Enrutamiento y filtrado

1. Configuración de un servidor Linux como router

La función de enrutamiento se integra de forma nativa en el núcleo de Linux. Por lo tanto, hay pocas preguntas que hacerse, cualquier máquina Linux es un router en potencia. Sin embargo, esta función no está activada por defecto tras el arranque. Por tanto, hay que configurarla antes de realizar cualquier operación de enrutamiento.

a. Activación del enrutamiento en un servidor Linux

Sabemos que cualquier sistema Linux presenta un sistema de archivos virtual **/proc** que permite observar en directo un cierto número de componentes y parámetros. La activación del enrutamiento se realiza modificando el contenido del archivo **/proc/sys/net/ipv4/ip_forward**. Este archivo contiene un solo carácter que por defecto es **0** para indicar que el enrutamiento está inactivo.

Modificación del archivo ip_forward para activar el enrutamiento

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

Una vez se ha realizado el cambio, la máquina Linux está lista para enrutar paquetes que lleguen a sus interfaces. Este parámetro es volátil y se perderá una vez que la máquina se detenga. Sin embargo, se puede anular el enrutamiento realizando la operación inversa.

Modificación del archivo ip_forward para desactivar el enrutamiento

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

Otra posibilidad es usar el comando **sysctl**, que permite modificar dinámicamente los parámetros funcionales del núcleo. **sysctl** permite modificar directamente todos los archivos que se encuentran en la estructura de directorios que cuelga de **/proc/sys**.

Activación del enrutamiento con sysctl

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

Estos comandos son efectivos durante toda la sesión y deben volver a introducirse después de cada reinicio. Por supuesto, se pueden poner en un script de servicio llamado en el arranque o modificar el archivo **/etc/sysctl.conf**.

Activación permanente del enrutamiento en el archivo /etc/sysctl.conf

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

b. Consulta de la tabla de enrutamiento

En este punto, el router Linux es perfectamente capaz de enrutar paquetes. Sin embargo, solo podrá hacerlo a redes conocidas, es decir, referenciadas en su tabla de enrutamiento.

La tabla de enrutamiento se mantiene en memoria, pero se puede consultar usando varios comandos.

Visualización de la tabla de enrutamiento con el comando route

```
route -n
```

El parámetro -n es opcional, pero sirve para ahorrar mucho tiempo en la visualización, ya que no fuerza a que el comando intente resolver los nombres de las redes devueltas. Además, si la dirección en cuestión no se informa en una zona DNS inversa, esta petición se realiza en vano y hay que esperar varios segundos para que se muestre la salida del comando.

Visualización de la tabla de enrutamiento con el comando netstat

```
netstat -nr
```

Donde la opción -r hace que el comando muestre la tabla de enrutamiento y -n evita que se realice la resolución de nombres. El comando **netstat** tiene muchos posibles usos, pero a menudo se utiliza en este simple contexto de consulta de la tabla de enrutamiento.

Ejemplo de visualización de la tabla de enrutamiento

A menudo, la visualización de la tabla de enrutamiento es el único método sencillo para consultar el valor de la puerta de enlace predeterminada.

```
beta:~# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.1.0     0.0.0.0         255.255.255.0   U     0      0        0 eth1
192.168.0.0     0.0.0.0         255.255.255.0   U     0      0        0 eth0
0.0.0.0         192.168.0.1     0.0.0.0         UG    0      0        0 eth0
```

c. Gestión de rutas estáticas

Las únicas entradas presentes de forma automática en la tabla de enrutamiento son las redes a las que el router está conectado directamente, así como la puerta de enlace predeterminada. Por tanto, el router puede usar estas entradas de la tabla de enrutamiento sin necesidad de otra configuración. Si el router tiene que enrutar paquetes a otras redes, habrá que añadir de forma manual las rutas en la tabla de enrutamiento.

Agregar rutas estáticas en la tabla de enrutamiento

```
route add -net red_objetivo netmask máscara gw router
```

Agregar rutas estáticas: opciones y parámetros	
-net	La ruta incluida es una red. (El objetivo podría ser un solo host, aunque es menos frecuente.)
red_objetivo	La dirección de red que la nueva ruta permitirá alcanzar.
máscara	La máscara de subred asociada a la nueva ruta.
gw *router*	Indica el router que se utilizará para alcanzar la red objetivo.

Añadir una puerta de enlace por defecto

```
route add default gw router
```

```
route add -net 0.0.0.0 gw router
```

En la segunda sintaxis, 0.0.0.0 representa la ruta por defecto. Esta representación de la ruta por defecto es universal y se puede aplicar en casi la totalidad de sistemas que usen un tabla de enrutamiento IP.

Por supuesto, también se pueden eliminar las rutas estáticas que ya no sean necesarias o que estén guardadas por error.

Eliminación de rutas estáticas de la tabla de enrutamiento

```
route del -net red_objetivo netmask máscara
```

Ejemplo de adición de rutas

```
beta:~# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.1.0     0.0.0.0         255.255.255.0   U     0      0        0 eth1
192.168.0.0     0.0.0.0         255.255.255.0   U     0      0        0 eth0
0.0.0.0         192.168.0.1     0.0.0.0         UG    0      0        0 eth0
beta:~# route add -net 10.0.0.0 netmask 255.0.0.0 gw 192.168.1.99
```

```
beta:~# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.1.0     0.0.0.0         255.255.255.0   U     0      0        0 eth1
192.168.0.0     0.0.0.0         255.255.255.0   U     0      0        0 eth0
10.0.0.0        192.168.1.99    255.0.0.0       UG    0      0        0 eth1
0.0.0.0         192.168.0.1     0.0.0.0         UG    0      0        0 eth0
```

Ejemplo de eliminación de rutas

```
beta:~# route del -net 10.0.0.0 netmask 255.0.0.0
beta:~# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.1.0     0.0.0.0         255.255.255.0   U     0      0        0 eth1
192.168.0.0     0.0.0.0         255.255.255.0   U     0      0        0 eth0
0.0.0.0         192.168.0.1     0.0.0.0         UG    0      0        0 eth0
beta:~#
```

2. iptables

Iptables se utiliza para gestionar el filtrado de paquetes IP en un sistema Linux. Usa un solo comando: **iptables**, y se configura mediante la aplicación de reglas de gestión de paquetes. Iptables puede filtrar el tráfico que transita por un router Linux, pero también el tráfico entrante y saliente de cualquier servidor o estación de trabajo en una sola interfaz.

Aunque iptables constituye una herramienta muy potente de gestión del tráfico, esta ventaja hace que su archivo de configuración sea todo excepto intuitivo. Sin embargo, con una aproximación estructurada se puede aprender su funcionamiento bastante rápido. Los siguientes párrafos exponen los conceptos fundamentales de iptables, para utilizarlos más tarde en configuraciones de cortafuegos.

a. Tablas

Iptables se basa en tablas asociadas a un modo funcional. Según el tipo de regla que se desea añadir en el funcionamiento de iptables, se precisará la tabla asociada. Las tablas principales utilizadas son **filter** para filtrar paquetes y **nat** para la traducción de direcciones entre una red privada y una red pública.

La tabla **filter** es la tabla por defecto. Por ello, cuando se establece una regla de iptables con el objetivo de filtrar paquetes, se sobreentiende y, por consiguiente, no se detalla.

La tabla **nat** sirve para traducir direcciones y debe precisarse sistemáticamente cuando se invoca.

b. Cadenas

Una cadena de iptables representa un tipo de tráfico desde el punto de vista de su circulación por una máquina. Las cadenas permiten especificar si una regla debe aplicarse al tráfico que entra en una máquina, que sale o que la cruza.

La cadena **INPUT** identifica el tráfico entrante, la cadena **OUTPUT** identifica el tráfico saliente y la cadena **FORWARD** identifica el tráfico que atraviesa la máquina, entrando por una interfaz y saliendo por otra. Atención, aunque un paquete que atraviesa el router es, desde un punto de vista físico, respectivamente entrante, encaminado y saliente, iptables lo considera solamente como encaminado (cadena FORWARD). Las cadenas INPUT y OUTPUT se reservan al tráfico con origen o destino explícito el host al que se le aplican las reglas.

También hay otra cadena llamada **POSTROUTING**, que se utiliza en la configuración de NAT y tiene como objetivo tratar paquetes después de una operación de enrutamiento.

Las cadenas siempre se indican en mayúsculas en la sintaxis de iptables.

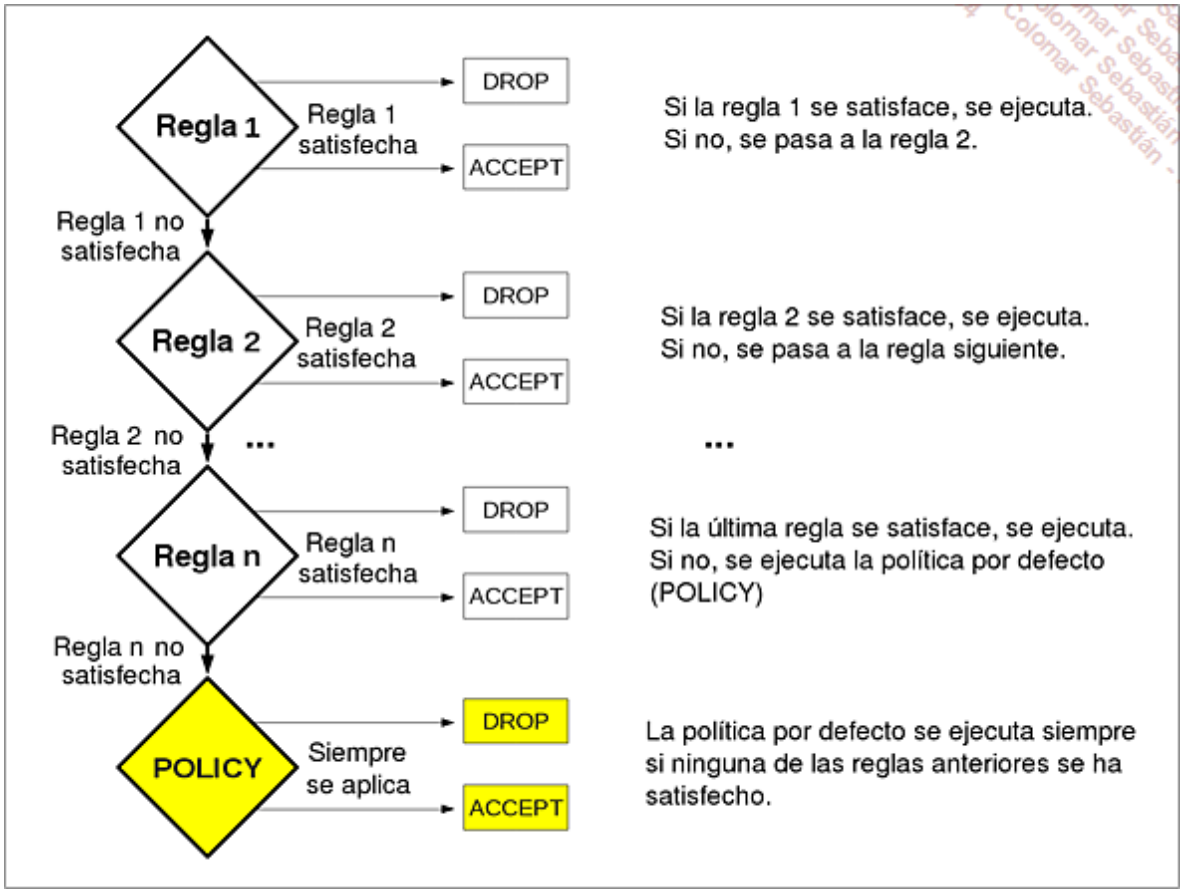
c. Acciones

Cuando se satisface una regla, el sistema genera una acción sobre el paquete comprobado. Las acciones principales son **ACCEPT**, que permite el paso del paquete, y **DROP**, que lo destruye.

En la sintaxis de iptables, la acción (**target** en el manual en línea) se anuncia con el parámetro **-j**.

Las acciones siempre se escriben en mayúsculas según la sintaxis de iptables.

d. Tratamiento de reglas



Se aplican las reglas una por una para cada paquete filtrado. Si se satisface una regla, se genera una acción sobre el paquete y finaliza su tratamiento. Si no se satisface, se comprueba la siguiente regla. En caso de que ninguna regla se haya satisfecho, el paquete recibe un tratamiento por defecto configurado con una regla específica llamada «política» (policy).

Se pueden mostrar las reglas aplicadas en orden para cada una de las cadenas.

Visualización de reglas efectivas

```
iptables -L
```

Ejemplo de visualización de reglas

Este ejemplo muestra las reglas activas en un sistema Linux sin configurar. Se ve la política aplicada para cada una de las cadenas y se comprueba la ausencia de reglas de filtrado.

```
alfa:~# 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 comando **iptables -L** muestra la interpretación de las reglas activas. Si se desea saber quién tiene permisos para establecer estas reglas, se recomienda usar la opción **-S**.

Ejemplo de visualización de reglas según la sintaxis

*La opción -**S** es particularmente útil cuando se enfrenta a un sistema configurado por otra persona y no sabe qué comandos se han introducido para llegar a esa configuración.*

```
alfa:~# iptables -S
-P INPUT ACCEPT
-P FORWARD ACCEPT
-P OUTPUT ACCEPT
alfa:~#
```