Pontes de Filtragem (Filtering Bridges)

Alex Dupre <ale@FreeBSD.org >
Revisão: 74f84751b4

2018-09-16 01:21:25 +0000 por Edson Brandi.

Resumo

Geralmente, é útil dividir uma rede física (como uma Ethernet) em dois segmentos separados, sem precisar criar sub-redes, e usar um roteador para vinculá-los. O dispositivo que conecta as duas redes dessa maneira é chamado de ponte (bridge). Um sistema FreeBSD com duas interfaces de rede é suficiente para atuar como uma ponte.

Uma ponte funciona examinando os endereços do nível MAC (endereços Ethernet) dos dispositivos conectados a cada uma de suas interfaces de rede e encaminhando o tráfego entre as duas redes apenas se a origem e o destino estiverem em diferentes segmentos. Sob muitos pontos de vista, uma ponte é semelhante a um switch Ethernet com apenas duas portas.

Índice

1. Por que usar uma ponte de filtragem?

Cada vez mais frequentemente, graças aos custos reduzidos de conexões de banda larga à Internet (xDSL) e também devido à redução de endereços IPv4 disponíveis, muitas empresas estão conectadas à Internet 24 horas por dia e com poucos (às vezes nem mesmo 2) endereços IP. Nestas situações, geralmente é desejável ter um firewall que filtre o tráfego de entrada e de saída de e para a Internet, mas uma solução de filtragem de pacotes baseada em roteador pode não ser aplicável, quer seja devido a problemas de sub-redes, quer seja porque o roteador é de propriedade do provedor de conectividade (ISP), ou porque ele não suporta tais funcionalidades. Nestes cenários, o uso de uma ponte de filtragem é altamente recomendado.

Um firewall baseado em uma ponte de filtragem pode ser configurado e inserido entre o roteador xDSL e seu hub/switch Ethernet sem causar nenhum problema de numeração IP.

2. Como instalar

Adicionar funcionalidades de bridge a um sistema FreeBSD não é difícil. Desde a versão 4.5 é possível carregar tais funcionalidades como módulos ao invés de ter que reconstruir o kernel, simplificando bastante o procedimento. Nas subseções seguintes, explicarei as duas formas de instalação.

Importante

Não siga as duas instruções: um procedimento *exclui* o outro. Selecione a melhor opção de acordo com suas necessidades e habilidades.

Antes de continuar, certifique-se de ter pelo menos duas placas Ethernet compatíveis com o modo promíscuo para recepção e transmissão, pois elas devem ser capazes de enviar pacotes Ethernet com qualquer endereço, não apenas o deles. Além disso, para ter uma boa taxa de transferência, as placas devem ser placas do barramento PCI. As melhores opções ainda são as Intel EtherExpress™ Pro, seguido pela série 3Com® 3c9xx. Para simplificar a configuração do firewall, pode ser útil ter duas placas de diferentes fabricantes (usando drivers diferentes) para distinguir claramente qual interface está conectada ao roteador e qual está conectada à rede interna.

2.1. Configuração do Kernel

Então você decidiu usar o método de instalação mais antigo, e também o mais bem testado. Para começar, você precisa adicionar as seguintes linhas ao seu arquivo de configuração do kernel:

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

A primeira linha adiciona o suporte para o serviço de ponte (bridge), a segunda adiciona o suporte ao firewall e a terceira é referente as funções de registro do firewall.

Agora é necessário compilar e instalar o novo kernel. Você pode encontrar instruções detalhadas na seção Compilando e Instalando um Kernel Personalizado do Handbook do FreeBSD.

2.2. Carregamento de módulos

Se você escolheu usar o método de instalação mais novo e mais simples, a única coisa a fazer agora é adicionar a seguinte linha ao `/boot/loader.conf` :

```
bridge_load="YES"
```

Desta forma, durante a inicialização do sistema, o módulo `bridge.ko` será carregado junto com o kernel. Não é necessário adicionar uma linha semelhante para o módulo `ipfw.ko` , pois ele será carregado automaticamente após a execução das etapas na seção a seguir.

3. Preparação final

Antes de reinicializar para carregar o novo kernel ou os módulos requeridos (de acordo com o método de instalação escolhido anteriormente), você deve fazer algumas alterações no arquivo de configuração `/etc/rc.conf` . A regra padrão do firewall é rejeitar todos os pacotes IP. Inicialmente vamos configurar um firewall `open` (aberto), a fim de verificar sua operação sem qualquer problema relacionado à filtragem de pacotes (no caso de você executar este

procedimento remotamente, tal configuração evitará que você permaneça isolado da rede). Coloque estas linhas no `/etc/rc.conf` :

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

A primeira linha ativará o firewall (e carregará o módulo `ipfw.ko` se ele não estiver compilado no kernel), a segunda irá configurá-lo no modo `open` (como explicado em `/etc/rc.firewall`), a terceira para não mostrar o carregamento de regras e a quarta para ativar o suporte aos logs.

Sobre a configuração das interfaces de rede, a maneira mais usada é atribuir um IP a apenas uma das placas de rede, mas a ponte funcionará igualmente, mesmo que ambas as interfaces ou nenhuma tenha um IP configurado. No último caso (IP-less) a máquina bridge ficará ainda mais oculta, e também inacessível pela rede: para configurá-la será necessário efetuar o login a partir do console ou através de uma terceira interface de rede separada da ponte. Às vezes, durante a inicialização do sistema, alguns programas exigem acesso à rede, digamos, para resolução de nomes de domínio: neste caso, é necessário atribuir um IP à interface externa (aquela conectada à Internet, onde o servidor DNS se encontra), uma vez que a ponte será ativada no final do procedimento de inicialização. Isso significa que a interface `fxp0` (no nosso caso) deve ser mencionada na seção ifconfig do arquivo `/etc/rc.conf` , enquanto o `xl0` não é. Atribuir um IP a ambas as placas de rede não faz muito sentido, a menos que, durante o procedimento de inicialização, os aplicativos acessem os serviços em ambos os segmentos de Ethernet.

Há outra coisa importante a saber. Ao executar o IP sobre Ethernet, existem dois protocolos Ethernet em uso: um é o IP, o outro é o ARP. O ARP faz a conversão do endereço IP de um host em seu endereço Ethernet (camada MAC). Para permitir a comunicação entre dois hosts separados pela ponte, é necessário que a ponte envie pacotes ARP. Esse protocolo não está incluído na camada IP, uma vez que ele existe apenas com IP sobre Ethernet. O firewall do FreeBSD filtra exclusivamente na camada IP e, portanto, todos os pacotes não-IP (ARP incluso) serão encaminhados sem serem filtrados, mesmo que o firewall esteja configurado para não permitir nada.

Agora é hora de reiniciar o sistema e usá-lo como antes: haverá algumas novas mensagens sobre a ponte e o firewall, mas a ponte não será ativada e o firewall, estando no modo `open`, não evitará operações.

Se houver algum problema, você deve resolvê-los agora antes de prosseguir.

4. Habilitando a ponte

Neste ponto, para habilitar a ponte, você tem que executar os seguintes comandos (tendo a perspicácia para substituir os nomes das duas interfaces de rede `fxp0` e `xl0` com as suas próprias):

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

A primeira linha especifica quais interfaces devem ser ativadas pela ponte, a segunda habilitará o firewall na ponte e, finalmente, a terceira habilitará a ponte.

Nota

Se você tem o FreeBSD 5.1-RELEASE ou anterior, as variáveis do sysctl são escritas de forma diferente. Veja bridge(4) para detalhes.

Neste ponto você deve ser capaz de inserir a máquina entre dois conjuntos de hosts sem comprometer quaisquer habilidades de comunicação entre eles. Em caso afirmativo, o próximo passo é adicionar as partes

`net.link.ether.bridge. [blah]=[blah]` dessas linhas ao arquivo `/etc/sysctl.conf` , para que eles sejam executados na inicialização.

5. Configurando o Firewall

Agora é hora de criar seu próprio arquivo com regras de firewall personalizadas, a fim de proteger a rede interna. Haverá alguma complicação em fazer isso porque nem todas as funcionalidades do firewall estão disponíveis em pacotes em ponte. Além disso, há uma diferença entre os pacotes que estão sendo encaminhados e os pacotes que estão sendo recebidos pela máquina local. Em geral, os pacotes de entrada passam pelo firewall apenas uma vez, não duas vezes, como é normalmente o caso; na verdade eles são filtrados somente após o recebimento, portanto, as regras que usam `out` ou `xmit` nunca darão match. Pessoalmente, eu uso `in via` que é uma sintaxe antiga, mas uma que tem sentido quando você a lê. Outra limitação é que você está restrito a usar somente comandos `pass` ou `drop` para os pacotes filtrados por uma ponte. Coisas sofisticadas como `divert`, `forwar` ou `reject` não estão disponíveis. Essas opções ainda podem ser usadas, mas apenas no tráfego para ou a partir da própria máquina ponte (se ela tiver um endereço IP).

O conceito de filtragem stateful é novo no FreeBSD 4.0. Esta é uma grande melhoria para o tráfego UDP, que normalmente é um pedido que sai, seguido pouco depois por uma resposta com exatamente o mesmo conjunto de endereços IP e números de porta (mas com origem e destino invertidos, é claro). Para firewalls que não possuem manutenção de estado, quase não há como lidar com esse tipo de tráfego como uma sessão única. Mas com um firewall que pode "lembrar" de um pacote UDP de saída e, nos próximos minutos, permitir uma resposta, o manuseio de serviços UDP é trivial. O exemplo a seguir mostra como fazer isso. É possível fazer a mesma coisa com pacotes TCP. Isso permite que você evite alguns ataques de negação de serviço e outros truques desagradáveis, mas também faz com que sua tabela de estado cresça rapidamente em tamanho.

Vamos ver um exemplo de configuração. Note primeiro que no topo do `/etc/rc.firewall` já existem regras padrão para a interface de loopback `lo0`, então não devemos mais precisar delas. Regras customizadas devem ser colocadas em um arquivo separado (digamos, `/etc/rc.firewall.local`) e carregadas na inicialização do sistema, modificando a linha de `/etc/rc.conf` onde definimos o firewall `open`:

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

Importante

Você tem que especificar o caminho *completo*, caso contrário ele não será carregado com o risco de permanecer isolado da rede.

Para nosso exemplo, imagine ter a interface `fxp0` conectada para o exterior (Internet) e a `xl0` para o interior (LAN). A máquina ponte tem o IP `1.2.3.4` (não é possível que o seu ISP possa lhe dar um endereço assim, mas para nosso exemplo ele é bom).

```
# 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 slave 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
```

Aqueles de vocês que já configuraram firewalls antes podem notar algumas coisas que estão faltando. Em particular, não há regras anti-spoofing, na verdade nós *não* adicionamos:

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

Ou seja, dropar os pacotes que estão vindo do lado de fora dizendo ser da nossa rede. Isso é algo que você normalmente faria para ter certeza de que alguém não tentaria escapar do filtro de pacotes, gerando pacotes nefastos que parecem ser de dentro. O problema é que existe *pelo menos* um host na interface externa que você não deseja ignorar: o roteador. Mas geralmente ISP tem anti-spoofs em seu roteador, então não precisamos nos incomodar muito.

A última regra parece ser uma duplicata exata da regra padrão, ou seja, não deixa passar nada que não seja especificamente permitido. Mas há uma diferença: todo tráfego suspeito será registrado.

Existem duas regras para passar o tráfego SMTP e o do DNS para o servidor de e-mail e o servidor de nomes, se você os tiver. Obviamente, todo o conjunto de regras deve ser definido de acordo com as suas preferências pessoais, isso é apenas um exemplo específico (o formato da regra é descrito com precisão na página de manual do ipfw(8)). Note que, para que o “relay” e o “ns” funcionem, as pesquisas de serviço de nomes devem funcionar *antes* da ponte ser ativada. Este é um exemplo de quando você precisa ter certeza de que definiu o IP na placa de rede correta. Como alternativa, é possível especificar o endereço IP em vez do nome do host (necessário se a máquina não tiver IP).

As pessoas que estão acostumadas a configurar firewalls provavelmente também estão acostumadas a ter uma regra `reset` ou `forward` para pacotes ident (TCP porta 113). Infelizmente, esta não é uma opção aplicável com a ponte, então o melhor é simplesmente passá-las ao seu destino. Enquanto essa máquina de destino não estiver executando um daemon ident, isso é relativamente inofensivo. A alternativa é descartar as conexões na porta 113, o que criará alguns problemas com serviços como por exemplo o IRC (o probe do ident irá dar timeout).

A única outra coisa que é um pouco estranha e que você pode ter notado é que existe uma regra para deixar a máquina ponte falar, e outra para os hosts internos. Lembre-se que isso ocorre porque os dois conjuntos de tráfego terão caminhos diferentes através do kernel e do filtro de pacotes. A rede interna passará pela ponte, enquanto a

máquina local usará a pilha IP normal para falar. Assim, as duas regras lidam com casos diferentes. As regras `in via fxp0` funcionam nos dois caminhos. Em geral, se você usar as regras `in via` em todo o filtro, precisará abrir uma exceção para pacotes gerados localmente, porque eles não vieram em nenhuma de nossas interfaces.

6. Colaboradores

Muitas partes deste artigo foram tiradas, atualizadas e adaptadas de um texto antigo sobre pontes, editado por Nick Sayer. Um par de inspirações deve-se a uma introdução sobre pontes de Steve Peterson.

Um grande obrigado ao Luigi Rizzo pela implementação do código de ponte (bridge) no FreeBSD e pelo tempo que ele dedicou a mim respondendo a todas as minhas perguntas relacionadas.

Agradeço também a Tom Rhodes, que olhou para o meu trabalho de tradução do italiano (a língua original deste artigo) para o inglês.