# Configuración de un router

Un router IP es un equipo que participa en distintas redes IP y que puede transferir (*forwarding*) paquetes entre esas redes.

La función de enrutamiento está integrada directamente en el núcleo de Linux pero, por defecto, no está activada.

El router hace de intermediario entre esas redes, y se puede utilizar para controlar y filtrar el tráfico entre esas redes (*firewall*), o para esconder las redes internas del exterior (*IP masquerading* o NAT, *Network Address Translation*).

## 1. Configuración de un servidor Linux como router

Para que un sistema Linux pueda enrutar mensajes entre dos redes IP en las que participa, basta con activar la función de enrutamiento en el núcleo. También puede ser necesario configurar rutas estáticas en la tabla de enrutamiento del sistema, para especificar hacia qué router hay que transferir los mensajes según las redes de destino.

Por defecto, Linux es un router estático IP, no intercambia información de enrutamiento con otros routers de la red, a través de un protocolo de enrutamiento (RIP, OSPF…). Para que pueda hacerlo, habría que instalar un programa de enrutamiento dinámico como Quagga, Bird o Zebra.

### a. Activación del enrutamiento

La activación del enrutamiento puede hacerse dinámicamente, usando el sistema de archivos virtual montado en `/proc`. El contenido del archivo `/proc/sys/net/ipv4/ip_forward`, `1` o `0`, determinará si el núcleo actual enruta en IPv4 o no.

*Ejemplo*

*Activación del enrutamiento:*

```
echo 1 > /proc/sys/net/ipv4/ip_forward
```

*Desactivación del enrutamiento:*

```
echo 0 > /proc/sys/net/ipv4/ip_forward
```

También se puede usar el comando `sysctl`, que permite modificar todos los parámetros dinámicos del núcleo accesible a partir de `/proc/sys`. La ventaja es que el comando comprueba que el parámetro que se va a escribir es válido.

*Ejemplo*

*Activación del enrutamiento:*

```
sysctl net.ipv4.ip_forward=1
```

*Desactivación del enrutamiento:*

```
sysctl net.ipv4.ip_forward=0
```

Para que este cambio se aplique cada vez que se inicie el sistema, hay que especificarlo en el archivo de configuración `/etc/sysctl.conf`.

*Sintaxis*

```
net.ipv4.ip_forward = 1
```

Para la configuración dinámica de IPv6, hay que usar el directorio `/proc/sys/net/ipv6/`. La activación o la desactivación del enrutamiento IPv6 se hace escribiendo `1` o `0` en el archivo `/proc/sys/net/ipv6/conf/all/forwarding`.

### b. La tabla de enrutamiento

Una vez que haya sido activado, el núcleo se basa en su tabla de enrutamiento para transferir los paquetes IP recibidos en una de las interfaces y que no están destinados a ninguna de las direcciones IP de router. Si el mensaje está destinado a una dirección que pertenece a una de las redes conocidas en su tabla de enrutamiento, transfiere el paquete al destinatario o a un router que sepa cómo enviar a esa red.

La tabla de enrutamiento por defecto está administrada por el núcleo, en función de las redes de sus diferentes tarjetas de interfaz de red, y la configuración de estas tarjetas durante el arranque del sistema. El núcleo puede también modificarla dinámicamente en función de los mensajes ICMP que hayan podido ser enviados por los routers vecinos.

La tabla de enrutamiento puede ser mostrada usando diferentes comandos (vistos en el capítulo sobre las redes), por ejemplo:

```
ip [-br] route show
route -n
netstat -rn
```

*Ejemplo*

```
ip -br route show
default via 192.168.0.254 dev enp38s0 proto dhcp metric 100
10.1.0.0/16 dev enp38s0 proto kernel scope link src 10.1.0.1
192.168.0.0/24 dev enp38s0 proto kernel scope link src 192.168.0.4 metric 100
```

*El sistema tiene una pasarela por defecto y una ruta para cada red en la que participa.*

### c. Gestión de las rutas estáticas

Para que el núcleo pueda enrutar paquetes hacia redes remotas, por otro sitio distinto que su pasarela por defecto, podemos añadir rutas en su tabla de enrutamiento.

Los dos comandos habituales para administrar la tabla de enrutamiento son `ip route` y `route`.

Si rutas estáticas tienen que ser permanentes, hay que declararlas en los archivos de configuración de arranque de redes, `/etc/sysconfig/network-scripts/route-*` para las distribuciones de tipo Red Hat, `/etc/network/interfaces` para las de tipo Debian.

### d. Modificar la tabla de enrutamiento: ip route

El comando `ip` con el objeto `route` permite configurar dinámicamente el enrutamiento de las tarjetas de interfaces de red.

*Sintaxis*

```
ip [ -6 ] [ Opciones ] [ route|r ] [ SubComando ] [ ArgSubComando ]
```

*Parámetros principales*

| | |
|---|---|
| `-6` | Para IPv6. |
| `Opciones` | Opciones. |
| `SubComando` | Subcomando que se ejecutará. |
| `ArgSubComando` | Argumentos del subcomando. |

*Descripción*

Este subcomando permite administrar (crear, mostrar, modificar y suprimir) la tabla de enrutamiento del sistema, en IPv4 y en IPv6.

Sin subcomando, mostrará la tabla de enrutamiento IPv4.

Para visualizar o modificar la tabla de enrutamiento IPv6, hay que usar la opción `-6`.

Los principales subcomandos y argumentos para estas rutas son:

Protocolo IPv4:

```
add default|IdRed via Dirección [ dev Interfaz ]
```

Añade como pasarela por defecto o como enrutador para la red especificada `IdRed` (en formato CIDR), la dirección `Dirección`, especificando opcionalmente qué tarjeta de interfaz de red hay utilizar.

```
show default|IdRed
```

Muestra la ruta o la pasarela por defecto.

```
change default|IdRed via Dirección [ dev Interfaz ]
```

Modifica la ruta o la pasarela por defecto.

```
delete default|IdRed
```

Suprime la ruta o la pasarela por defecto.

Protocolo IPv6:

Los comandos son iguales, especificando la opción `-6`. En general, la pasarela por defecto se detecta automáticamente en el enlace local y no hay necesidad de configurarla.

*Ejemplos*

*Configurar la pasarela por defecto:*

```
ip r show default
```

*Ninguna pasarela definida.*

*Definimos la pasarela por defecto:*

```
ip r add default via 192.168.0.254
ip r show default
default via 192.168.0.254 dev enp0s10
```

*Añadir una ruta estática:*

```
ip r add 10.0.1.0/24 via 192.168.0.12
ip r show
default via 192.168.0.254 dev enp0s10 onlink
10.0.1.0/24 via 192.168.0.12 dev enp0s10
```

### e. Modificar la tabla de enrutamiento: route

El comando `route` también permite administrar dinámicamente la tabla de enrutamiento IPv4 y IPv6.

Este comando ya no está instalado por defecto en las versiones recientes de las distribuciones de tipo Red Hat o Debian, forma parte del paquete `net-tools`.

*Sintaxis*

```
route [ Opciones ] [ Subcomando ] [ ArgSubcomando ]
```

*Parámetros principales*

| | |
|---|---|
| `Opciones` | Opciones. |
| `Subcomando` | Subcomando que se va a ejecutar. |
| `ArgSubcomando` | Argumentos del subcomando. |

### *Descripción*

Este comando permite gestionar (crear, mostrar, modificar y suprimir) la tabla de enrutamiento del sistema, en IPv4 y en IPv6.

Sin subcomando, muestra la tabla de enrutamiento IPv4.

Para visualizar o modificar la tabla de enrutamiento IPv6, hay que usar la opción `-6` o `-A inet6`.

Por defecto, el comando intenta usar los nombres de máquina o los nombres lógicos antes que las direcciones. La opción `-n` fuerza el uso de valores numéricos (direcciones IP).

En ese caso, la dirección correspondiente al destino por defecto es `0.0.0.0` (`[::]/0` en IPv6).

Los principales subcomandos y argumentos para administrar las rutas son:

Protocolo IPv4:

```
add default gw Direccion
```

Añade como pasarela por defecto la dirección `Direccion`.

```
del default
```

Suprime la pasarela por defecto.

```
add -net IdRed gw Direccion
```

Añade como pasarela hacia la red `IdRed` (en formato CIDR) la dirección `Direccion`.

Protocolo IPv6:

Los comandos son iguales, especificando la opción `-6`. En general, la pasarela por defecto se detecta automáticamente en el enlace local y no hay necesidad de configurarla.

*Ejemplos*

*Gestión de la pasarela por defecto IPv4:*

*Definición de la pasarela por defecto:*

```
route add default gw 192.168.0.251
route -n
Kernel IP routing table
Destination     Gateway         Mask            Flags Metric Ref    Use Iface
0.0.0.0         192.168.0.251   0.0.0.0         UG    0      0        0 enp0s10
169.254.0.0     0.0.0.0         255.255.0.0     U     1000   0        0 enp0s10
192.168.0.0     0.0.0.0         255.255.255.0   U     100    0        0 enp0s10
```

*Incorporación de una ruta estática:*

```
route add -net 10.2.0.0/16  gw 192.168.0.252
route -n
Kernel IP routing table
Destination     Gateway         Mask            Flags Metric Ref    Use Iface
0.0.0.0         192.168.0.251   0.0.0.0         UG    0      0        0 enp0s10
10.2.0.0        192.168.0.252   255.255.0.0     UG    0      0        0 enp0s10
169.254.0.0     0.0.0.0         255.255.0.0     U     1000   0        0 enp0s10
192.168.0.0     0.0.0.0         255.255.255.0   U     100    0        0 enp0s10
```

# 2. iptables

El núcleo Linux integra `netfilter`, componente de software de filtrado de paquetes de red. Esta capa se puede conectar con un software de usuario que permita definir reglas de filtrado y de modificación de paquetes de red, para implementar funcionalidades de firewall y de traducción de direcciones o de puertos. Los más usados son iptables y su sustituto nftables. Este último se puede usar nativamente o, de manera transparente, en modo compatible iptables.

iptables tiene que conocerse en el marco de la certificación LPIC-2.

Como la configuración de iptables es bastante delicada, hay programas que permiten implementarla más fácilmente, particularmente UFW (Uncomplicated Firewall) para la distribución Ubuntu y firewalld para las distribuciones de tipo Red Hat y Debian.

### a. Las tablas

iptables se basa en **tablas** asociadas a diferentes funciones. Estas tablas contienen **reglas** (*rules*), que se aplicarán en un orden definido formando **cadenas** (*chains*).

Las principales son `filter` para el filtrado de paquetes, `nat` (*network address translation*) para la traducción de direcciones entre una red privada y una red pública y `mangle` para la modificación de los paquetes entrantes o salientes.

La tabla `filter` es la tabla por defecto.

### b. Las cadenas

Una cadena `iptables` describe un conjunto ordenado de reglas que se aplican a un tipo de tráfico de red atendido por el núcleo del sistema local: paquetes dirigidos al sistema local (cadena `INPUT`), paquetes emitidos por el sistema local (cadena `OUTPUT`), o paquetes enrutados (cadena `FORWARD`).

Un paquete que atraviesa el router solo se verá afectado por las reglas de la cadena `FORWARD`. Las cadenas `INPUT` y `OUTPUT` están reservadas para los paquetes con destino o emitidos por el sistema local.

Hay otras dos cadenas, `PREROUTING` y `POSTROUTING`, utilizadas para las tablas `nat` y `mangle`, para aplicar un tratamiento a un paquete antes o después de una operación de enrutamiento.

En la sintaxis `iptables`, las cadenas siempre están especificadas en mayúsculas.

### c. Las acciones (targets)

Una regla está asociada a una **acción** (*target*), aplicada por el núcleo al paquete objeto de la regla. Las acciones principales son:

| | |
|---|---|
| `ACCEPT` | Aceptar el paquete. |
| `DROP` | Destruir silenciosamente el paquete. |
| `REJECT` | Destruir el paquete enviando un mensaje de rechazo ICMP al emisor. |
| `LOG` | Escribir un registro del paquete usando el daemon de registro del sistema y pasar a la siguiente regla. |

En la línea de comandos `iptables`, especificamos una acción usando la opción `-j NOMBRE_ACCIÓN`.

Cada cadena tiene una estrategia (policy) de acción por defecto, que se aplicará si ninguna regla de la cadena corresponde con el paquete analizado. Las dos acciones de estrategia posibles son `ACCEPT` o `DROP`.

En la línea de comandos `iptables`, especificamos la acción de estrategia de una cadena usando la opción `-P Nombre_CADENA NOMBRE_ACCIÓN`.

### d. Los criterios de selección

Una regla está asociada a criterios de selección que permiten determinar si un paquete encaja o no en una regla.

Los principales criterios de selección son los siguientes:

| | |
|---|---|
| `Protocolo` | Protocolo del paquete: `tcp`, `udp`, `icmp` o `all`. |
| `Dirección origen` | Dirección origen del paquete. Puede ser un nombre de host, de red, una dirección `IP CIDR` o simple. |
| `Dirección destino` | Dirección destino del paquete. Puede ser un nombre de host, de red, una dirección `IP CIDR` o simple. |
| `Interfaz origen` | Interfaz de red por donde se recibe el paquete. |
| `Interfaz destino` | Interface de red por donde se emitirá el paquete. |
| `Puerto origen` | Número de puerto origen o nombre del puerto declarado en `/etc/services`. |
| `Puerto destino` | Número de puerto destino o nombre del puerto declarado en `/etc/services`. |

Si el valor de un criterio empieza por el carácter `!`, está invertido el sentido de la prueba.

### e. El tratamiento de las reglas

Las reglas de una cadena se aplican una por una y en el orden de aparición en el archivo para un paquete en curso. Si una regla corresponde al paquete, la acción correspondiente será aplicada al paquete y el filtrado se terminará (excepto para la acción `LOG`, que no suspende el recorrido del filtrado). Si la regla no se puede aplicar a paquete en curso, se confrontará a la regla siguiente. Si ninguna regla corresponde, se aplicará la acción por defecto (acción de estrategia) de la cadena.

La opción `-L` del comando `iptables` muestra para cada cadena de la tabla `filter` la acción de estrategia y sus reglas, en orden.

*Sintaxis*

```
iptables -L
```

*Ejemplo*

*Reglas por defecto, sin configuración.*

```
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
```

*El núcleo no filtra ningún paquete.*

# 3. Administración de un firewall con iptables

El comando `iptables` permite configurar las tablas, cadenas y reglas que tienen que someterse al módulo `netfilter` del núcleo para controlar el tráfico de red del sistema.

La configuración del filtrado de los paquetes IPv6 se realiza gracias al comando `ip6tables`, según los mismos principios.

## a. Estrategias de cadenas

La estrategia (*policy*) de una cadena define la acción por defecto aplicada a un paquete si ninguna regla de la cadena encaja.

La opción `-P` del comando `iptables` permite especificar la estrategia de una cadena:

```
iptables -P Cadena Acción
```

Donde:

| | |
|---|---|
| `Cadena` | Cadena respectiva: `INPUT`, `OUTPUT` o `FORWARD`. |
| `Acción` | Acción por defecto: `DROP` o `ACCEPT`. |

*Ejemplo*

*Implementación de una estrategia restrictiva: por defecto, se prohíben todos los paquetes entrantes salientes o transferidos.*

```
iptables -P INPUT DROP
```

```
iptables -P OUTPUT DROP
iptables -P FORWARD DROP
```

## b. Creación de reglas

La creación de las reglas de una cadena se hace gracias al comando `iptables`, especificando la cadena en cuestión, el o los criterios de selección del paquete y la acción asociada a la regla:

*Sintaxis*

```
iptables -A Cadena [ -s IP_origen -d IP_destino -p Protocolo --sport Puerto
--dport Puerto ] -j Acción
```

Donde:

| | |
|---|---|
| `-A Cadena` | Cadena donde se añadirá la regla: `INPUT`, `OUTPUT` o `FORWARD`. |
| `-s IP_origen` | Dirección IP origen. |
| `-d IP_destino` | Dirección IP destino. |
| `-p Protocolo` | Protocolo: `tcp`, `udp`, `icmp` o `all`. |
| `--sport Puerto` | Puerto origen. |
| `--dport Puerto` | Puerto destino. |
| `-j Acción` | Acción asociada a la regla: `ACCEPT`, `DROP`, `REJECT` o `LOG`. |

Las reglas son añadidas en el orden de creación. Para añadir una regla dentro de una cadena, hay que usar la opción `-I` en lugar de `-A`:

```
-I Cadena [NúmRegla]
```

`NœmRegla` es el número de la regla en la cadena (`1` por defecto, primera posición).

*Ejemplo*

*Autorización del tráfico SSH, entrante y saliente:*

```
iptables -A OUTPUT -p tcp --sport 22  -j ACCEPT
iptables -A OUTPUT -p tcp --dport 22  -j ACCEPT
iptables -A INPUT -p tcp --dport 22 -j ACCEPT
iptables -A INPUT -p tcp --sport 22 -j ACCEPT
iptables -L
Chain INPUT (policy DROP)
target     prot opt source               destination
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:ssh
ACCEPT     tcp  --  anywhere             anywhere             tcp spt:ssh

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy DROP)
target     prot opt source               destination
ACCEPT     tcp  --  anywhere             anywhere             tcp spt:ssh
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:ssh
```

*Solo se autoriza el tráfico en el puerto bien conocido SSH (22), de emisión o de recepción. El enrutamiento no está filtrado.*

```
ping -c1 192.168.0.60
PING 192.168.0.60 (192.168.0.60) 56(84) bytes of data.
ping: sendmsg: Operación no permitida

^C
--- 192.168.0.60 ping statistics ---
1 packets transmitted, 0 received, 100% packet loss, time 0ms
```

*El envío de un paquete ICMP no está permitido.*

*Conexión en SSH desde una máquina remota:*

```
ssh debian10
root@debian10's password:
Linux debian10 4.19.117 #2 SMP Thu Apr 23 10:04:02 CEST 2020 x86_64
[...]
```

```
Last login: Mon Jun 15 15:52:40 2020 from 192.168.0.60
```

*Conexión en SSH hacia una máquina remota:*

```
ssh 192.168.0.60
Bienvenido al servidor CentOS8.
root@192.168.0.60's password:
Bienvenido al servidor CentOS 8
Last login: Mon Jun 15 15:22:39 2020 from 192.168.0.24
```

### c. Gestión de las reglas

Las reglas de una cadena se aplican en el orden de su creación. Están numeradas a partir de 1.

Distintas opciones del comando `iptables` permiten mostrar los números de las reglas y usarlos para suprimir o añadir una regla.

*Visualización de los números de las reglas*

```
iptables -L Cadena --line-numbers [ -n ]
```

La opcion `-n` suprime las tentativas de resolución de direcciones y de números como nombres, acelerando de esta manera la visualización.

*Inserción de una regla*

```
iptables -I Cadena [ NúmRegla ] [ -s IP_origen -d IP_destino
-p Protocolo --sport Puerto --dport Puerto ] -j Acción
```

Inserta la regla que tienen como posición `NœmRegla` (defecto `1`) en la cadena `Cadena`.

*Supresión de una règle*

```
iptables -D Cadena NúmRegla
```

Suprime la regla que tiene como posición `NœmRegla`.

*Supresión de todas las reglas*

```
iptables -F
```

Suprime todas las reglas de todas las cadenas, pero conserva la estrategia de cadenas actual.

*Ejemplo*

```
iptables -L --line-numbers
Chain INPUT (policy DROP)
num  target     prot opt source               destination
1    ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:ssh
2    ACCEPT     tcp  --  anywhere             anywhere             tcp spt:ssh

Chain FORWARD (policy ACCEPT)
num  target     prot opt source               destination

Chain OUTPUT (policy DROP)
num  target     prot opt source               destination
1    ACCEPT     tcp  --  anywhere             anywhere             tcp spt:ssh
2    ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:ssh
```

### d. Gestión de los paquetes de retorno

El control de una solicitud de conexión entrante puede hacerse gracias al número de puerto destino. Se trata a menudo de un número de puerto bien conocido asociado al

protocolo del servicio solicitado (HTTP, SSH, FTP, etc.). Sin embargo, el número de puertos del cliente remoto se atribuye de manera dinámica, por lo tanto es difícil autorizar al servidor a responder al cliente.

Para evitar estos problemas, podemos configurar el firewall para que autorice todos los paquetes, entrantes o salientes, relacionados con una conexión que ya se ha establecido.

*Autorización de los paquetes retorno*

```
iptables -A Cadena -m state --state ESTABLISHED,RELATED -j ACCEPT
```

La opción `-m state` permite especificar un filtro relacionado con el estado del paquete. Los estados aceptados son, `ESTABLISHED` y `RELATED`, representan respectivamente paquetes de una conexión establecida o vinculados a una conexión establecida (en particular, la conexión de transferencia de datos de una conexión FTP).

*Ejemplo*

```
iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
iptables -A OUTPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:ssh
ACCEPT     all  --  anywhere             anywhere             state RELATED,ESTABLISHED

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:ssh
ACCEPT     all  --  anywhere             anywhere             state RELATED,ESTABLISHED
```

### e. Ejemplo de configuración

Configuramos un router como firewall con `iptables`. La estrategia de filtrado por defecto es restrictiva: la acción por defecto es suprimir los paquetes que no correspondan a ninguna de las reglas.

El sistema es un router que protege a un servidor HTTP/HTTPS. Tiene que estar accesible en SSH. Además, tiene que autorizar los intercambios entre los clientes y los servidores DNS, en UDP.

```
iptables -A INPUT -p tcp --dport 22 -j ACCEPT
iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
iptables -A OUTPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
iptables -A FORWARD -s 192.168.0.0/24 -p tcp --dport 80 -j ACCEPT
iptables -A FORWARD -s 192.168.0.0/24 -p tcp --dport 443 -j ACCEPT
iptables -A FORWARD -s 192.168.0.0/24 -p udp --dport 53 -j ACCEPT
iptables -P INPUT DROP
iptables -P OUTPUT DROP
iptables -P FORWARD DROP
iptables -L
Chain INPUT (policy DROP)
target     prot opt source               destination
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:ssh
ACCEPT     all  --  anywhere             anywhere             state RELATED,ESTABLISHED

Chain FORWARD (policy DROP)
target     prot opt source               destination
ACCEPT     tcp  --  192.168.0.0/24       anywhere             tcp dpt:http
ACCEPT     tcp  --  192.168.0.0/24       anywhere             tcp dpt:https
ACCEPT     udp  --  192.168.0.0/24       anywhere             udp dpt:domain

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

## 4. Gestión de NAT (Network Address Translation)

Los rangos de direcciones IP siguientes están reservados para un uso interno, los routers

de Internet no pueden transferir paquetes vinculados con ellos:

- `10.0.0.0 - 10.255.255.254`
- `172.16.0.0 - 172.31.255.254`
- `192.168.0.0 - 192.168.255.254`

Esta restricción permite separar eficazmente las redes internas de las organizaciones de las redes públicas externas.

Sin embargo, la mayoría de las veces, los hosts de las redes internas tienen que poder acceder a servidores públicos externos. Para resolver este problema, se usa la técnica de traducción de direcciones NAT (*Network Address Translation*).

En IPv6, las direcciones privadas son administradas por un tipo particular de direcciones, las direcciones de enlace local (*Link Local Addresses*), generalmente atribuidas automáticamente.

### a. Principio de la traducción de direcciones NAT

La traducción de direcciones NAT consiste en modificar el encabezado IP de un paquete que pasa de una red pública hacia una red privada y viceversa. El router que efectúa la traducción reemplaza la dirección privada con una dirección pública, y viceversa, lo que permite que un host de la red privada pueda comunicarse con un host de la red pública.

### b. Configuración NAT de un router iptables

`iptables` gestiona la configuración de traducción NAT en la tabla específica nat. La cadenas gestionadas en esta tabla son `PREROUTING`, `POSTROUTING` y `OUTPUT`, para especificar los paquetes que se tienen que modificar antes del enrutamiento o directamente en salida de la máquina.

Para administrar esta tabla, hay que usar la opción `-t nat` del comando `iptables`.

### c. Conexión de una red privada a una red pública

Para configurar la traducción de direcciones entre una red privada y una red pública (la más frecuente), la dirección IP origen de los hosts de la red privada se reemplaza con una dirección pública gracias al comando siguiente:

```
iptables -t nat -A POSTROUTING -o Interf -j MASQUERADE
```

Donde:

| | |
|---|---|
| `-t nat` | Cadena de la tabla `nat`. |
| `-A POSTROUTING` | Cadena en la salida de enrutamiento. |
| `-o Interf` | Interfaz de red usada para emitir el paquete. |
| `-j MASQUERADE` | Acción que se aplicará: `MASQUERADE` (dirección pública dinámica). |

## 5. Respaldos y restauración de las reglas de filtrado

La sintaxis de definición de las cadenas de filtrado es compleja y delicada de implementar, sobre todo porque el orden de las reglas en una cadena es importante.

Los comandos `iptables-save` y `iptables-restore` permiten respaldar en un archivo el conjunto de la configuración actual, y volver a cargarla a continuación a partir de ese archivo.

En la mayoría de las distribuciones, se puede especificar ese archivo para que sea aplicado durante el arranque del sistema. De esta manera las reglas de filtrado siempre

son aplicadas.

Para las distribuciones de tipo Debian, hay que instalar el paquete `iptables-persistent` y guardar las reglas en el archivo `/etc/iptables/rules.v4`.

Para la distribución de tipo Red Hat antiguas (antes de Red Hat 7), el archivo es `/etc/sysconfig/iptables`. Las distribuciones recientes ya no utilizan `iptables`, sino `firewalld` o `nftables`.

### a. Respaldo de las reglas actuales

El comando `iptables-save` muestra el conjunto de la configuración en curso de las tablas `iptables`. Podemos redirigir la visualización hacia un archivo, o especificarlo con la opción `-f Camino Archivo`.

*Ejemplo*

```
iptables-save -f rules.txt
head rules.txt
# Generated by xtables-save v1.8.2 on Mon Jun 15 18:51:18 2020
*filter
:INPUT ACCEPT [995:177418]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [948:77027]
-A INPUT -p tcp -m tcp --dport 22 -j ACCEPT
-A INPUT -p tcp -m tcp --sport 22 -j ACCEPT
-A OUTPUT -p tcp -m tcp --sport 22 -j ACCEPT
-A OUTPUT -p tcp -m tcp --dport 22 -j ACCEPT
COMMIT
```

*Las reglas se guardan con la sintaxis `iptables`.*

### b. Restauración de las reglas de filtrado

El comando `iptables-restore` recarga el conjunto de la configuración `iptables` guardada anteriormente en un archivo, leído en la entrada estándar o especificado en la línea de comandos.

*Ejemplo*

```
iptables-restore rules.txt
```

*Las reglas guardadas en el archivo se han restaurado.*