Red frontera - Firewalls en Linux

Planificación y gestión de redes de ordenadores

Departamento de Teoría de la Señal y Comunicaciones y Sistemas Telemáticos y Computación (GSyC)

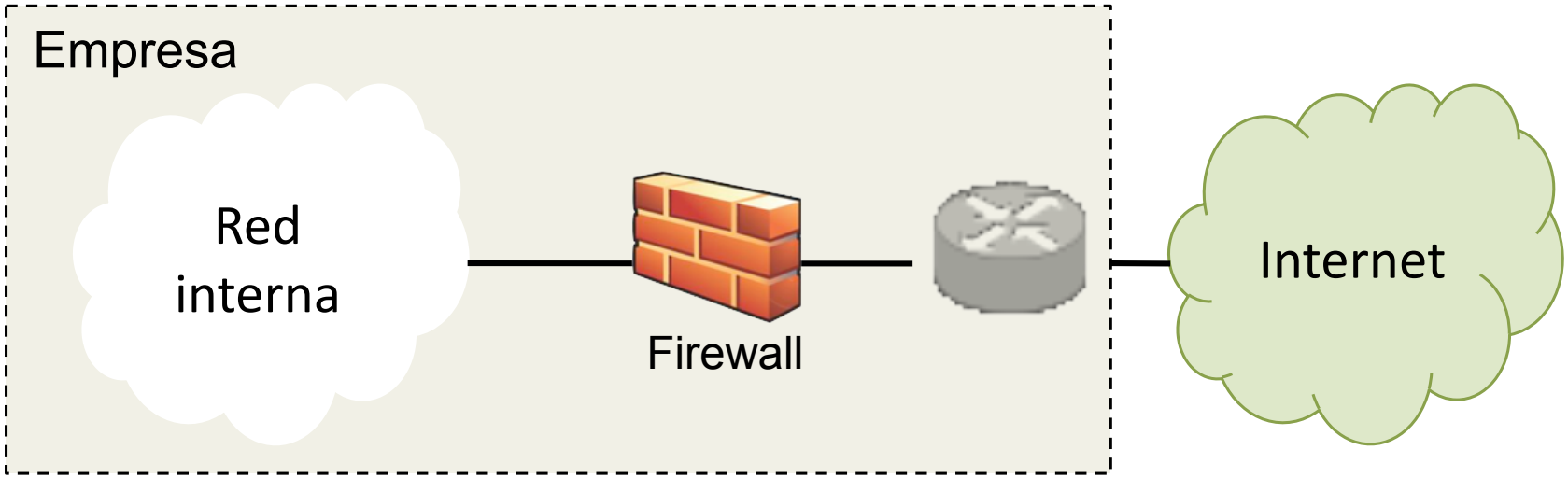
Septiembre de 2014

Contenidos

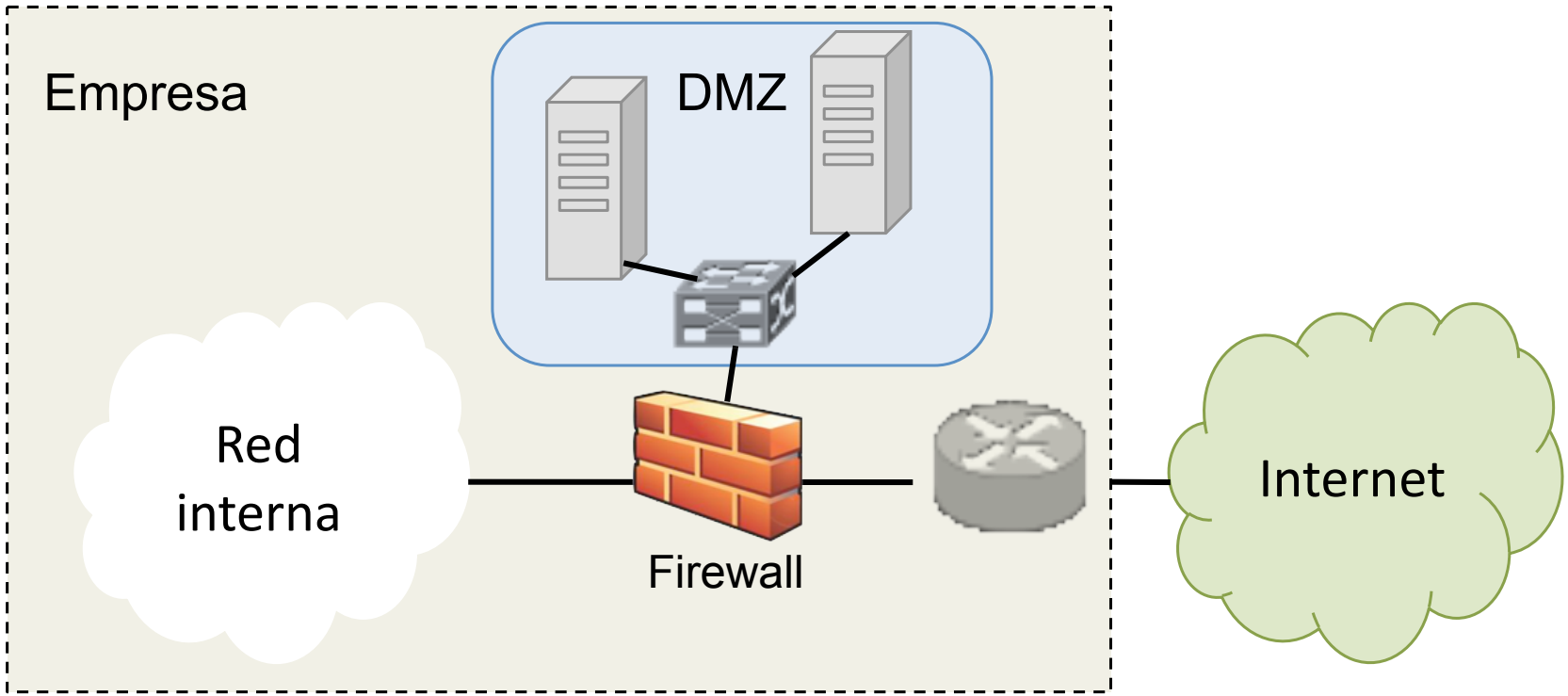
La red frontera

- La red frontera es la parte de la red que comunica la red interna de una empresa con otras redes externas.
- La seguridad en la red frontera es clave para proteger los equipos y servicios de la empresa de ataques externos. Para ello, las empresas instalan *firewalls* que permiten filtrar el tráfico y detectar posibles ataques maliciosos desde el exterior. Adicionalmente los *firewalls* permiten restringir el tráfico que sale de los equipos internos de la empresa.

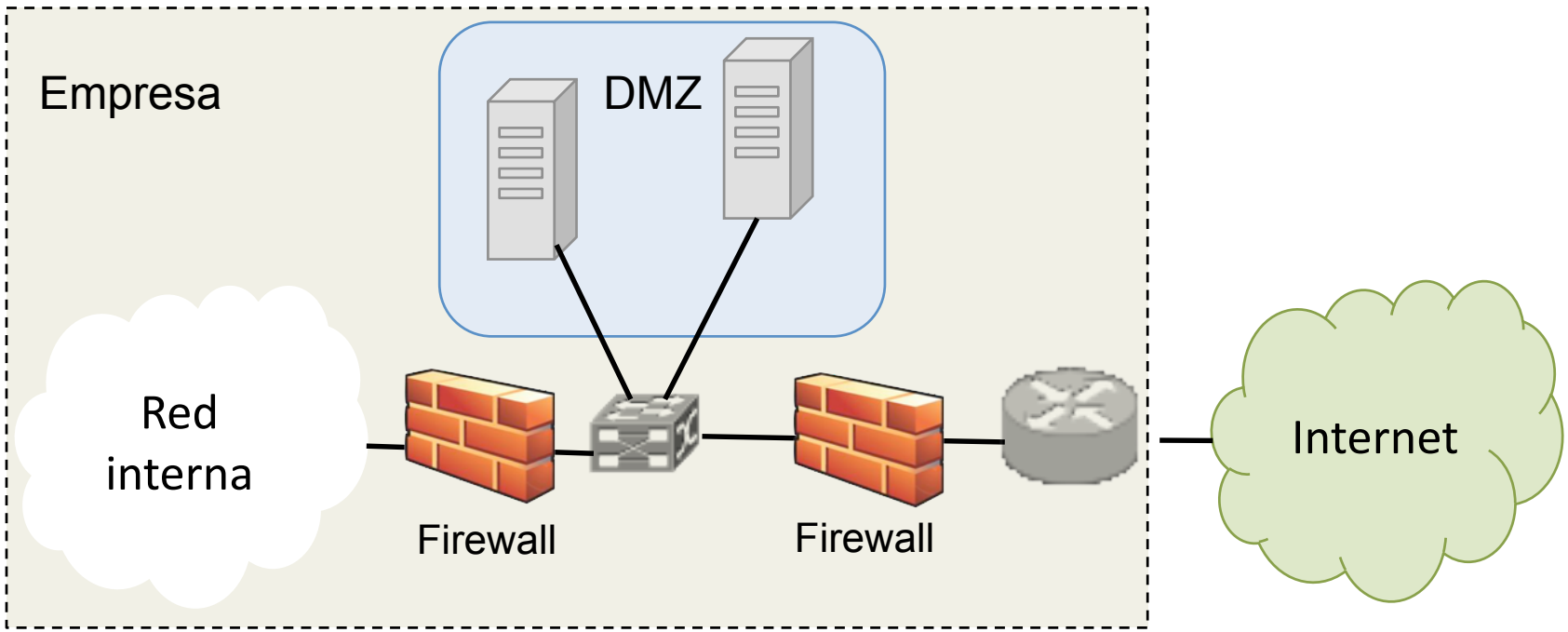
Un único firewall



Un único firewall con zona DMZ



Dos firewalls



Contenidos

Contenidos

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

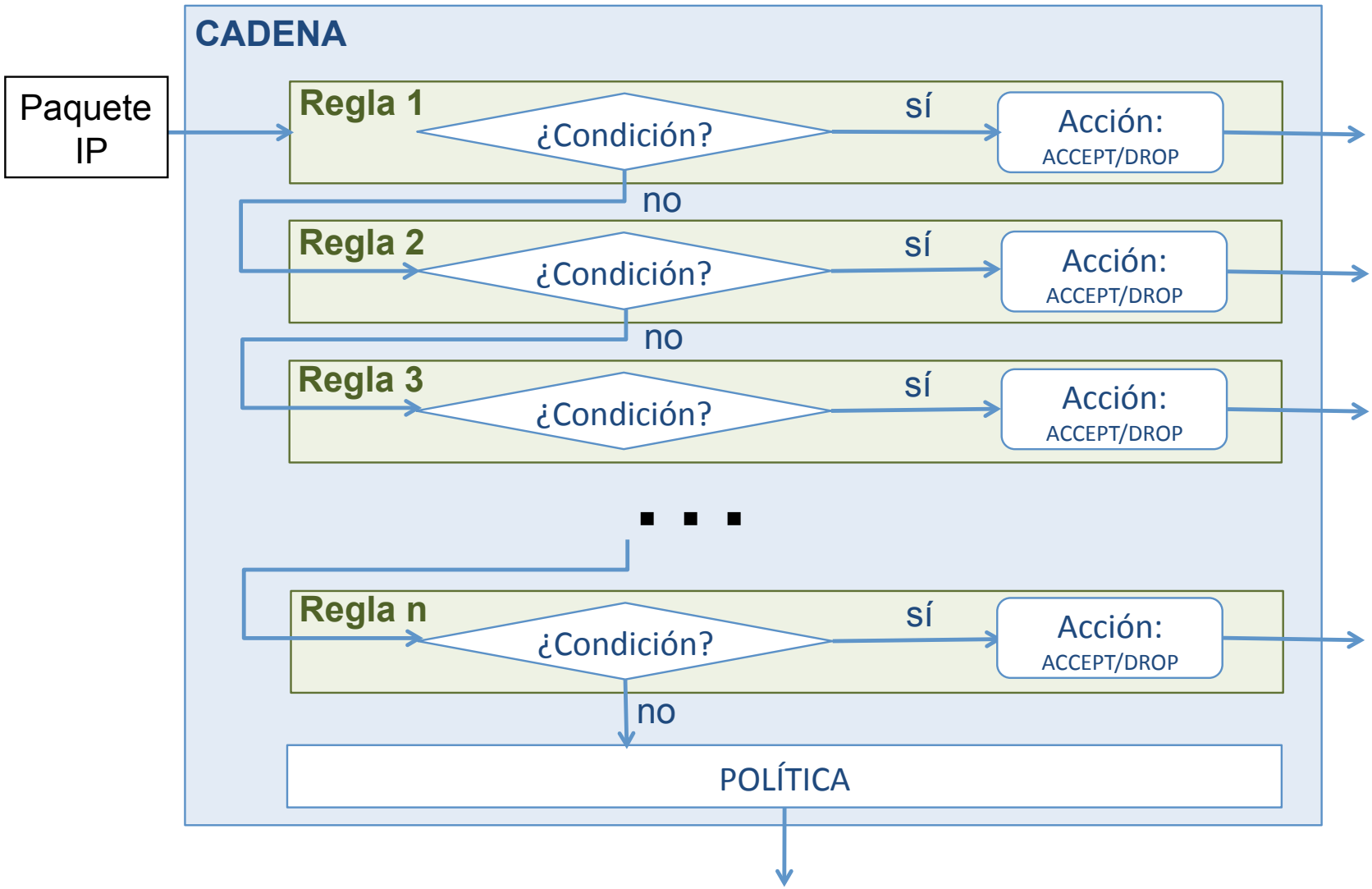
    `-s 13.1.2.0/24`: la dirección de origen es de la subred 13.1.2.0/24.
  - **acción**: indica lo que se hace con el paquete si cumple la condición de la regla. Ejemplos de acciones:

    `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. Dependiendo del tipo de acción:
    - El paquete abandona la comprobación del resto de las reglas y pasa a la siguiente cadena (acciones ej: ACCEPT, DROP)
    - El paquete continúa con la siguiente regla de la cadena (acción ej: LOG)
- Una cadena puede tener definida una política, que es la **acción por defecto** para la cadena. La política predefinida para todas las cadenas predefinidas 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)

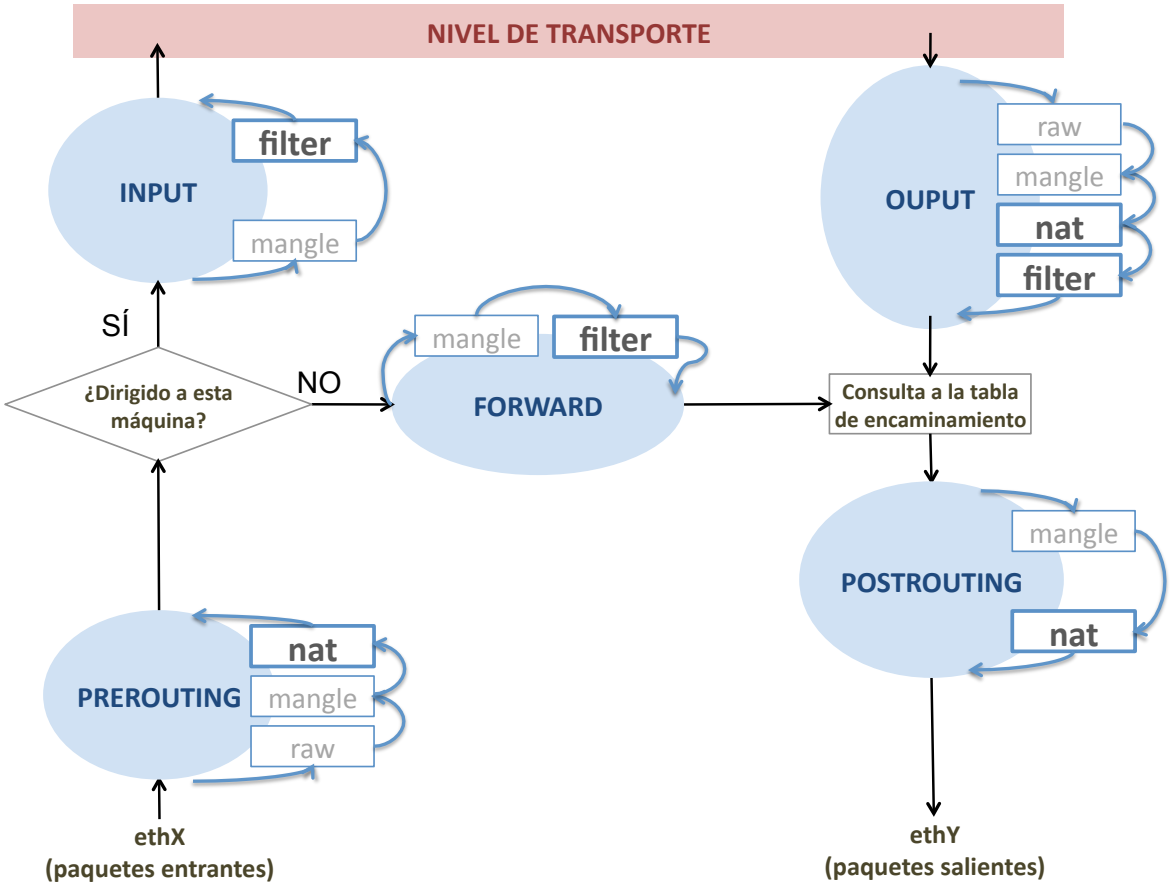

CADENA
Paquete IP
Regla 1
¿Condición?
sí
Acción:
ACCEPT/DROP
no
Regla 2
¿Condición?
sí
Acción:
ACCEPT/DROP
no
Regla 3
¿Condición?
sí
Acción:
ACCEPT/DROP
Regla n
¿Condición?
sí
Acción:
ACCEPT/DROP
no
POLÍTICA

Cadenas (III)

- Las reglas tienen una posición determinada (**número de regla**) dentro de la cadena. A la hora de añadir una nueva regla en una cadena, hay tres posibilidades:
  - **añadir la regla al final 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 (IV): Tipos de cadenas

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



Cadenas (V): 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)

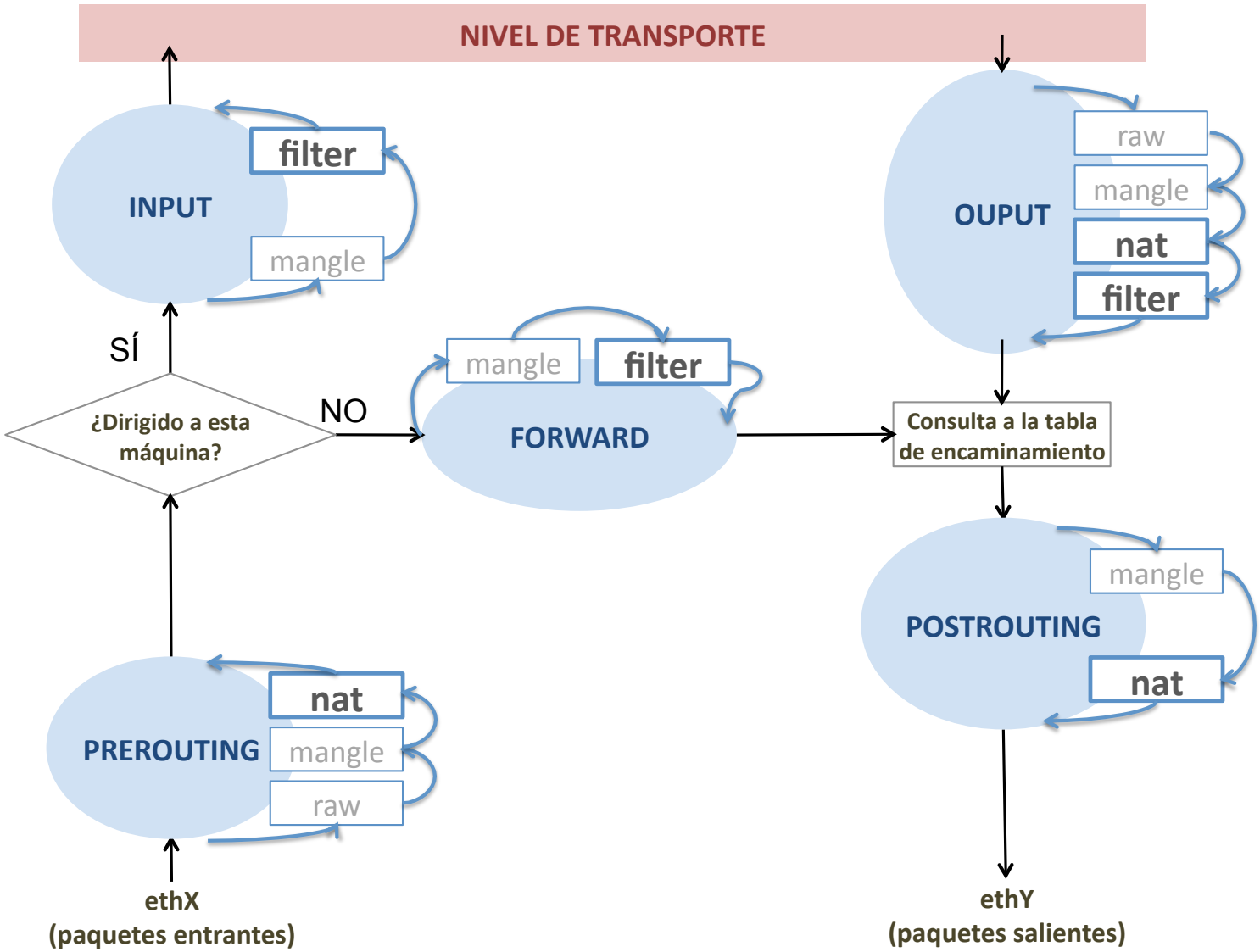
- Una tabla de `iptables` contiene un conjunto de cadenas, tanto predefinidas como de usuario.
- Una tabla concreta engloba las reglas (agrupadas en cadenas) relacionadas con un tipo de procesamiento de los paquetes.
- `Netfilter` define las siguientes tablas:
  - `filter`: engloba las reglas de filtrado 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: ToS
  - `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 sólo 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



Contenidos

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] [-n]` 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:

Interfaz	`-i <interfaz>`: interfaz de entrada `-o <interfaz>`: interfaz de salida
Dirección IP	`-s <dirIP[/máscara]>`: dirección (o direcciones) origen `-d <dirIP[/máscara]>`: dirección (o direcciones) destino
Protocolo	`-p <protocolo>` Se pueden especificar adicionalmente números de puerto: `-p <protocolo> --sport <puerto|puertoInicio:puertoFin>`: puerto origen `-p <protocolo> --dport <puerto|puertoInicio:puertoFin>`: puerto destino
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	`-p tcp --syn`: segmento SYN
	`-p tcp --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 (I)

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.

iptables: acciones (II)

Todas las tablas	`-j LOG [--log-prefix <texto>]` se guarda información de ese paquete en el fichero de `/var/log/kern.log` anteponiendo la cadena de caracteres `<texto>` 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 siguiente regla de la cadena para ese paquete**.

Contenidos

Seguimiento de "conexiones" (I)

- 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`.
- El **estado** en el que puede estar una determinada "conexión" es:
  - **ESTABLISHED**: el paquete está asociado a una "conexión" donde se han transmitido paquetes en ambos sentidos. Por ejemplo: conexión TCP.
  - **NEW**: el paquete está asociado a una nueva conexión o a una conexión en la que no se han transmitido paquetes en ambos sentidos
  - **RELATED**: el paquete está relacionado con una conexión existente pero no pertenece a ella (ej: datos FTP, ICMP error).
  - **INVALID**: el paquete no puede ser identificado o no está asociado a ningún estado.
- Los cálculos de seguimiento se realizan en la cadena PREROUTING (para los paquetes que recibe y reenvía el router) o en la cadena OUTPUT (para los paquetes que crea el router).

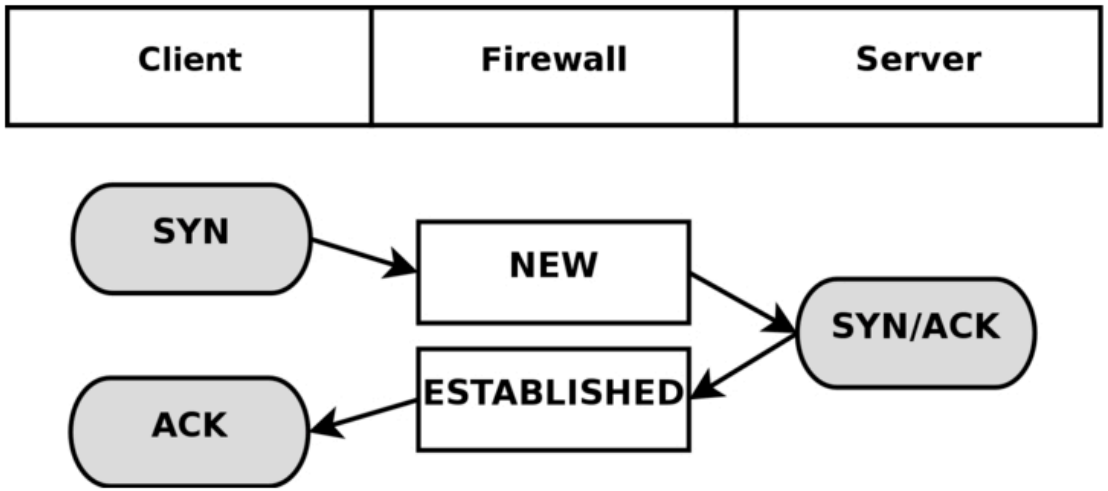
Seguimiento de "conexiones" (II)

- 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
```

- Para cada conexión se mostrará la siguiente información:
  - El primer y segundo campo muestran el protocolo utilizado (el nombre y el código).
  - El tercer campo muestra el tiempo que le queda a dicha conexión para que el sistema de seguimiento borre su entrada. Con cada paquete recibido se actualiza este campo con el valor configurado por defecto, que va disminuyendo hasta que llega a cero.
  - El cuarto y noveno campos muestran información de la "conexión": SYN_SENT, SYN_RECV, ESTABLISHED, UNREPLIED, etc. Esta información incluye el estado de la conexión según la máquina de estados de TCP, que da información más detallada que simplemente **NEW, ESTABLISHED, RELATED, INVALID** (por ejemplo, SYN_SENT implica que la conexión para `conntrack` está en estado `NEW`).
  - El resto de los campos muestran información de los paquetes en ambos sentidos.

Conexiones TCP (I)



- El firewall recibe SYN, estado **NEW**:

```
tcp   6 117 SYN_SENT src=192.168.1.5  dst=192.168.1.35 sport=1031 dport=23 \
        [UNREPLIED] src=192.168.1.35 dst=192.168.1.5  sport=23   dport=1031 use=1
```

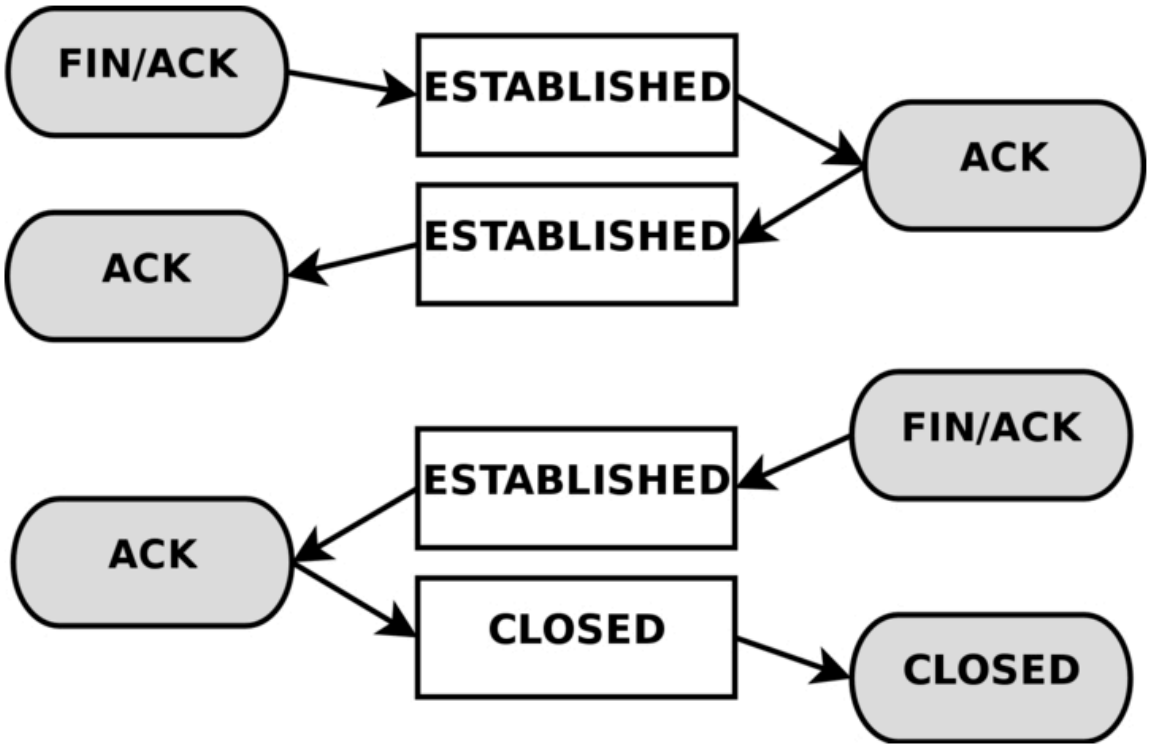
- El firewall recibe SYN/ACK, estado **ESTABLISHED**:

```
tcp   6 57 SYN_RECV src=192.168.1.5  dst=192.168.1.35 sport=1031 dport=23 \
                    src=192.168.1.35 dst=192.168.1.5  sport=23   dport=1031 use=1
```

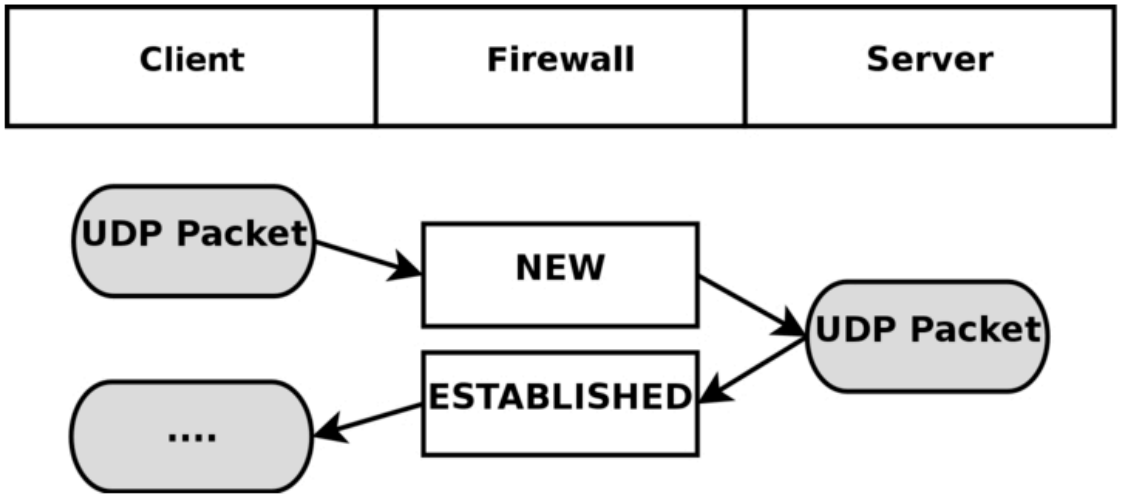
- El firewall recibe ACK y datos TCP, estado **ESTABLISHED**:

```
tcp   6 431999 ESTABLISHED src=192.168.1.5  dst=192.168.1.35 sport=1031 dport=23 \
                           src=192.168.1.35 dst=192.168.1.5  sport=23   dport=1031 \
                           [ASSURED] use=1
```

Conexiones TCP (II)



"Conexiones" UDP



- El firewall recibe un primer paquete UDP, estado **NEW**:

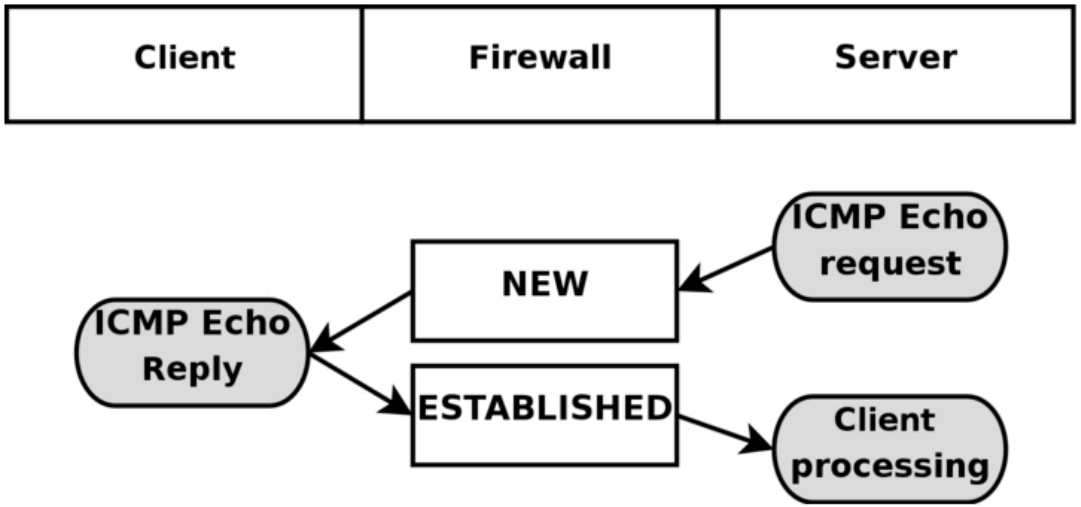
```
udp    17 20     src=192.168.1.2 dst=192.168.1.5 sport=137   dport=1025 \
   [UNREPLIED] src=192.168.1.5 dst=192.168.1.2 sport=1025  dport=137 use=1
```

- El firewall recibe un paquete UDP de respuesta, estado **ESTABLISHED**:

```
udp    17 170 src=192.168.1.2 dst=192.168.1.5 sport=137  dport=1025 \
              src=192.168.1.5 dst=192.168.1.2 sport=1025 dport=137 [ASSURED] use=1
```

"Conexiones" ICMP: Mensajes ICMP de diagnóstico

- Para paquetes ICMP que tienen respuesta: ICMP diagnóstico.



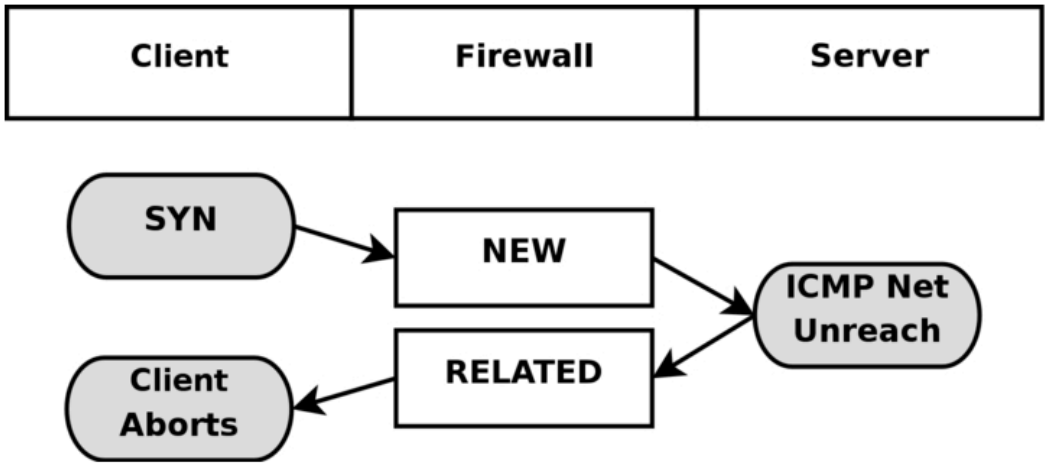
- El firewall recibe primer paquete ICMP Echo Request, estado **NEW**:

```
icmp  1 25   src=192.168.1.6  dst=192.168.1.10 type=8 code=0 id=33029  \
 [UNREPLIED] src=192.168.1.10 dst=192.168.1.6  type=0 code=0 id=33029 use=1
```

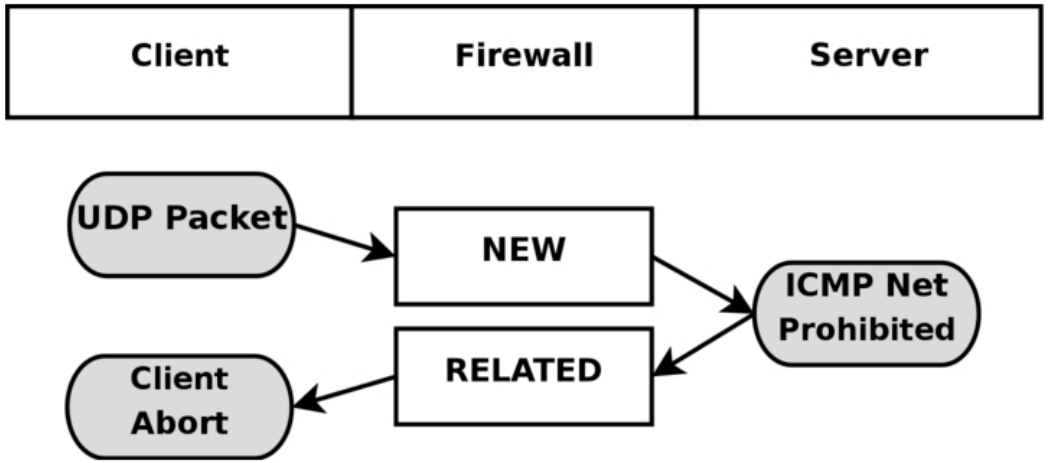
- Cuando el firewall recibe ICMP Echo Reply, el estado pasa a **ESTABLISHED** y se elimina del sistema de seguimiento. Es una "conexión" terminada.

"Conexiones" ICMP: Mensajes ICMP error

- Mensajes ICMP error: no son respuesta a otros mensajes ICMP.
  - En conexión TCP:



  - En "conexión" UDP:



Contenidos

Contenidos

Ejemplos de traducción de direcciones IP y puertos con `iptables` (I)

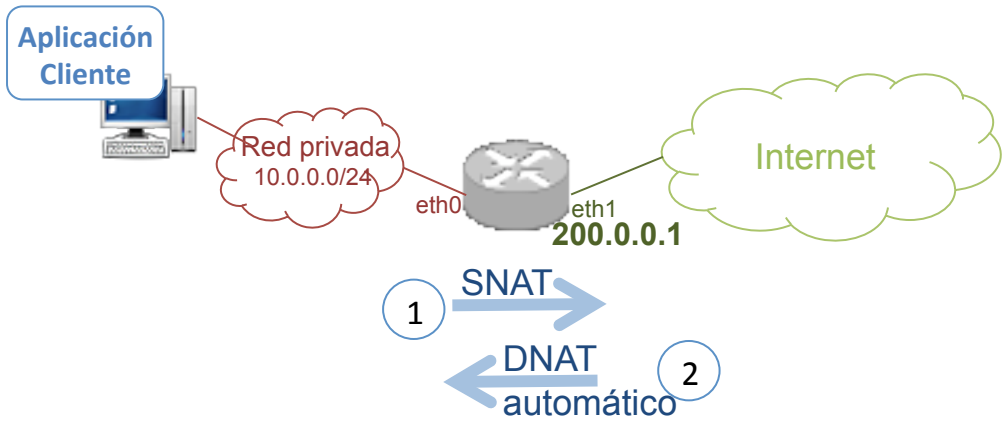
- **Inicialización**
  - Borrar las reglas y reiniciar los contadores:

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

Ejemplos de traducción de direcciones IP y puertos con iptables (II)

- **Source NAT**
  - 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 (eth1) 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 eth1 \
         -j SNAT --to-source 200.0.0.1
```

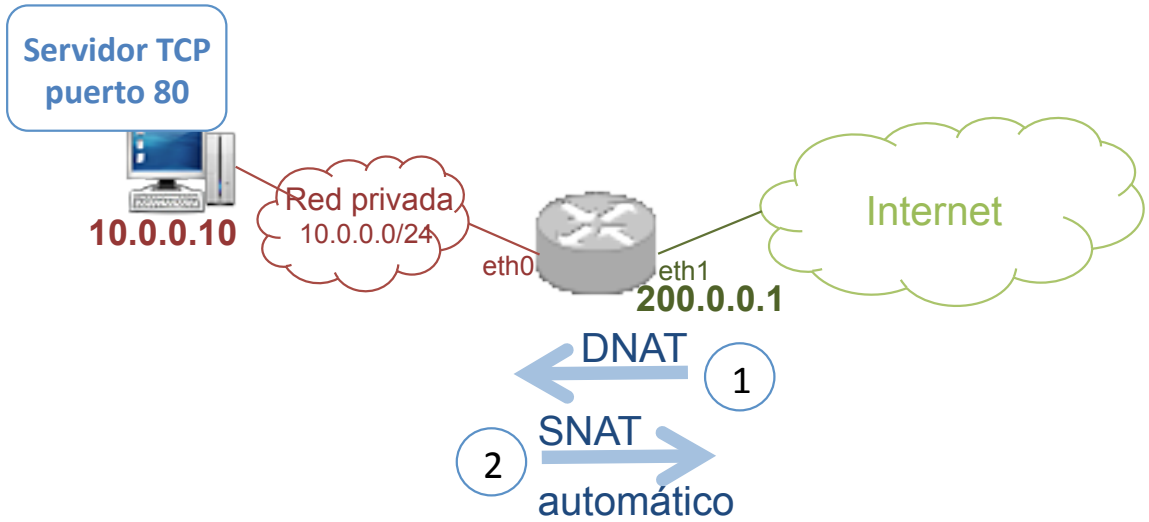


Si la dirección pública del router no es fija, se puede usar MASQUERADE que consulta la dirección de la interfaz de salida del router antes de cambiar la dirección IP origen del paquete.

Ejemplos de traducción de direcciones IP y puertos con `iptables` (III)

- **Destination NAT**
  - 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 -i eth1 -d 200.0.0.1 \
   -p tcp --dport 8080 -j DNAT --to-destination 10.0.0.10:80
```



Ejemplos de traducción de direcciones IP y puertos con iptables (IV)

- Al consultar las reglas de todas las cadenas de la tabla nat:

```
r1:~# iptables -t nat -L -v --line-numbers -n
  Chain PREROUTING (policy ACCEPT 0 packets, 0 bytes)
num  pkts bytes target  prot opt  in  out   source      destination
1    0    0    DNAT    tcp  --  eth0 any   anywhere    200.0.0.1   tcp dpt:http-alt to:10.0.0.10:80

  Chain POSTROUTING (policy ACCEPT 0 packets, 0 bytes)
num  pkts bytes target  prot opt  in  out   source      destination
1    0    0    SNAT    all  --  any eth1   10.0.0.0/24 anywhere    to:200.0.0.1

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target  prot opt  in  out   source  destination
```

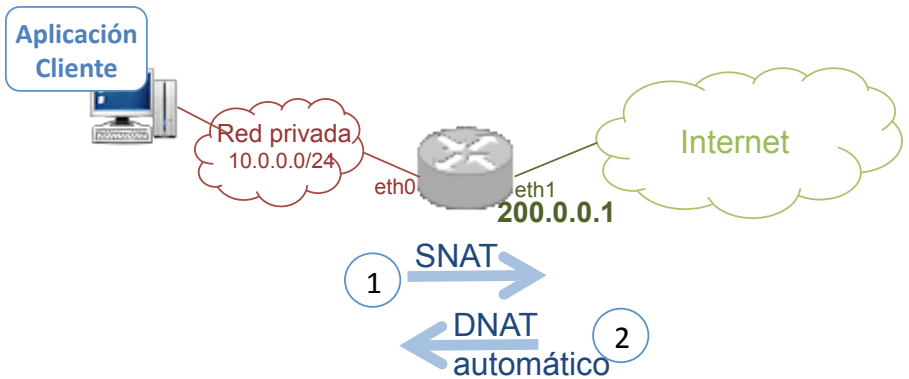
La opción `--line-numbers` imprime al principio de cada regla, la posición en la que se encuentra la regla dentro de la cadena (columna `num`).

La opción `-n` se utiliza para que `iptables` no realice una resolución de DNS inversa de las direcciones IP que haya configuradas en sus reglas.

Prueba1: Conexión con un servidor en Internet desde la máquina interna (I)

- Nada más cargar la configuración del firewall, si desde el pc 10.0.0.10 se inicia una conexión TCP con la máquina en Internet (100.0.0.100) y puerto 13 en la que hay varios paquetes intercambiados, al mostrar la tabla nat, se observa que sólo se ha utilizado la regla SNAT (una sola vez):

```
r1:~# iptables -t nat -L -v --line-numbers -n
  Chain PREROUTING (policy ACCEPT 0 packets, 0 bytes)
num  pkts bytes target  prot opt  in  out   source      destination
1    0    0     DNAT    tcp  --  eth0 any   anywhere    200.0.0.1   tcp dpt:http-alt to:10.0.0.10:80

  Chain POSTROUTING (policy ACCEPT 0 packets, 0 bytes)
num  pkts bytes target  prot opt  in  out   source      destination
1    1    60    SNAT    all  --  any eth1  10.0.0.0/24 anywhere    to:200.0.0.1

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
num  pkts bytes target  prot opt  in  out   source  destination
```

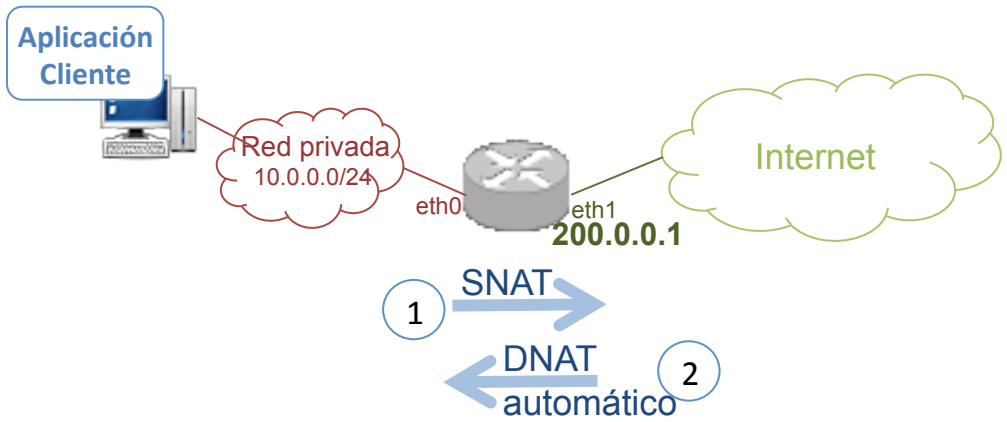


No se ejecuta la regla DNAT, el cambio de la dirección IP destino para los paquetes de entrada es automático.

Prueba1: Conexión con un servidor en Internet desde la máquina interna (II)

- Si consultamos la información del seguimiento de conexiones puede verse como se han contabilizado:
  - 4 paquetes de salida: desde la máquina interna 10.0.0.10 con destino a la máquina en Internet 100.0.0.100. El firewall debe modificar la dirección IP origen (SNAT).
  - 4 paquetes de entrada: desde la máquina en Internet 100.0.0.100 con destino al firewall 200.0.0.1. El firewall debe modificar la dirección IP destino (DNAT automático).

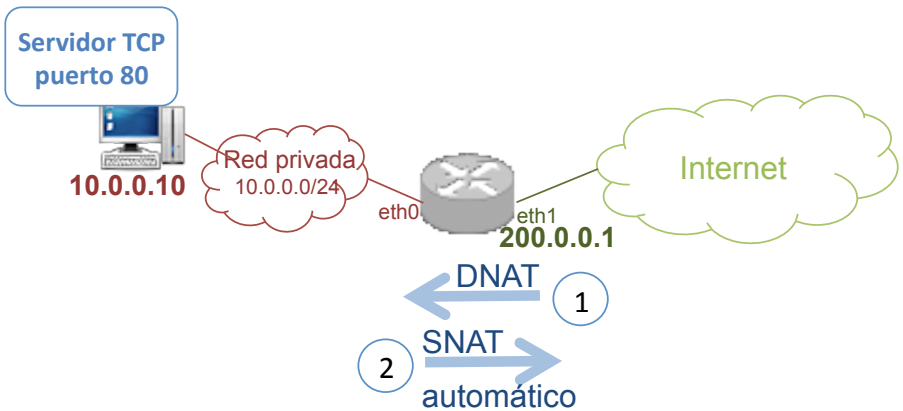
```
r1:~# cat /proc/net/ip_conntrack
tcp      6 98 TIME_WAIT src=10.0.0.10 dst=100.0.0.100 sport=38323 dport=13 packets=4 bytes=216 \
            src=100.0.0.100 dst=200.0.0.1 sport=13 dport=38323 packets=4 bytes=242 [ASSURED]
```



Prueba2: Conexión con el servidor TCP en 10.0.0.10 puerto 80 desde Internet (I)

- Nada más cargar la configuración del firewall, si desde Internet se inicia una conexión TCP con la máquina 200.0.0.1 y puerto 8080 en la que hay varios paquetes intercambiados, al mostrar la tabla nat, se observa que sólo se ha utilizado la regla DNAT (una sola vez):

```
r1:~# iptables -t nat -L -v --line-numbers -n
  Chain PREROUTING (policy ACCEPT 0 packets, 0 bytes)
num  pkts bytes target  prot opt  in  out   source      destination
1    1    60    DNAT    tcp  --  eth0 any   anywhere    200.0.0.1   tcp dpt:http-alt to:10.0.0.10:80

  Chain POSTROUTING (policy ACCEPT 0 packets, 0 bytes)
num  pkts bytes target  prot opt  in  out   source      destination
1    0    0     SNAT    all  --  any eth1  10.0.0.0/24 anywhere    to:200.0.0.1

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
num  pkts bytes target  prot opt  in  out   source  destination
```

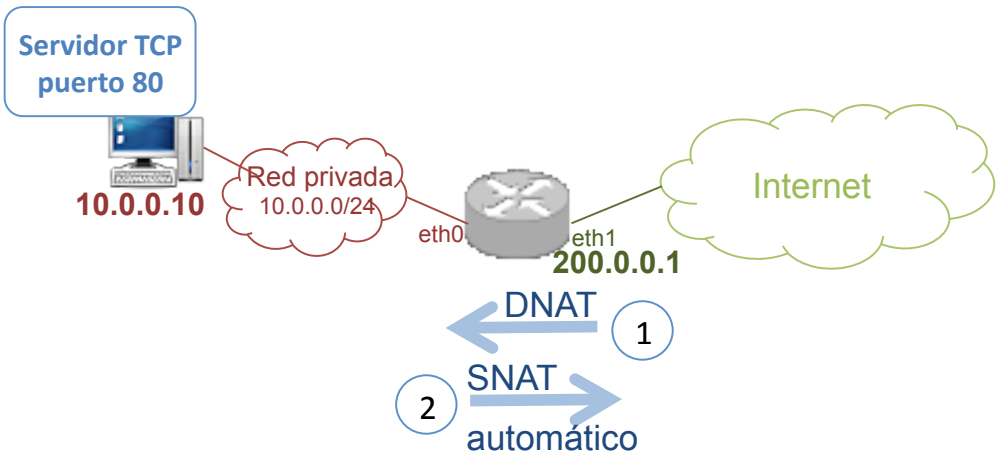


No se ejecuta la regla SNAT, el cambio de la dirección IP origen para los paquetes de salida es automático.

Prueba2: Conexión con el servidor TCP en 10.0.0.10 puerto 80 desde Internet (II)

- Si consultamos la información del seguimiento de conexiones puede verse como se han contabilizado:
  - 4 paquetes de entrada: desde la máquina en Internet 100.0.0.100 con destino el firewall 200.0.0.1. El firewall debe modificar la dirección IP destino y puerto (DNAT).
  - 4 paquetes de salida: desde la máquina interna 10.0.0.10 con destino en Internet 100.0.0.100. El firewall debe modificar la dirección IP origen (SNAT automático).

```
r1:~# cat /proc/net/ip_conntrack
tcp      6 98 TIME_WAIT src=100.0.0.100 dst=200.0.0.1 sport=32775 dport=8080 packets=4 bytes=216 \
            src=10.0.0.10 dst=100.0.0.100 sport=80 dport=32775 packets=4 bytes=242 [ASSURED]
```



Contenidos

Ejemplos de configuración de filtrado con iptables (I)

1. **Inicialización**
   - 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
```

Ejemplos de configuración de filtrado con iptables (II)

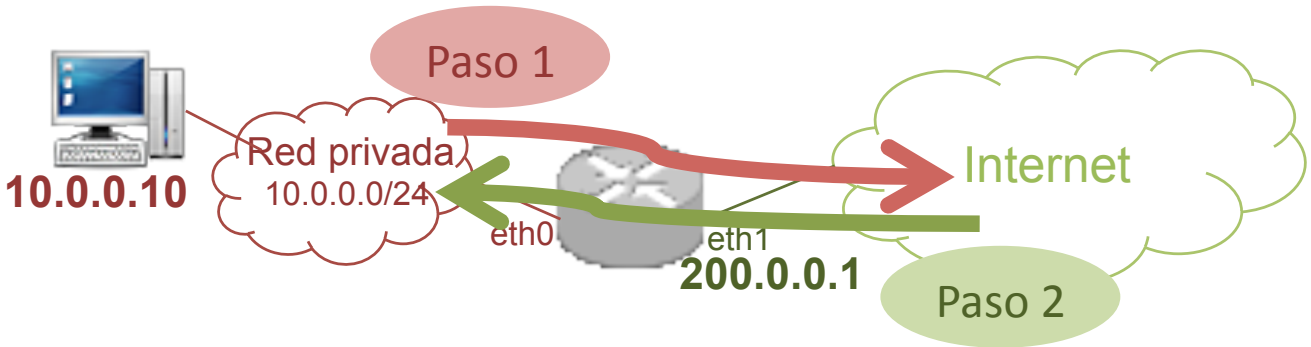
2. **Filtrado: permitir cualquier tráfico saliente de la red privada y el tráfico entrante de respuesta**

   - 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
```



Ejemplos de configuración de filtrado con iptables (III)

- **Guardar en un fichero de log el contenido de los paquetes que se reenvían**
  Para entender el funcionamiento de cómo se aplican las reglas se puede almacenar en el fichero de log la información de los paquetes a los que se les aplican las reglas que permiten el reenvío del tráfico saliente y las posibles respuestas a dicho tráfico.

Regla 1
Regla 2
Regla 3
Regla 4

```
iptables -t filter -A FORWARD -i eth0 -o eth1 -j LOG --log-prefix "Regla 2 accept "
iptables -t filter -A FORWARD -i eth0 -o eth1 -j ACCEPT

iptables -t filter -A FORWARD -i eth1 -o eth0 -m state \
         --state RELATED,ESTABLISHED -j LOG --log-prefix "Regla 4 accept "
iptables -t filter -A FORWARD -i eth1 -o eth0 -m state \
         --state RELATED,ESTABLISHED -j ACCEPT
```

- **Mostrar la información de una configuración**
  Se ha enviado un paquete ICMP *echo request* desde la red privada a Internet y se ha recibido respuesta. La información muestra la cantidad de paquetes a los que se les ha aplicado cada regla.

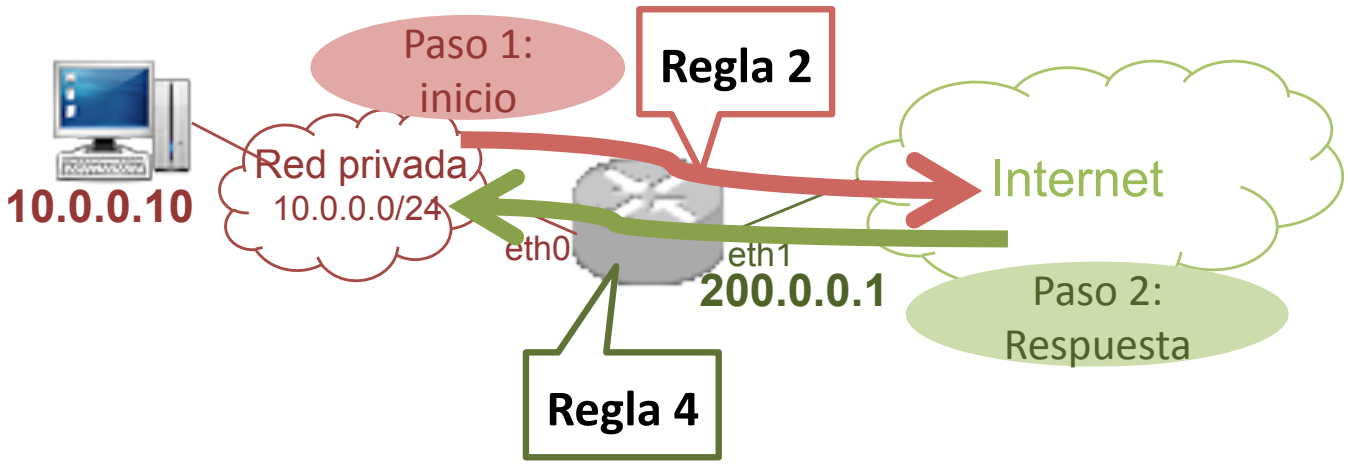
Regla 1
Regla 2
Regla 3
Regla 4

```
r1:~# iptables -t filter -L -v
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target  prot opt  in  out   source  destination

Chain FORWARD (policy ACCEPT 4 packets, 336 bytes)
 pkts bytes target  prot opt  in  out   source  destination
 1    84    LOG     all  --  eth0 eth1 anywhere  anywhere    LOG level warning prefix 'Regla 2 accept '
 1    84    ACCEPT  all  --  eth0 eth1 anywhere  anywhere
 1    84    LOG     all  --  eth1 eth0 anywhere  anywhere    state RELATED,ESTABLISHED
                                                             LOG level warning prefix 'Regla 4 accept '
 1    84    ACCEPT  all  --  eth1 eth0 anywhere  anywhere    state RELATED,ESTABLISHED

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target  prot opt  in  out   source  destination
```

Ejemplos de configuración de filtrado con iptables (IV)

- **Mostrar el fichero de log**
  Los mensajes generados por la configuración previa de LOG en iptables pueden consultarse al final del fichero `/var/log/messages`. En este caso el LOG debe contener el paquete ICMP *echo request* y el ICMP *echo reply*:

```
r1:~# less /var/log/messages
Nov  4 19:07:00 r1 kernel: Regla 2 accept IN=eth0 OUT=eth1 SRC=10.0.0.10 DST=12.0.0.10 \
   LEN=84 TOS=0x00 PREC=0x00 TTL=63 ID=0 DF PROTO=ICMP TYPE=8 CODE=0 ID=32522 SEQ=1

Nov  4 19:07:00 r1 kernel: Regla 4 accept IN=eth1 OUT=eth0 SRC=12.0.0.10 DST=10.0.0.10 \
   LEN=84 TOS=0x00 PREC=0x00 TTL=63 ID=20573 PROTO=ICMP TYPE=0 CODE=0 ID=32522 SEQ=1
```



Ejemplos de configuración de filtrado con iptables (V)

- **Consultar el sistema de seguimiento ip_conntrack**
  El sistema de seguimiento no muestra ninguna información porque hay una anotación del paquete ICMP *echo request* pero en cuanto se recibe el paquete ICMP *echo reply* se borra la información de dicha "conexión". Por tanto, el siguiente comando no muestra ninguna información.

```
r1:~# watch -n 1 cat /proc/net/ip_conntrack
```

Nótese que si la máquina a la que se dirige el paquete ICMP *echo request* no responde, el sistema de seguimiento tendría anotado durante un tiempo el paquete ICMP echo request:

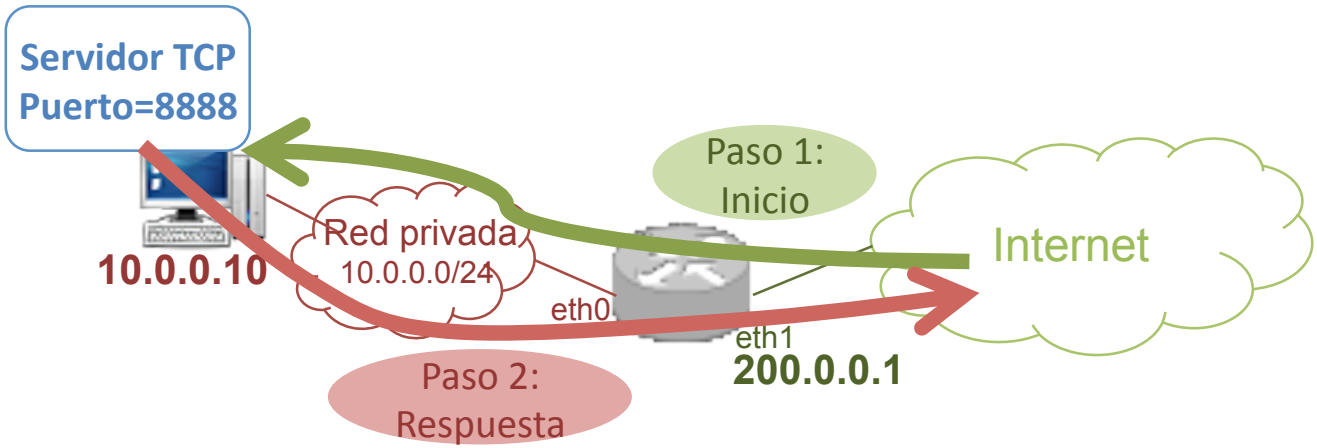
```
r1:~# watch -n 1 cat /proc/net/ip_conntrack

icmp 1 28 src=11.0.0.10 dst=12.0.0.56 type=8 code=0 id=11023 packets=1 bytes=84 [UNREPLIED]
          src=12.0.0.56 dst=11.0.0.10 type=0 code=0 id=11023 packets=0 bytes=0 mark=0 use=2
```

Ejemplos de configuración de filtrado con iptables (VI)

3 Filtrado: permitir tráfico tcp entrante en la red privada

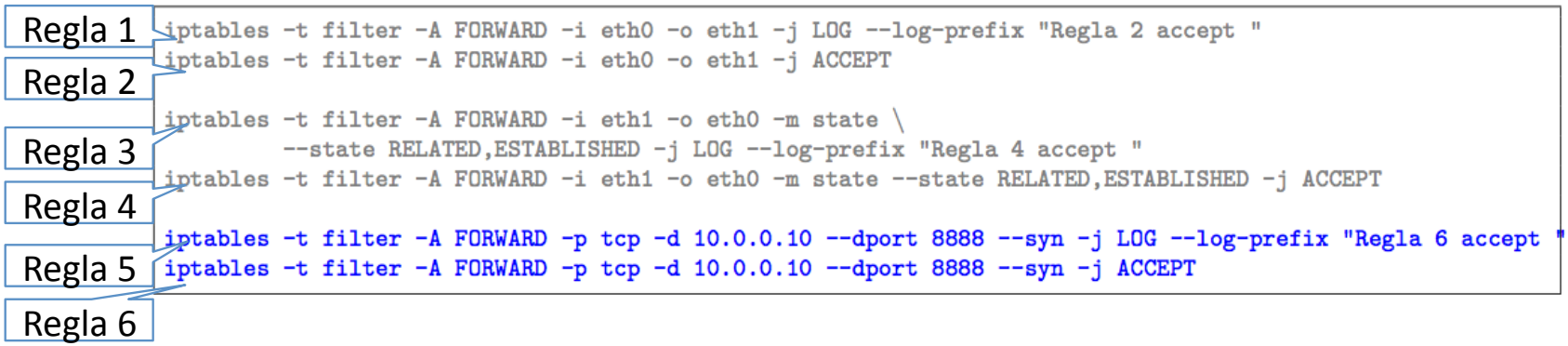
- Permitir el paso de segmentos TCP de establecimiento de conexión de entrada dirigidos a una dirección IP de la red interna (10.0.0.10) y a un puerto (8888).

```
iptables -t filter -A FORWARD -d 10.0.0.10 \
      -p tcp --dport 8888 --syn -j ACCEPT
```



Ejemplos de configuración de filtrado con iptables (VII)

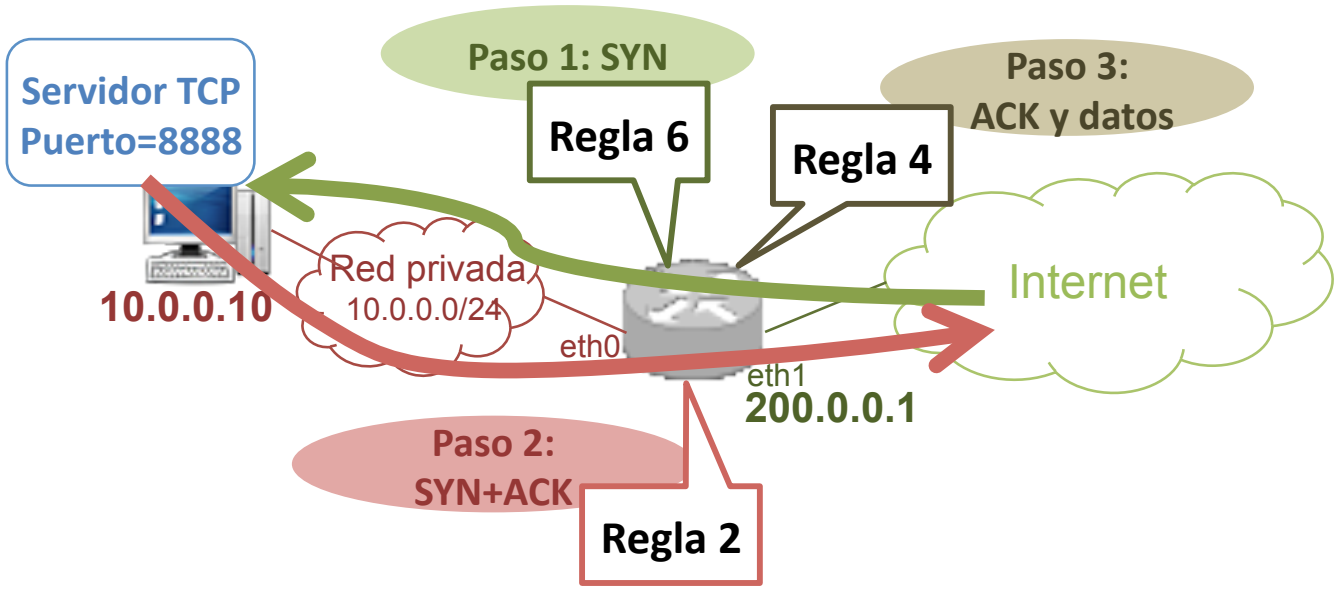
- Si añadimos esta regla a las anteriores y configuramos iptables para almacenar los paquetes que cumplan la regla en el fichero de log:



Ejemplos de configuración de filtrado con iptables (VIII)

- Se abre una conexión TCP desde una máquina externa (SYN, SYN+ACK, ACK). Al mostrar el fichero de log:

```
r1:~# less /var/log/messages
Nov  5 00:31:04 r1 kernel: Regla 6 accept IN=eth1 OUT=eth0 SRC=12.0.0.4 DST=11.0.0.10 LEN=60
TOS=0x00 PREC=0x00 TTL=63 ID=11521 DF PROTO=TCP SPT=58228 DPT=8888 WINDOW=5840 RES=0x00
SYN URGP=0
Nov  5 00:31:04 r1 kernel: Regla 2 accept IN=eth0 OUT=eth1 SRC=11.0.0.10 DST=12.0.0.4 LEN=60
TOS=0x00 PREC=0x00 TTL=63 ID=0 DF PROTO=TCP SPT=8888 DPT=58228 WINDOW=5792 RES=0x00 ACK
SYN URGP=0
Nov  5 00:31:04 r1 kernel: Regla 4 accept IN=eth1 OUT=eth0 SRC=12.0.0.4 DST=11.0.0.10 LEN=52
TOS=0x00 PREC=0x00 TTL=63 ID=11522 DF PROTO=TCP SPT=58228 DPT=8888 WINDOW=2920 RES=0x00
ACK URGP=0
```



Ejemplos de configuración de filtrado con iptables (IX)

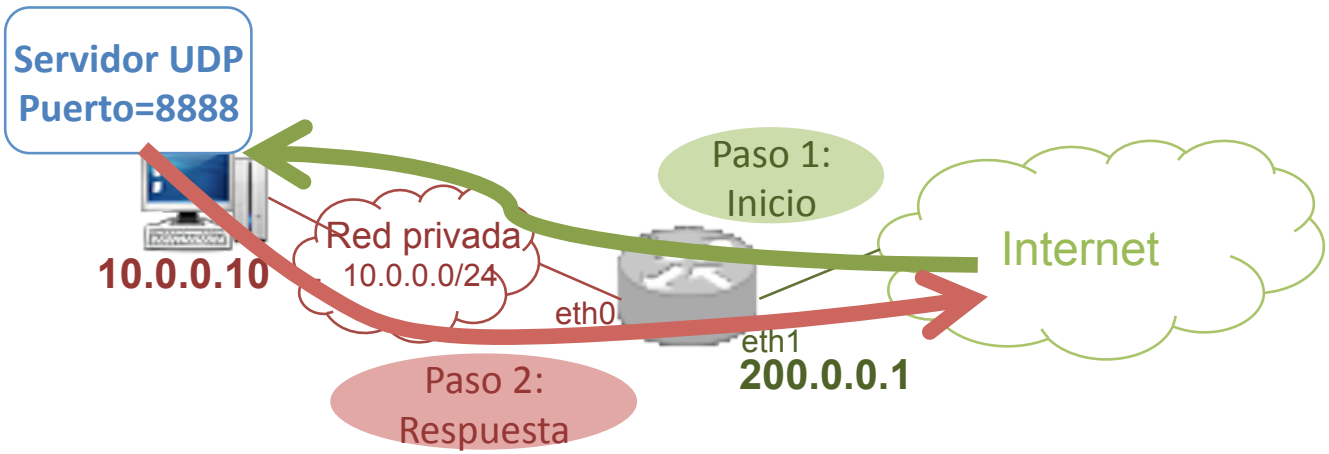
- Al mostrar el sistema de seguimiento ip_conntrack después del establecimiento de la conexión:

```
r1:~# watch -n 1 cat /proc/net/ip_conntrack

tcp 6 430970 ESTABLISHED src=12.0.0.10 dst=10.0.0.10 sport=58228 dport=8888 packets=2 bytes=112
   src=10.0.0.10 dst=12.0.0.10 sport=8888 dport=58228 packets=1 bytes=60 [ASSURED] mark=0 use=1
```

Ejemplos de configuración de filtrado con iptables (X)

4. **Filtrado: permitir tráfico UDP entrante en la red privada**
   - 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 (8888).

```
iptables -t filter -A FORWARD -d 10.0.0.10 \
    -p udp --dport 8888 -j ACCEPT
```



Ejemplos de configuración de filtrado con iptables (XI)

- Si añadimos esta regla a las anteriores y configuramos iptables para almacenar los paquetes que cumplan la regla en el fichero de log:

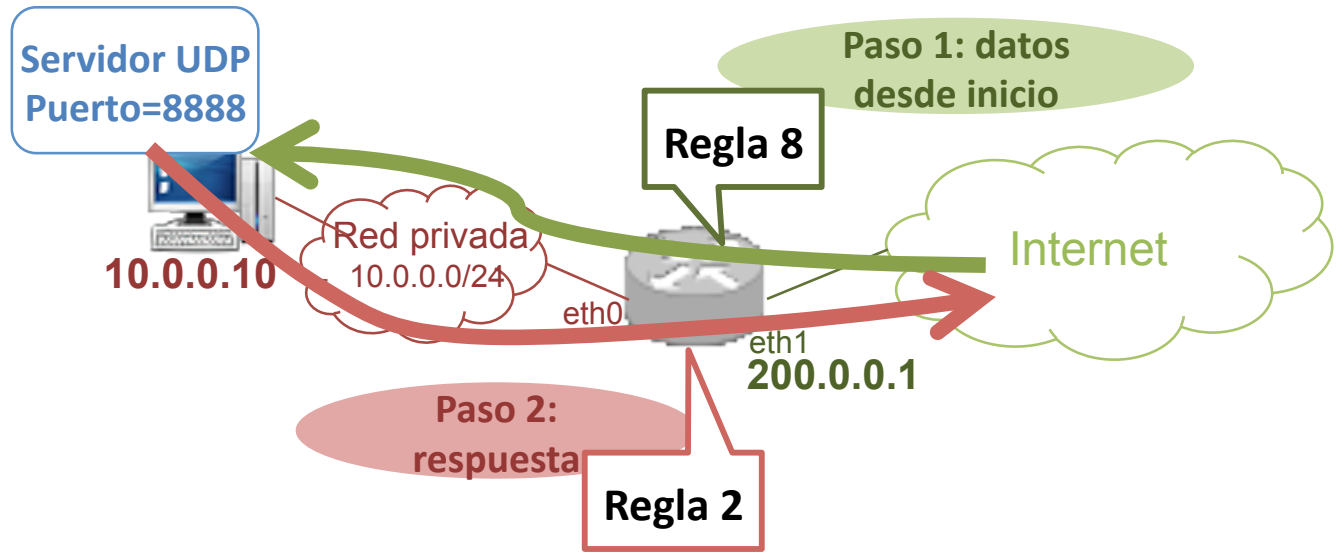
Regla 1
Regla 2
Regla 3
Regla 4
Regla 5
Regla 6
Regla 7
Regla 8

```
iptables -t filter -A FORWARD -i eth0 -o eth1 -j LOG --log-prefix "Regla 2 accept "
iptables -t filter -A FORWARD -i eth0 -o eth1 -j ACCEPT

iptables -t filter -A FORWARD -i eth1 -o eth0 -m state \
         --state RELATED,ESTABLISHED -j LOG --log-prefix "Regla 4 accept "
iptables -t filter -A FORWARD -i eth1 -o eth0 -m state --state RELATED,ESTABLISHED -j ACCEPT

iptables -t filter -A FORWARD -p tcp -d 10.0.0.10 --dport 8888 --syn -j LOG --log-prefix "Regla 6 accept "
iptables -t filter -A FORWARD -p tcp -d 10.0.0.10 --dport 8888 --syn -j ACCEPT

iptables -t filter -A FORWARD -p udp -d 10.0.0.10 --dport 8888 -j LOG --log-prefix "Regla 8 accept "
iptables -t filter -A FORWARD -p udp -d 10.0.0.10 --dport 8888 -j ACCEPT
```

Ejemplos de configuración de filtrado con iptables (XII)

- Se envía un paquete de datos UDP desde una máquina externa. Al mostrar el fichero de log:

```
r1:~# less /var/log/messages
Nov  5 00:32:04 r1 kernel: Regla accept 8 IN=eth1 OUT=eth0 SRC=12.0.0.4 DST=11.0.0.10 LEN=34
TOS=0x00 PREC=0x00 TTL=63 ID=2916 DF PROTO=UDP SPT=33666 DPT=8888 LEN=14
```



- Al mostrar el sistema de seguimiento ip_conntrack:

```
r1:~# watch -n 1 cat /proc/net/ip_conntrack

udp 17 21  src=12.0.0.10 dst=10.0.0.10 sport=58300 dport=8888 packets=1 bytes=34 [UNREPLIED]
   src=10.0.0.10 dst=12.0.0.10 sport=8888 dport=58300 packets=0 bytes=0 mark=0 use=1
```

Referencias

- Iptables Tutorial:
  `http://www.iptables.info`
- Linux Advanced Routing & Traffic Control:
  `http://www.lartc.org`