Bridges de Filtrado

Resumen

A menudo es útil dividir una red física (por ejemplo una Ethernet) en dos segmentos separados sin tener que crear subredes y usar un router para vincularlas. El dispositivo que conecta las dos redes se llama bridge. Un sistema FreeBSD con dos interfaces de red es suficiente para actuar como bridge.

Un bridge funciona escaneando las direcciones del nivel MAC (direcciones Ethernet) de los dispositivos conectados a cada una de sus interfaces de red y luego reenvía el tráfico entre las dos redes solo si la fuente y el destino están en diferentes segmentos. En muchos aspectos, un bridge es similar a un switch de Ethernet con solo dos puertos.

Tabla de contenidos

1. ¿Por qué utilizar un bridge que haga filtrado? . . . 1
2. Proceso de instalación . . . 1
3. Preparación final . . . 2
4. Habilitando el bridge . . . 3
5. Configurando el firewall . . . 4
6. Colaboradores . . . 6

1. ¿Por qué utilizar un bridge que haga filtrado?

Sucede con bastante frecuencia que, gracias a la reducción del coste de las conexiones de banda ancha a Internet (xDSL) y a la reducción de las direcciones IPv4 disponibles, muchas empresas están conectadas a Internet las 24 horas del día y con pocas (a veces ni siquiera dos) direcciones IP. A menudo en estas situaciones es necesario tener un firewall (también conocido como cortafuegos) que filtre el tráfico entrante y saliente desde y hacia Internet, pero una solución de filtrado de paquetes puede que no sea posible, ya sea por problemas de subredes, porque el router sea de propiedad del proveedor de servicios de internet (ISP), o porque no admite tales funcionalidades. En escenarios como estos se recomienda el uso de un brigde que realice el filtrado.

Una buena solución sería configurar un firewall basado en un bridge. Lo instalaremos entre el router xDSL y su hub/switch Ethernet, evitando así problemas de numeración IP.

2. Proceso de instalación

No es difícil añadir funcionalidades de brigde a un sistema FreeBSD. Desde la versión 4.5 es posible

cargar funcionalidades como módulos en lugar de tener que volver a compilar el kernel, lo cual simplifica mucho el procedimiento. En las siguientes subsecciones explicaré ambas formas de instalación.



No sigas ambas instrucciones: un procedimiento *es excluyente* con el otro. Escoge la mejor opción de acuerdo a tus necesidades y habilidades.

Antes de continuar asegúrese de tener al menos dos tarjetas Ethernet que admitan el modo promiscuo tanto para la recepción como para la transmisión, ya que deben poder enviar paquetes Ethernet con cualquier dirección, no solo la suya. Además, para tener una buena tasa de transferencia, las tarjetas deben ser tarjetas del bus PCI. Las mejores opciones siguen siendo Intel EtherExpress™ Pro, seguida de la 3Com® 3c9xx series. Para simplificar la configuración del firewall, puede ser útil tener dos tarjetas de diferentes fabricantes (con diferentes controladores) para distinguir claramente qué interfaz está conectada al router y cuál a la red interna.

2.1. Configuración del kernel

Si sigues este método es porque has decidido utilizar el método de instalación más antiguo y también el que ha sido más probado. Para empezar, debes agregar las siguientes líneas a tu archivo de configuración del kernel:

```
options BRIDGE
options IPFIREWALL
options IPFIREWALL_VERBOSE
```

La primera línea añade el soporte para el bridge, la segunda añade la compatibilidad con el firewall y la tercera se refiere a las funciones de logging del firewall.

Ahora es necesario construir e instalar el nuevo kernel. Puedes encontrar información detallada en la sección Building and Installing a Custom Kernel del FreeBSD Handbook.

2.2. Carga de módulos

Si has elegido usar el nuevo método de instalación (más simple), lo único que debes hacer es añadir la siguiente línea a /boot/loader.conf

```
bridge_load="YES"
```

Así el módulo bridge.ko se cargará junto con el kernel durante el inicio del sistema. No es necesario añadir una línea similar para el módulo ipfw.ko, ya que se cargará automáticamente después de la ejecución de los pasos de la siguiente sección.

3. Preparación final

Antes de reiniciar para cargar el nuevo kernel o los módulos requeridos (de acuerdo con el método

de instalación elegido anteriormente) debes realizar algunos cambios en el archivo de configuración /etc/rc.conf. La regla predeterminada del firewall es rechazar todos los paquetes IP. Inicialmente configuraremos un firewall en modo `open` para verificar que funciona sin ningún problema en relación con el filtrado de paquetes (en el caso de que vaya a ejecutar este procedimiento de forma remota dicha configuración evitará que permanezca aislado de la red). Coloca estas líneas en el archivo /etc/rc.conf:

```
firewall_enable="YES"
firewall_type="open"
firewall_quiet="YES"
firewall_logging="YES"
```

La primera línea activará el firewall (y cargará el módulo ipfw.ko si no está compilado en el kernel), la segunda lo configurará en modo `open` (como se explica en el archivo <[.filename]/etc/rc.firewall), la tercera hará que no se muestren la carga de las reglas y la cuarta habilitará el soporte de logging.

En cuanto a la configuración de las interfaces de red la forma más utilizada es asignar solo una IP a una de las tarjetas de red; el bridge funcionará igualmente, aunque ambas interfaces tengan una o no tengan ninguna IP configurada. En el último caso (IP-less) la máquina bridge quedará aún más oculta, ya que es inaccesible desde la red. Para configurarla, debes iniciar sesión desde la consola o mediante una tercera interfaz de red separada del bridge. A veces durante el inicio del sistema algunos programas requieren acceso a la red, por ejemplo para la resolución del dominio. En este caso es necesario asignar una IP a la interfaz externa (la que está conectada a Internet, donde se encuentra el servidor DNS) ya que el bridge se activará al final del procedimiento de arranque. Esto significa que la interfaz fxp0 (en nuestro caso) debe añadirse en la sección ifconfig del archivo /etc/rc.conf, mientras que fxp0 no. Asignar una IP a ambas tarjetas de red no tiene mucho sentido, a menos que durante el procedimiento de inicio las aplicaciones tengan que acceder a servicios en ambos segmentos Ethernet.

Hay otra cosa importante que hay que saber. Cuando se ejecuta IP over Ethernet, en realidad hay dos protocolos Ethernet en uso: uno es IP, el otro es ARP. ARP realiza la conversión de la dirección IP de un host a su dirección de Ethernet (capa MAC). Para permitir la comunicación entre dos hosts separados por el bridge, es necesario que el bridge reenvíe los paquetes ARP. Dicho protocolo no está incluido en la capa IP, ya que solo existe con IP over Ethernet. El firewall de FreeBSD filtra exclusivamente en la capa IP y, por lo tanto, todos los paquetes no IP (ARP incluido) se reenvían sin ser filtrados, aunque el firewall esté configurado para no permitir nada.

Ahora es el momento de reiniciar el sistema y usarlo como antes: habrá algunos mensajes nuevos sobre el bridge y el firewall, pero el bridge no se activará y el firewall, en el modo `open`, no bloqueará ninguna operación.

Si hay algún problema, debes solucionarlo ahora antes de continuar.

4. Habilitando el bridge

En este momento para habilitar el bridge debes ejecutar los siguientes comandos (no olvide reemplazar los nombres de las dos interfaces de red <[.filename]fxp0 y xl0 por las suyas):

```
# sysctl net.link.ether.bridge.config=fxp0:0,xl0:0
# sysctl net.link.ether.bridge.ipfw=1
# sysctl net.link.ether.bridge.enable=1
```

La primera línea especifica qué interfaces deben ser activadas por el bridge, la segunda habilitará el firewall en el bridge y finalmente la tercera habilitará el bridge.

En este punto deberías poder insertar la máquina entre dos conjuntos de hosts sin comprometer las habilidades de comunicación entre ellas. Si es así, el siguiente paso es añadir las partes `net.link.ether.bridge.[blah]=[blah]` de estas filas al fichero /etc/sysctl.conf para que se ejecuten en el arranque.

5. Configurando el firewall

Ahora es el momento de crear tu propio archivo de configuración con las reglas personalizadas del firewall para proteger la red interna. Te encontrarás con algunas complicaciones porque no todas las funcionalidades del firewall están disponibles en los paquetes bridge. Hay además una diferencia entre los paquetes que están en proceso de reenvío y los paquetes que estás recibiendo la máquina local. En general, los paquetes de entrada pasan por el firewall solo una vez, no dos veces, como suele ser el caso; en realidad se filtran solo después de la recepción, por lo que las reglas que usan `out` o `xmit` nunca coincidirán. Yo utilizo`in via`, que es una sintaxis más antigua pero tiene sentido cuando la lees. Otra limitación es que solo puedes usar solo los comandos `pass` o `reject` para los paquetes filtrados por un bridge. Opciones más complejas como `divert`, `forward` o `reject` no están disponibles. Estas opciones pueden seguir utilizándose, pero solo en el tráfico hacia o desde la propia máquina bridge (si tiene una dirección IP).

El concepto de firewall con estado se incluyó por primera vez en FreeBSD 4.0. Es una gran mejora para el tráfico UDP, el cual generalmente es una solicitud de salida seguida poco después por una respuesta con exactamente el mismo conjunto de direcciones IP y números de puerto (pero obviamente con origen y destino invertidos). Con los firewalls que no mantienen el estado no hay forma de lidiar con este tipo de tráfico en una única sesión. Pero con un firewall que puede "recordar" un paquete saliente de UDP y, durante los próximos minutos, permitir una respuesta el manejo de servicios UDP es trivial. El siguiente ejemplo muestra cómo hacerlo. Es posible hacer lo mismo con los paquetes TCP. Esto le permite evitar algunos ataques de denegación de servicio y y otras maldades, pero también hace que su tabla de estado crezca rápidamente de tamaño.

Veamos una configuración de ejemplo. Lo primero, ten en cuenta que en la parte superior del archivo /etc/rc.firewall ya existen reglas predeterminadas para la interfaz de loopback lo0, por lo que no es necesario preocuparse de ellas. Las reglas personalizadas deben colocarse en un archivo separado (por ejemplo, /etc/rc.firewall.local) y cargarse al inicio del sistema, modificando la línea en el archivo /etc/rc.conf donde definimos el firewall en modo `open`:

```
firewall_type="/etc/rc.firewall.local"
```



Tienes que especificar la ruta *completa*, de otro modo no será cargada con el riesgo

> de quedar aislado de la red.

Para nuestro ejemplo imagina tener el interfaz fxp0 conectado hacia el exterior (Internet) y el xl0 hacia el interior (LAN). La máquina bridge tiene la IP 1.2.3.4 (no es posible que tu ISP te proporcione una dirección como esta, pero para nuestro ejemplo vale).

```
# Things that we have kept state on before get to go through in a hurry
add check-state

# Throw away RFC 1918 networks
add drop all from 10.0.0.0/8 to any in via fxp0
add drop all from 172.16.0.0/12 to any in via fxp0
add drop all from 192.168.0.0/16 to any in via fxp0

# Allow the bridge machine to say anything it wants
# (if the machine is IP-less do not include these rows)
add pass tcp from 1.2.3.4 to any setup keep-state
add pass udp from 1.2.3.4 to any keep-state
add pass ip from 1.2.3.4 to any

# Allow the inside hosts to say anything they want
add pass tcp from any to any in via xl0 setup keep-state
add pass udp from any to any in via xl0 keep-state
add pass ip from any to any in via xl0

# TCP section
# Allow SSH
add pass tcp from any to any 22 in via fxp0 setup keep-state
# Allow SMTP only towards the mail server
add pass tcp from any to relay 25 in via fxp0 setup keep-state
# Allow zone transfers only by the secondary name server [dns2.nic.it]
add pass tcp from 193.205.245.8 to ns 53 in via fxp0 setup keep-state
# Pass ident probes.  It is better than waiting for them to timeout
add pass tcp from any to any 113 in via fxp0 setup keep-state
# Pass the "quarantine" range
add pass tcp from any to any 49152-65535 in via fxp0 setup keep-state

# UDP section
# Allow DNS only towards the name server
add pass udp from any to ns 53 in via fxp0 keep-state
# Pass the "quarantine" range
add pass udp from any to any 49152-65535 in via fxp0 keep-state

# ICMP section
# Pass 'ping'
add pass icmp from any to any icmptypes 8 keep-state
# Pass error messages generated by 'traceroute'
add pass icmp from any to any icmptypes 3
add pass icmp from any to any icmptypes 11
```

```
# Everything else is suspect
add drop log all from any to any
```

Aquellos de vosotros que hayáis configurado firewalls antes habréis notado que faltan algunas cosas. En particular, no hay reglas anti-spoofing, de hecho *no* añadimos:

```
add deny all from 1.2.3.4/8 to any in via fxp0
```

Es decir, descarta paquetes que vienen de fuera pero que dicen que son de nuestra red. Esto es algo que harías de forma habitual para intentar que nadie evita el filtrado de paquetes generando paquetes malvados que parece que provienen del interior. El problema con eso es que hay *al menos* un host en el interfaz exterior que no quieres ignorar: el router. Pero normalmente, el ISP hace anti-spoofing en su router, así que no nos tenemos que preocupar demasiado.

La última regla parece ser un duplicado exacto de la regla predeterminada, es decir, no dejar pasar nada que no esté específicamente permitido. Pero hay una diferencia: todo tráfico sospechoso será registrado.

Hay dos reglas para pasar el tráfico SMTP y DNS hacia el servidor de correo y el servidor de nombres si los tienes. Obviamente el conjunto entero de reglas debería estar personalizado al estilo de cada uno, esto es sólo un ejemplo específico (el formato de las reglas está descrito con detalle en la página de manual de ipfw(8)). Como alternativa es posible especificar la dirección IP en lugar del nombre del host (requerido si la máquina no tiene IP).

Quienes estén acostumbrados a configurar firewalls probablemente también suelan usar una regla `reset` o `forward` para los paquetes ident (`TCP` puerto 113). Por desgracia esta no es una opción válida con el bridge, por lo tanto la mejor opción es simplemente pasarlos a su destino. A menos que la máquina de destino esté ejecutando un demonio ident es realmente inofensivo. La alternativa es eliminar las conexiones en el puerto 113, lo que creará algunos problemas con servicios como IRC (el probe del ident dará timeout).

Lo único raro que puedes haber notado es que existe una regla para permitir que la máquina que hace de bridge hable y otra para los hosts internos. Recuerda que esto sucede porque los dos conjuntos de tráfico tendrán diferentes rutas a través del kernel y del filtro de paquetes. La red interna pasará por el bridge, mientras que la máquina local utilizará el stack normal de IP para hablar. Por lo tanto, cada regla se ocupa de una cosa diferente. Las reglas `in via fxp0` funcionan para ambas rutas. En general, si utiliza las reglas `in via` en todo el filtro, debe añadir una excepción para los paquetes generados localmente, ya que no llegaron a través de ninguna de nuestras interfaces.

6. Colaboradores

Muchas partes de este artículo han sido obtenidas, actualizadas y adaptadas de un texto antiguo sobre bridging, editado por Nick Sayer. Unas cuantas ideas muy inspiradoras vienen de una introducción sobre bridging que escribió Steve Peterson.

Mi más sincero agradecimiento a Luigi Rizzo por la implementación del código de bridge en

FreeBSD y por el tiempo que ha dedicado a responder todas mis preguntas.

Un agradecimiento también a Tom Rhodes, quien revisó mi trabajo de traducción del italiano (el idioma original de este artículo) al inglés.