lo que se denomina socket de Internet, que queda definido con lo siguiente:

* Protocolo (normalmente TCP o UDP)
* Dirección IP equipo cliente
* Puerto equipo cliente
* Dirección IP equipo servidor
* Puerto equipo servidor

Si volvemos a nuestro problema, el único equipo de la red local que es accesible desde Internet es el dispositivo de NAT a través de su dirección IP pública 80.58.1.14, ya que sería el único con el que un equipo de Internet podría establecer un socket y por tanto sería el único equipo de la red que podría alojar servicios. Todo esto cambia si utilizamos NAT, ya que en el equipo que tiene las dos interfaces de red podemos cambiar la dirección IP destino (DNAT) de una petición que llegue de Internet y mandarla a un equipo de la red local.

Supongamos que instalamos un servidor web en un equipo de la red local con dirección IP 192.168.3.2 y queremos que sea accesible desde Internet, tendremos que modificar las peticiones que lleguen al puerto 80/tcp del equipo que tiene la dirección IP pública y que cambie la dirección IP destino 80.58.1.14 por 192.168.3.2, esto se hace con la siguiente regla:  
  
iptables -t nat -A PREROUTING -p tcp --dport 80 -i eth0 -j DNAT --to 192.168.3.2  
  
Explicación de los parámetros:

* -A PREROUTING: Añade (Add) una regla a la cadena PREROUTING
* -p tcp: Especifica el (p)rotocolo de transporte (tcp en este caso)
* --dport 80 (equivalente a --destination-port 80): Puerto destino 80 (ligada al parámentro anterior)
* -i eth0: Especifica eth0 como interfaz de entrada (in-interface)
* -j DNAT --to 192.168.3.2 (--to aquí es equivalente a --to-destination): Cambia la dirección IP destino (inicialmente 80.58.1.14) a 192.168.3.2

Es lógico que haya que hacerlo en la cadena PREROUTING, porque las reglas de esta cadena se aplican antes de tomar la decisión de enrutamiento, así se tomará la decisión de encaminamiento con la nueva dirección IP destino.

Para otros servicios bastaría con poner el protocolo y puerto adecuados, aunque el caso del servicio ftp es más complicado y necesitaría una discusión mas detallada. La principal limitación de utilizar DNAT con una sola dirección pública es que no es posible poner más de un servicio en el mismo puerto, ya que sólo se puede hacer DNAT a un equipo de la red local (el socket debe estar totalmente determinado).

Hay algunos servicios que permiten utilizar puertos diferentes a los estándar, como por ejemplo http, ya que podemos acceder a un servidor web que esté en un puerto diferente al 80/tcp, simplemente especificándolo en el navegador. Como iptables nos permite no sólo modificar la dirección IP destino sino también el puerto destino, podríamos poner un segundo servidor web en el equipo 192.168.3.3 y añadir la siguiente regla:  
  
iptables -t nat -A PREROUTING -p tcp --dport 880 -i eth0 -j DNAT --to 192.168.3.3:80  
  
Donde hemos especificado el puerto destino 880 (que no es un puerto estándar para ningún servicio) y cambiamos tanto la IP destino como el puerto destino con --to 192.168.3.3:80.

Para acceder a ese servicio desde Internet tendremos que escribir en el navegador:  
  
[http://80.58.1.14:880](http://80.58.1.14:880/)

## Guardar todas las entradas: iptables-save e iptables-restore

## Cortafuegos

Uno de los mecanismos más extendidos para proteger una red de equipos que estén conectados a otra red (el caso más habitual es Internet), es colocar un dispositivo conectado a ambas redes que analiza el tráfico entre una y otra que, de acuerdo a una serie de criterios, permite o no pasar paquetes de un lado a otro. Este dispositivo conocido como cortafuegos o firewall puede implementar esta funcionalidad de forma nativa o mediante software sobre un sistema operativo genérico, en el primer caso el dispositivo se conoce como hardware firewall (cortafuegos por hardware) y en el segundo como software firewall (cortafuegos por software).

Hay una enorme variedad de cortafuegos por hardware: Juniper, Cisco y casi cualquier marca de componentes de redes ofrecen estos equipos, además este tipo de cortafuegos suele incluir otras funcionalidades como VPN, QoS, proxies, etc. En el caso de los cortafuegos por software destacan iptables del kérnel linux e ipfw de freebsd, en nuestro caso nos centramos en la configuración de un cortafuegos software con iptables.

### Tipos de cortafuegos

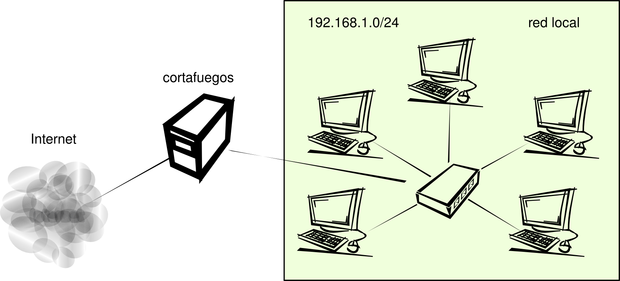
Una clasificación clásica de los cortafuegos distingue entre tres tipos o generaciones:

* Cortafuegos de filtrado de paquetes: Este tipo de cortafuegos analiza y filtra los paquetes que lo atraviesan en función de algunos parámetros a nivel de red, transporte o incluso enlace (por ejemplo direcciones IP origen o destino, puertos, direcciones MAC, etc.).
* Cortafuegos de estado: Consideran el estado del paquete en la comunicación, distinguiendo una nueva conexión de otra establecida por ejemplo.
* Cortafuegos de capa de aplicación: Analizan el contenido del paquete a nivel de aplicación, pudiendo hacer un filtrado más exhaustivo

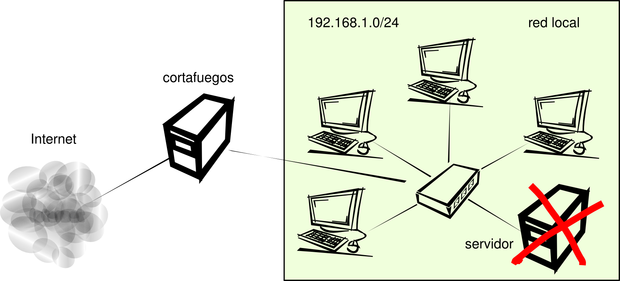
Estos tipos de cortafuegos no son excluyentes, las siguientes generaciones o características se van añadiendo sobre una configuración básica de un dispositivo para el filtrado de paquetes.

### Esquemas de red

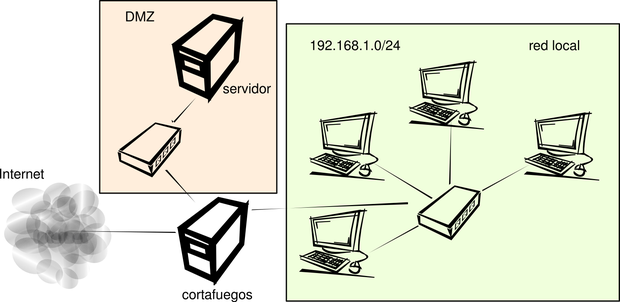
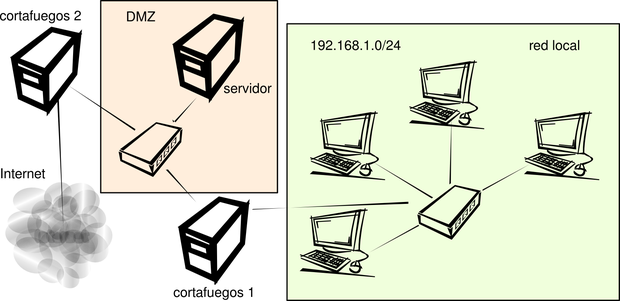
En primer lugar hay que decidir la ubicación en la red del cortafuegos, lo más frecuente es que el cortafuegos separe la red local de la otra red a la que queremos conectarnos (Internet por ejemplo), bien integrado en el mismo router que interconecta las dos redes o conectado en serie con éste.



El esquema anterior se corresponde con la configuración más sencilla, en la que no tenemos en nuestra red ningún servidor que tenga que ser accesible desde el exterior (servidor web por ejemplo), por lo que el cortafuegos se limita a permitir a los equipos de la red local acceso a determinados puertos. Esta configuración se puede complicar un poco si tenemos que incluir un servidor que ofrezca algún servicio al exterior:



Sin embargo, por motivos de seguridad, nunca debe ubicarse un servidor que sea accesible desde el exterior en nuestra red local, ya que al tratarse de un servidor expuesto, puede ser objetivo de ataques y conviene tenerlo separado del resto de equipos de nuestra red y obviamente de los demás servidores si los hubiera. Un esquema de red más adecuado sería ubicar el servidor o servidores accesibles desde Internet en un segmento de red específico, que se conoce como Zona Desmilitarizada o DMZ por analogía a la zona desmilitarizada militar, bien duplicando el número de cortafuegos o con un cortafuegos conectado a tres redes, lo que se conoce como cortafuegos de 3 patas:



Nota: Los esquemas se han realizado con la aplicación de gráficos vectoriales [inkscape](http://inkscape.org/) y con imágenes libres disponibles en <http://focaclipart.net23.net/hardware/> y <http://www.openclipart.org/>.

Las configuraciones reales pueden ser más complicadas todavía, fundamentalmente porque pueden incluir varios cortafuegos y varias redes, pero un equilibrio razonable entre un caso más o menos real y suficientemente sencillo para empezar con iptables puede ser un cortafuegos de 3 patas, que será el que utilicemos.

### La tabla filter de iptables

iptables incluye tres cuatro tablas inicialmente: filter, nat, mangle y raw, la tabla filter es la encargada del filtrado de paquetes y es la tabla por defecto, por lo que será igual poner iptables -t filter que iptables. Esta tabla contiene a su vez tres cadenas: INPUT, OUTPUT y FORWARD, las dos primeras se utilizan para paquetes que tienen como destino (INPUT) o como origen (OUTPUT) el propio cortafuegos, mientras que FORWARD es para paquetes que atraviesan el mismo, como por ejemplo cualquier paquete de la red local con destino un equipo de Internet o la respuesta del mismo.

### Política asumida o por defecto

Hay dos formas básicas de configurar un cortafuegos, la primera es permitir todo y restringir uno a uno los paquetes que queramos, es lo que se conoce como política por defecto ACCEPT. De forma opuesta, se prohibe todo y se va abriendo el cortafuegos a los paquetes que sea necesario, lo que se conoce como política por defecto DROP. Desde el punto de vista del control del tipo de conexiones que se van a permitir, parece mucho más razonable utilizar un cortafuegos con política por defecto DROP, y es lo que haremos aquí, por lo que ya podemos escribir nuestra primera regla de iptables en el cortafuegos:

iptables -P FORWARD DROP

¡Tenemos el cortafuegos perfecto! no se permite el paso de NINGÚN paquete

Cuando se utiliza un cortafuegos con política por defecto DROP, las reglas de iptables de la tabla filter suelen ir por parejas, ya que cada proceso que se permite incluye dos tipos de paquetes: las solicitudes y las respuestas. Estas parejas son de reglas de INPUT/OUTPUT para procesos que tienen como origen o destino el propio cortafuegos y FORWARD/FORWARD para procesos que tiene como origen y destino otros equipos.

Para que un cortafuegos sea efectivo, sobretodo si se trata de uno con política por defecto DROP, las reglas de los paquetes que se permiten debe ser lo más concretas posibles, es mucho mejor una regla como:

iptables -A FORWARD -i eth1 -s 10.0.0.1 -o eth0 -d 8.8.8.8 -p udp --dport 53 -j ACCEPT

que una regla como:

iptables -A FORWARD -p udp --dport 53 -j ACCEPT

Las dos consiguen lo mismo, pero la primera concreta mucho más el tipo de paquete que se permite y por tanto consigue mucho mayor control sobre lo que atraviesa el cortafuegos.

### Problema tipo

Vamos a plantear una situación concreta, sencilla pero con suficientes detalles como para comprender la configuración de un caso real. Vamos a configurar un cortafuegos de 3 patas con los siguientes condicionantes:

Cortafuegos

* Se conecta a Internet mediante una dirección IP estática
* Tiene tres interfaces de red eth0 (IP 40.12.1.14), que da acceso a Internet, eth1 (IP 192.168.1.1), que es la puerta de enlace de la red local y eth2 (IP 192.168.2.1), que es la puerta de enlace de la DMZ.
* Es servidor DHCP de la red local
* No debe ser accesible desde Internet
* Actúa como dispositivo NAT
* Se puede acceder a él por ssh desde la red local
* Puede hacer consultas DNS al servidor de la red local
* Responde a ping hecho desde la red local o la DMZ

DMZ

* Direccionamiento IP 192.168.2.0/24
* Conectada a la interfaz eth2 del cortafuegos
* Tiene un servidor web (http y https) y un servidor de correo (smtp, pop3s e imaps) en el mismo equipo (IP 192.168.2.2)

Red local

* Direccionamiento IP 192.168.1.0/24
* Conectada a la interfaz eth1 del cortafuegos
* Hay un servidor DNS (IP 192.168.1.2) que puede realizar consultas DNS al exterior.
* Los equipos de la red local deben tener acceso a todos los servicios ofrecidos por los equipos de la DMZ
* Los equipos de la red local pueden utilizar los servicios web (http y https) de cualquier servidor de Internet

### Resolución del problema

Establecemos la política por defecto:

iptables -P INPUT DROP

iptables -P OUTPUT DROP

iptables -P FORWARD DROP

Ahora bien se colocan las reglas NAT siguientes:

# Source NAT de la red local y la DMZ:

1- iptables -t nat -A POSTROUTING -o eth0 -s 192.168.1.0/24 -j SNAT --to 40.12.1.14

2- iptables -t nat -A POSTROUTING -o eth0 -s 192.168.2.0/24 -j SNAT --to 40.12.1.14

# DNAT de los servicios http,https,smtp,pop3s e imaps al servidor de la DMZ:

1- iptables -t nat -A PREROUTING -i eth0 -d 40.12.1.14 -p tcp --dport 80 -j DNAT --to 192.168.2.2

2- iptables -t nat -A PREROUTING -i eth0 -d 40.12.1.14 -p tcp --dport 443 -j DNAT --to 192.168.2.2

3- iptables -t nat -A PREROUTING -i eth0 -d 40.12.1.14 -p tcp --dport 25 -j DNAT --to 192.168.2.2

4- iptables -t nat -A PREROUTING -i eth0 -d 40.12.1.14 -p tcp --dport 993 -j DNAT --to 192.168.2.2

5- iptables -t nat -A PREROUTING -i eth0 -d 40.12.1.14 -p tcp --dport 995 -j DNAT --to 192.168.2.2

Con esto seguiríamos sin tener ningún tipo de conexión, ya que la política por defecto DROP seguiría descartando todos los paquetes que llegaran al cortafuegos. Vamos a incluir las reglas para que los equipos de la DMZ sean accesibles desde Internet:

# Cuando pasan por FORWARD los paquetes ya tienen como dirección IP destino la del

# servidor de la DMZ porque la ha cambiado DNAT:

# Aceptamos las peticiones al servidor web (http) y las respuestas de éste:

1- iptables -A FORWARD -i eth0 -s 0.0.0.0/0 -o eth2 -d 192.168.2.2 -p tcp --dport 80 -j ACCEPT

2- iptables -A FORWARD -o eth0 -d 0.0.0.0/0 -i eth2 -s 192.168.2.2 -p tcp --sport 80 -j ACCEPT

# Aceptamos las peticiones al servidor web (https) y las respuestas de éste:

1- iptables -A FORWARD -i eth0 -s 0.0.0.0/0 -o eth2 -d 192.168.2.2 -p tcp --dport 443 -j ACCEPT

2- iptables -A FORWARD -o eth0 -d 0.0.0.0/0 -i eth2 -s 192.168.2.2 -p tcp --sport 443 -j ACCEPT

# Aceptamos las peticiones al servidor smtp y las respuestas de éste:

1- iptables -A FORWARD -i eth0 -s 0.0.0.0/0 -o eth2 -d 192.168.2.2 -p tcp --dport 25 -j ACCEPT

2- iptables -A FORWARD -o eth0 -d 0.0.0.0/0 -i eth2 -s 192.168.2.2 -p tcp --sport 25 -j ACCEPT

# El servidor smtp también envía correo, por lo que se comporta como cliente:

1- iptables -A FORWARD -o eth0 -d 0.0.0.0/0 -i eth2 -s 192.168.2.2 -p tcp --dport 25 -j ACCEPT

2- iptables -A FORWARD -i eth0 -s 0.0.0.0/0 -o eth2 -d 192.168.2.2 -p tcp --sport 25 -j ACCEPT

# Aceptamos las peticiones al servidor pop3s y las respuestas de éste:

1- iptables -A FORWARD -i eth0 -s 0.0.0.0/0 -o eth2 -d 192.168.2.2 -p tcp --dport 995 -j ACCEPT

2- iptables -A FORWARD -o eth0 -d 0.0.0.0/0 -i eth2 -s 192.168.2.2 -p tcp --sport 995 -j ACCEPT

# Aceptamos las peticiones al servidor imaps y las respuestas de éste:

1- iptables -A FORWARD -i eth0 -s 0.0.0.0/0 -o eth2 -d 192.168.2.2 -p tcp --dport 993 -j ACCEPT

2- iptables -A FORWARD -o eth0 -d 0.0.0.0/0 -i eth2 -s 192.168.2.2 -p tcp --sport 993 -j ACCEPT

Para que los equipos de la DMZ puedan hacer resolución de nombres (imprescindible por ejemplo para enviar correo), les permitimos hacer consultas DNS:

1- iptables -A FORWARD -i eth2 -s 192.168.2.0/24 -o eth0 -d 0.0.0.0/0 -p udp --dport 53 -j ACCEPT

2- iptables -A FORWARD -o eth2 -d 192.168.2.0/24 -i eth0 -s 0.0.0.0/0 -p udp --sport 53 -j ACCEPT

3- iptables -A FORWARD -i eth2 -s 192.168.2.0/24 -o eth0 -d 0.0.0.0/0 -p tcp --dport 53 -j ACCEPT

4- iptables -A FORWARD -o eth2 -d 192.168.2.0/24 -i eth0 -s 0.0.0.0/0 -p tcp --sport 53 -j ACCEPT

Vamos a poner ahora las reglas necesarias para que los equipos de la red local puedan acceder a los servicios de la DMZ y a los servicios que se les permite de Internet:

# Aceptamos las peticiones al servidor web (http) y las respuestas de éste:

1- iptables -A FORWARD -i eth1 -s 192.168.1.0/24 -o eth2 -d 192.168.2.2 -p tcp --dport 80 -j ACCEPT

2- iptables -A FORWARD -o eth1 -d 192.168.1.0/24 -i eth2 -s 192.168.2.2 -p tcp --sport 80 -j ACCEPT

# Aceptamos las peticiones al servidor web (https) y las respuestas de éste:

1- iptables -A FORWARD -i eth1 -s 192.168.1.0/24 -o eth2 -d 192.168.2.2 -p tcp --dport 443 -j ACCEPT

2- iptables -A FORWARD -o eth1 -d 192.168.1.0/24 -i eth2 -s 192.168.2.2 -p tcp --sport 443 -j ACCEPT

# Aceptamos las peticiones al servidor smtp y las respuestas de éste:

1- iptables -A FORWARD -i eth1 -s 192.168.1.0/24 -o eth2 -d 192.168.2.2 -p tcp --dport 25 -j ACCEPT

2- iptables -A FORWARD -o eth1 -d 192.168.1.0/24 -i eth2 -s 192.168.2.2 -p tcp --sport 25 -j ACCEPT

# El servidor smtp también envía correo, por lo que se comporta como cliente:

1- iptables -A FORWARD -o eth1 -d 192.168.1.0/24 -i eth2 -s 192.168.2.2 -p tcp --dport 25 -j ACCEPT

2- iptables -A FORWARD -i eth1 -s 192.168.1.0/24 -o eth2 -d 192.168.2.2 -p tcp --sport 25 -j ACCEPT

# Aceptamos las peticiones al servidor pop3s y las respuestas de éste:

1- iptables -A FORWARD -i eth1 -s 192.168.1.0/24 -o eth2 -d 192.168.2.2 -p tcp --dport 995 -j ACCEPT

2- iptables -A FORWARD -o eth1 -d 192.168.1.0/24 -i eth2 -s 192.168.2.2 -p tcp --sport 995 -j ACCEPT

# Aceptamos las peticiones al servidor imaps y las respuestas de éste:

1- iptables -A FORWARD -i eth1 -s 192.168.1.0/24 -o eth2 -d 192.168.2.2 -p tcp --dport 993 -j ACCEPT

2- iptables -A FORWARD -o eth1 -d 192.168.1.0/24 -i eth2 -s 192.168.2.2 -p tcp --sport 993 -j ACCEPT

# Aceptamos las peticiones a los servidores web (http) de Internet y las respuetas:

1-iptables -A FORWARD -i eth1 -s 192.168.1.0/24 -o eth0 -d 0.0.0.0/0 -p tcp --dport 80 -j ACCEPT

2- iptables -A FORWARD -o eth1 -d 192.168.1.0/24 -i eth0 -s 0.0.0.0/0 -p tcp --sport 80 -j ACCEPT

# Aceptamos las peticiones a los servidores web (https) de Internet y las respuetas:

1- iptables -A FORWARD -i eth1 -s 192.168.1.0/24 -o eth0 -d 0.0.0.0/0 -p tcp --dport 443 -j ACCEPT

2- iptables -A FORWARD -o eth1 -d 192.168.1.0/24 -i eth0 -s 0.0.0.0/0 -p tcp --sport 443 -j ACCEPT

# Aceptamos las peticiones DNS del equipo 192.168.1.2:

1- iptables -A FORWARD -i eth1 -s 192.168.1.2 -o eth0 -d 0.0.0.0/0 -p udp --dport 53 -j ACCEPT

2- iptables -A FORWARD -o eth1 -d 192.168.1.2 -o eth0 -s 0.0.0.0/0 -p udp --sport 53 -j ACCEPT

3- iptables -A FORWARD -i eth1 -s 192.168.1.2 -o eth0 -d 0.0.0.0/0 -p tcp --dport 53 -j ACCEPT

4- iptables -A FORWARD -o eth1 -d 192.168.1.2 -i eth0 -s 0.0.0.0/0 -p tcp --sport 53 -j ACCEPT

Y para terminar incluimos las reglas de INPUT/OUTPUT necesarias en este problema:

# Se permiten las peticiones y respuestas DHCP:

1- iptables -A INPUT -i eth1 -p udp -s 0.0.0.0/0 -d 255.255.255.255 --dport 67 --sport 68 -j ACCEPT

2- iptables -A OUTPUT -o eth1 -s 192.168.1.1 -d 255.255.255.255 -p udp --sport 67 --dport 68 -j ACCEPT

# Se permite entrar por ssh desde la red local:

1- iptables -A INPUT -i eth1 -s 192.168.1.0/24 -d 192.168.1.1 -p tcp --dport 22 -j ACCEPT

2- iptables -A OUTPUT -o eth1 -d 192.168.1.0/24 -s 192.168.1.1 -p tcp --sport 22 -j ACCEPT

# Se permiten consultas DNS al servidor de la red local:

1- iptables -A OUTPUT -o eth1 -s 192.168.1.1 -d 192.168.1.2 -p udp --dport 53 -j ACCEPT

2- iptables -A INPUT -i eth1 -d 192.168.1.1 -s 192.168.1.2 -p udp --sport 53 -j ACCEPT

3- iptables -A OUTPUT -o eth1 -s 192.168.1.1 -d 192.168.1.2 -p tcp --dport 53 -j ACCEPT

4- iptables -A INPUT -i eth1 -d 192.168.1.1 -s 192.168.1.2 -p tcp --sport 53 -j ACCEPT

# Se permite ping desde la red local y la DMZ:

1- iptables -A INPUT -i eth1 -p icmp --icmp-type echo-request -j ACCEPT

2- iptables -A OUTPUT -o eth1 -p icmp --icmp-type echo-reply -j ACCEPT

3- iptables -A INPUT -i eth2 -p icmp --icmp-type echo-request -j ACCEPT

4- iptables -A OUTPUT -o eth2 -p icmp --icmp-type echo-reply -j ACCEPT