Cortafuegos (Firewalls) en Linux con iptables

Sistemas Telemáticos

Departamento de Sistemas Telemáticos y Computación (GSyC)

Abril de 2012

Netfilter - iptables

- Netfilter[1] es un framework de Linux que permite interceptar y modificar paquetes IP.
- iptables es una herramienta de Netfilter que permite al administrador la definición de conjuntos de reglas aplicables a los paquetes IP que entran y/o salen de una máquina para realizar las siguientes operaciones:
  - Filtrado de paquetes *(packet filtering)*.
  - Seguimiento de conexiones *(connection tracking)*.
  - Traducción de direcciones IP y puertos (NAT, *Network Address Translation*).
- Hay 3 conceptos básicos en iptables:
  - **reglas**
  - **cadenas**
  - **tablas**

[1] http://www.netfilter.org

Reglas

- Una regla de `iptables` especifica una **condición** y una **acción**:
  - **condición**: características que debe cumplir un paquete para que la regla le sea aplicable. Ejemplos de condiciones:

    `-p tcp --dport 80`: el protocolo es TCP y el puerto destino es 80

    `-s 13.1.2.0/24`: la dirección de origen es de la subred 13.1.2.0/24.
  - **acción**: indica lo que se hace con el paquete si cumple la condición de la regla. Ejemplos de acciones:

    `ACCEPT`: el paquete se acepta
    `DROP`: el paquete se descarta
    `SNAT --to-source 13.1.2.1`: se cambia la IP origen del paquete
- Las reglas se agrupan en listas de reglas, llamadas cadenas.
- Las cadenas se agrupan en tablas.

Cadenas (I)

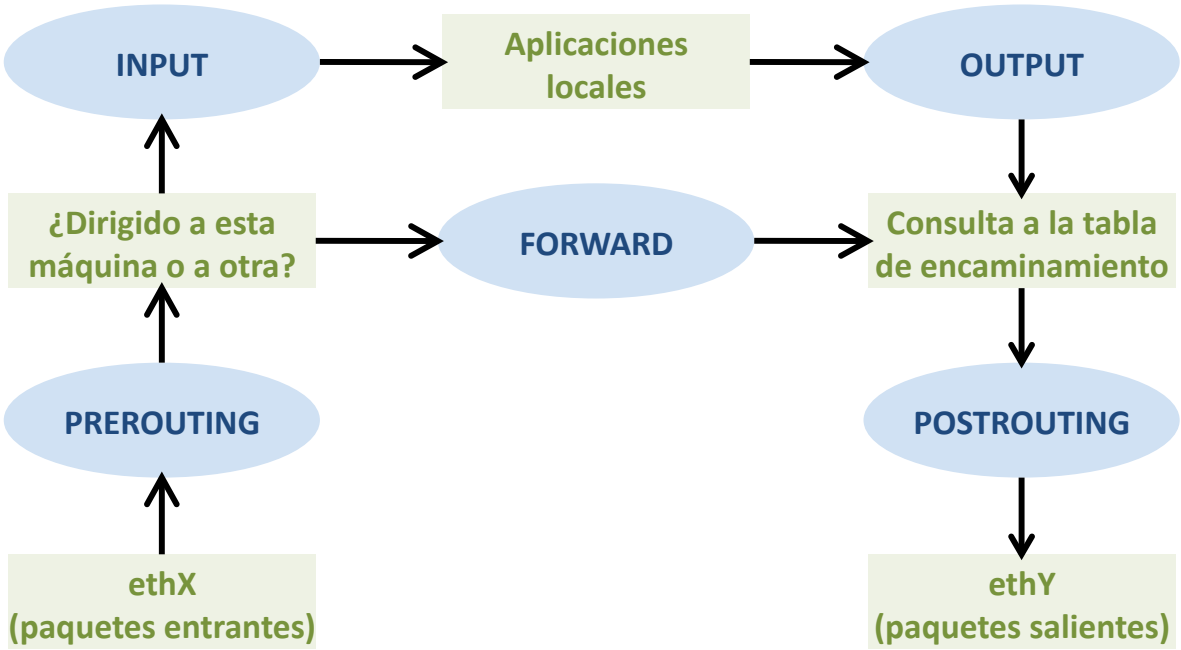
- Una cadena es una **lista ordenada de reglas**.
- Para cada paquete se va comprobando si se le aplica cada regla de la cadena (es decir, si cumple la **condición**):
  - Si una regla NO se aplica a un paquete, se pasa a la siguiente regla de la cadena.
  - Si una regla SÍ se aplica a un paquete, se ejecuta la **acción** definida en dicha regla y, (salvo excepciones) ya no se comprobarán más reglas de la cadena.
- Una cadena puede tener definida una política, que es la **acción por defecto** para la cadena. La política predefinida para todas las cadenas es `ACCEPT` (es decir, aceptar el paquete).
- Cuando para un paquete NO se aplica NINGUNA de las reglas de la cadena, se ejecuta para él la política de la cadena (si dicha cadena la tiene).

Cadenas (II)

- Las reglas tienen una posición determinada (**número de regla**) dentro de la cadena. A la hora de poner una nueva regla en una cadena, hay tres posibilidades:
  - **añadir** la regla al fina de la cadena, detrás de las ya existentes
  - **reemplazar** en una posición a otra regla ya existente
  - **insertar** la regla en una posición ya existente, desplazando un lugar a las reglas existentes desde esa posición en adelante.

Cadenas (III): Tipos de cadenas

- Existen diferentes tipos de cadenas:
  - Predefinidas:
    `PREROUTING`, `INPUT`, `FORWARD`, `OUTPUT`, `POSTROUTING`
  - Definidas por el usuario. Dichas cadenas no tienen **política** predefinida.
- Un paquete al llegar a una máquina se le aplican las reglas de las cadenas predeterminadas en distintos momentos según el siguiente esquema:



Cadenas (IV): Cadenas predefinidas

- Cadena `PREROUTING`:
  - Reglas que se aplican a los paquetes que llegan a la máquina. Esta cadena se ejecuta antes de comprobar si el paquete es para la propia máquina o hay que reenviarlo.
- Cadena `INPUT`:
  - Reglas que se aplican a los paquetes destinados a la propia máquina. Esta cadena se ejecuta justo antes de entregarlos a la aplicación local.
- Cadena `FORWARD`:
  - Reglas que se aplican a los paquetes que han llegado a la máquina pero van destinados a otra y hay que reenviarlos. Esta cadena se ejecuta antes de consultar la tabla de encaminamiento.
- Cadena `OUTPUT`:
  - Reglas que se aplican a los paquetes creados por la propia máquina. Esta cadena se ejecuta justo después de que la aplicación le pase los datos a enviar al *kernel* del sistema operativo y antes de consultar la tabla de encaminamiento.
- Cadena `POSTROUTING`:
  - Reglas que se aplican a los paquetes que salen de la máquina, tanto los creados por ella como los que se reenvían. Esta cadena se ejecuta después de consultar la tabla de encaminamiento.

Tablas (I)

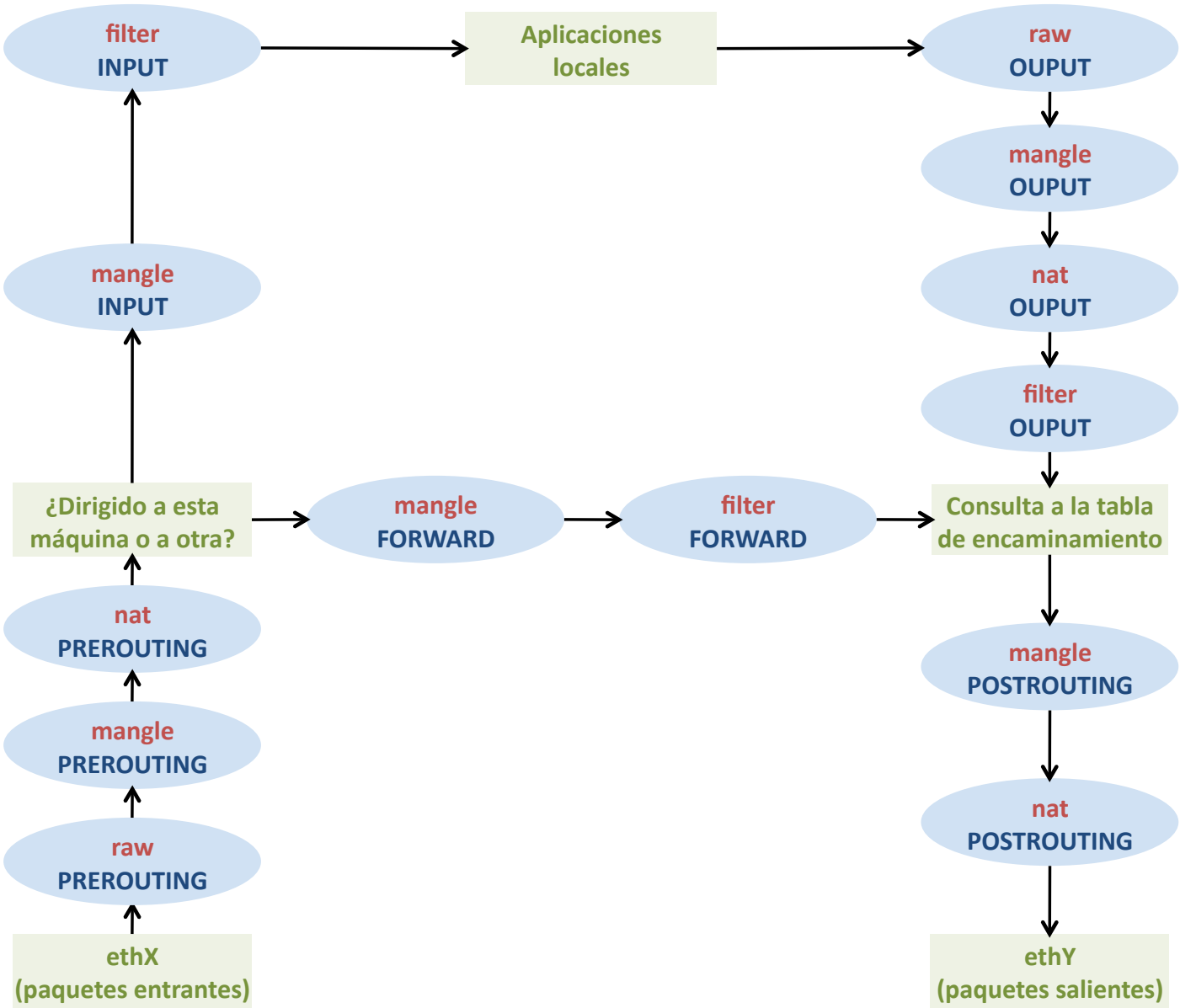
- Una tabla de `iptables` contiene un conjunto de cadenas, tanto predefinidas como de usuario.
- Una tabla concreta engloba las reglas (agrupadas en cadenas) relacionadas con un tipo de procesamiento de los paquetes.
- `Netfilter` define las siguientes tablas:
  - `filter`: engloba las reglas de filtrado de paquetes, es decir, de las que deciden que un paquete continúe su camino o sea descartado.
  - `nat`: engloba las reglas de modificación de direcciones IP y puertos de los paquetes
  - `mangle`: engloba las reglas de modificación de algunos campos de las cabeceras del paquete. Ejemplo: TTL
  - `raw`: engloba las reglas que permiten marcar excepciones al seguimiento que hace el *kernel* de las "conexiones"[2] de la máquina.

[2] "conexiones" en sentido amplio: no solo conexiones TCP, sino también tráfico UDP enviado/recibido para las mismas direcciones y puertos, tráfico ICMP de petición/respuesta de eco...

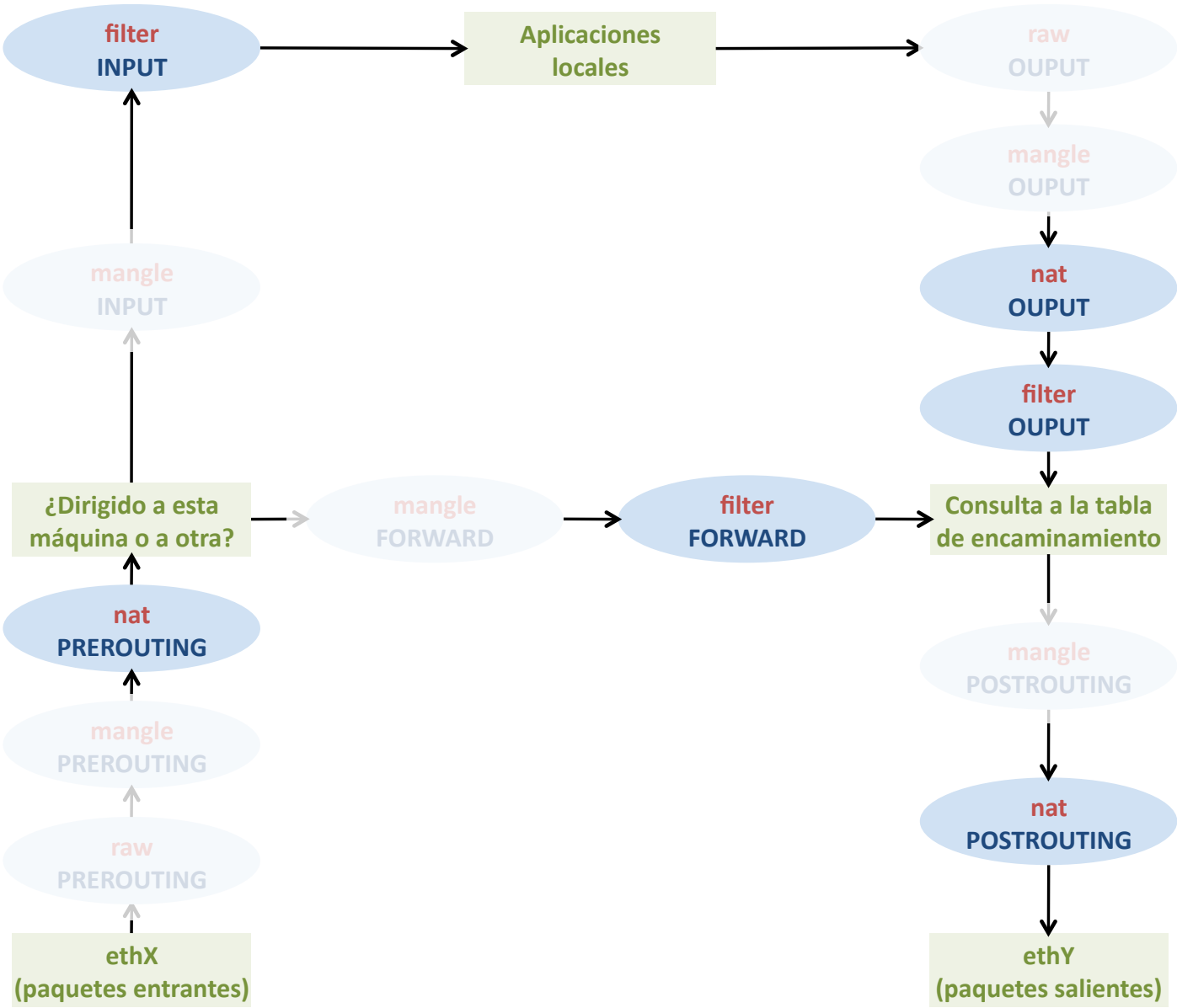
Tablas (II): Cadenas predefinidas de cada tabla

- La tabla `filter` incluye las cadenas:
  - `FORWARD`
  - `INPUT`
  - `OUTPUT`
- La tabla `nat` incluye las cadenas:
  - `PREROUTING`
  - `OUTPUT`
  - `POSTROUTING`
- La tabla `mangle` incluye las cadenas:
  - `PREROUTING`
  - `FORWARD`
  - `INPUT`
  - `OUTPUT`
  - `POSTROUTING`
- La tabla `raw` incluye las cadenas:
  - `PREROUTING`
  - `OUTPUT`

Movimiento de los paquetes por tablas y cadenas



Movimiento de los paquetes (versión simplificada)



iptables: comandos

`iptables [-t <tabla>] <comando> [<condición>] [<acción>]`

Si no se especifica una tabla se utilizará por defecto la tabla `filter`.

- Comandos más utilizados:

`iptables [-t <tabla>] -L [<cadena>] [-v]` lista las reglas definidas en una `cadena` de una `tabla`. Si se omite la cadena el comando actúa sobre todas. Con `-v` se mostrará también el número de paquetes y bytes que han cumplido la condición de cada regla.
`iptables [-t <tabla>] -F [<cadena>]` borra la lista de reglas que hay en una `cadena` de una `tabla`. Si se omite la cadena el comando actúa sobre todas.
`iptables [-t <tabla>] -Z [<cadena>]` reinicia los contadores de una `cadena` de una `tabla`: número de paquetes y bytes que cumplen las condiciones de sus reglas. Si se omite la cadena el comando actúa sobre todas.
`iptables [-t <tabla>] -N [<cadena-usuario>]` crea en una tabla una nueva cadena definida por el usuario.
`iptables [-t <tabla>] -P <cadena> <política>` establece la política por defecto para una cadena predefinida de una tabla, donde la política puede ser `DROP` o `ACCEPT`.
`iptables [-t <tabla>] -A <cadena> <condición> <acción>` añade una regla al final de las reglas que tiene definidas una cadena de una tabla. La regla queda definida por la ejecución de una `acción` si un paquete cumple una `condición`.

`iptables [-t <tabla>] -D <cadena> <condición> <acción>`
`iptables [-t <tabla>] -D <cadena> <numregla>`
borra una regla de una cadena de una tabla dada su especificación o dado su número de regla.

`iptables [-t <tabla>] -R <cadena> <numregla> <condición> <acción>` reemplaza la regla número `numregla` de una cadena por una nueva regla.
`iptables [-t <tabla>] -I <cadena> <numregla> <condición> <acción>` inserta una regla en la posición `numregla` en una cadena de una tabla.

iptables: condiciones

- Condiciones:

Protocolo	`-p <protocolo>`
Dirección IP	`-s <dirIP[/máscara]>`: dirección origen `-d <dirIP[/máscara]>`: dirección destino
Puerto	`--sport <puerto|puertoInicio:puertoFin>`: puerto origen `--dport <puerto|puertoInicio:puertoFin>`: puerto destino
Interfaz	`-i <interfaz>`: interfaz de entrada `-o <interfaz>`: interfaz de salida
Estado de la conexión[3]	`-m state --state <estado>` situación de un paquete con respecto a la conexión a la que pertenece. Estado: `INVALID`: no pertenece a una conexión existente `ESTABLISHED`: es parte de una conexión existente con paquetes en ambos sentidos `NEW`: es parte de una nueva conexión que aún no está establecida `RELATED`: está relacionado con otra conexión ya existente Ejemplo: un mensaje ICMP de error
Flags TCP	`--syn`: segmento SYN
	`--tcp-flag <flagsAComprobar> <flagsQueDebenEstarActivados>` flags: `SYN, FIN, ACK, RST, PSH, URG, ALL, NONE` Ejemplo: `-p tcp --tcp-flags ALL SYN,ACK` (deben estar activados `SYN, ACK` y desactivados `FIN, RST, PSH, URG`)

- La negación de una condición se expresa anteponiendo el caracter `!` al valor de la condición. Ejemplos:

`-p tcp --sport ! 80`	protocolo TCP y puerto origen distinto del 80
`-p ! icmp`	protocolo distinto de icmp

[3] "conexión" en sentido amplio

iptables: acciones

La acción se especifica empezando con `-j`

Tabla `filter`	`-j ACCEPT` se acepta el paquete
	`-j DROP` se descarta el paquete
	`-j REJECT [--reject-with <tipo>]` se rechaza el paquete, informando al origen con un ICMP, se puede especificar el tipo de ICMP, por defecto `icmp-port-unreachable`
Tabla `nat`	`-j SNAT --to-source [<dirIP>][:<puerto>]` Realiza *Source NAT* sobre los paquetes salientes (es decir, se cambia dirección IP y/o puerto origen). Sólo se puede realizar en la cadena `POSTROUTING`. **NOTA: esta regla hace que también se cambie automáticamente la dirección de destino del tráfico entrante de respuesta al saliente de la misma "conexión".**
	`-j DNAT --to-destination [<dirIP>][:<puerto>]` Realiza *Destination NAT* sobre los paquetes entrantes (es decir, se cambia dirección IP y/o puerto destino). Sólo se puede realizar en la cadena `PREROUTING`. Esta regla sólo es necesaria para "abrir puertos", es decir, permitir tráfico entrante nuevo.
Tabla `mangle`	`-j TTL --ttl-set <valorTTL>` Modifica el valor de TTL
Todas las tablas	`-j LOG` se guarda información de ese paquete en el fichero de log y **se continúa con la siguiente regla de la cadena**
	`-j <cadena-de-usuario>` Salta a aplicar al paquete las reglas de una cadena definida por el usuario. Si termina esa cadena sin cumplirse la condición de ninguna de sus reglas, continuará en la cadena desde la que se saltó, por la regla siguiente a la que hizo la llamada.

Si en una regla **no se especifica ninguna acción** (no hay cláusula `-j`), si se cumple la condición se actualizan los contadores de paquetes y bytes para la regla, pero **se continúa aplicando la seiguiente regla de la cadena para ese paquete**.

Seguimiento de "conexiones"

- Las "conexiones" (en sentido amplio) de las comunicaciones TCP, UDP, ICMP que atraviesan una máquina se pueden monitorizar a través del módulo `conntrack` de `iptables`.
- Para visualizar dichas conexiones hay que mostrar el contenido del fichero `/proc/net/ip_conntrack`:

```
r1:~# cat /proc/net/ip_conntrack
tcp      6 117 SYN_SENT src=192.168.1.6 dst=192.168.1.9 sport=32775 \
     dport=22 [UNREPLIED] src=192.168.1.9 dst=192.168.1.6 sport=22 \
     dport=32775 use=2
```

Ejemplos de configuración de un *firewall* con `iptables`

- Borrar las reglas y reiniciar los contadores:

```
iptables -t filter -F
iptables -t filter -Z
```

- Definir las políticas por defecto: Descartar cualquier cosa salvo paquetes de salida:

```
iptables -t filter -P INPUT DROP
iptables -t filter -P FORWARD DROP
iptables -t filter -P OUTPUT ACCEPT
```

- Permitir el reenvío de todos los paquetes que se reciben en un router a través de una interfaz (eth0) para que se envíen a través de otra interfaz (eth1) (por ejemplo, permitir tráfico saliente de una organización):

```
iptables -t filter -A FORWARD -i eth0 -o eth1 -j ACCEPT
```

- Permitir el reenvío paquetes entrantes que pertenezcan a "conexiones" ya existentes:

```
iptables -t filter -A FORWARD -i eth1 -o eth0
         -m state --state RELATED,ESTABLISHED -j ACCEPT
```

- Permitir el paso de segmentos TCP de establecimiento de conexión de entrada dirigidos a una dirección IP de la red interna (10.0.0.10) y a un puerto (80).

```
iptables -t filter -A FORWARD -p tcp -d 10.0.0.10 --dport 80 --syn -j ACCEPT
```

- Permitir el paso de datagramas UDP de entrada dirigidos a una dirección IP de la red interna (10.0.0.10) y a un puerto (80).

```
iptables -t filter -A FORWARD -p udp -d 10.0.0.10 --dport 80 -j ACCEPT
```

Ejemplos de traducción de direcciones IP y puertos con iptables

- Borrar las reglas y reiniciar los contadores:

```
iptables -t nat -F
iptables -t nat -Z
```

- Modificar la dirección IP origen de los datagramas IP al salir de una red privada (10.0.0.0/24) a través de la interfaz de salida (eth0) de un router NAT. Todos los datagramas llevarán la dirección IP pública del router NAT (200.0.0.1):

```
iptables -t nat -A POSTROUTING -s 10.0.0.0/24 -o eth0
         -j SNAT --to-source 200.0.0.1
```

- Modificar la dirección IP destino y puerto destino de los segmentos TCP al entrar dentro de una red privada (10.0.0.0/24). Los segmentos van dirigidos inicialmente a la dirección IP del router NAT (200.0.0.1) y puerto 8080, recibiéndose en su interfaz (eth1). Antes de comprobar la tabla de encaminamiento (PREROUTING) se modificará su dirección IP destino a 10.0.0.10 y puerto destino 80.

```
iptables -t nat -A PREROUTING -p tcp -i eth1 -d 200.0.0.1
         --dport 8080 -j DNAT --to-destination 10.0.0.10:80
```

Configuración de un router NAT

- El comportamiento de un *router* NAT se implementa con `iptables` como una colección de reglas de filtrado (tabla `filter`) y reglas de traducción de direcciones y puertos (tabla `nat`).