Mosty filtrujące

Alex Dupre

sysadmin@alexdupre.com

$FreeBSD: release/8.4.0/pl_PL.ISO8859-2/articles/filtering-bridges/article.xml 39632 2012-10-01 11:56:00Z gabor $

$FreeBSD: release/8.4.0/pl_PL.ISO8859-2/articles/filtering-bridges/article.xml 39632 2012-10-01 11:56:00Z gabor $

Częstokroć zdarza się, że trzeba podzielić jedną sieć fizyczną (np. Ethernet) na dwa oddzielne segmenty, nie tworząc przy tym podsieci, a oba segmenty połączyć ze sobą ruterem. Urządzenie łączące w ten sposób dwie sieci nazywane jest mostem. Komputer z FreeBSD posiadający dwa interfejsy sieciowe może z powodzeniem pracować jako most.

Zadaniem mostu jest analizowanie adresów MAC (adresów ethernetowych) należących do urządzeń przyłączonych do obu interefejsów sieciowych, a następnie przekazywaniu danych pomiędzy obiema sieciami tylko wtedy, gdy nadawca i odbiorca należą do innych segmentów. Pod wieloma względami most przypomina przełącznik ethernetowy wyposażony w jedynie dwa porty.

1. Dlaczego korzysta się z mostów filtrujących?

Dzięki obniżającym się kosztom szerokopasmowych połączeń internetowych (xDSL), jak również z powodu niewielkiej liczby dostępnych adresów IPv4, coraz częściej zdarza się, że firmy dysponują stałym połączeniem z Internetem, nie posiadając przy tym zbyt wielu adresów IP. W takiej sytuacji przydatne jest stosowanie firewalla filtrującego pakiety wysyłane do Internetu i z niego nadchodzące. Może się jednak zdarzyć, że filtrowania pakietów na poziomie rutera nie da się zrealizować, na przykład ze względu na podział sieci, lub dlatego, że to dostawca usług interentowych jest właścicielem rutera, bądź też sam ruter nie umożliwia takiego rozwiązania. Wtedy właśnie wskazane jest skorzystanie z mostu filtrującego.

Firewall będący jednocześnie mostem może być wstawiony pomiędzy ruter xDSL a koncentrator/przełącznik ethernetowy. Jego konfiguracja nie wymaga zajmowania się numeracją IP.

2. Instalacja

W FreeBSD włączenie funkcji mostu nie jest trudnym przedsięwzięciem. Począwszy od wydania 4.5 owe funkcje mogą być dołączone jako moduły, nie trzeba więc przebudowywać jądra, co jest znacznym udogodnieniem. Poniżej opisuję obydwa sposoby.

> **WAŻNE:** *Nie należy* postępować według obu poniższych przepisów: skorzystanie z jednego z nich *wyklucza* korzystanie z drugiego. Wybór powinien zależeć od własnych potrzeb i możliwości.

Przed rozpoczęciem należy upewnić się, że dysponujemy przynajmniej dwiema kartami sieciowymi zdolnymi do pracy w trybie pośredniczenia zarówno przy odbiorze, jak i nadawaniu; karty będą wysyłać pakiety opatrzone niekoniecznie ich własnymi adresami. Co więcej, by osiągnąć dobrą wydajność, powinny być to karty PCI obsługujące zarządzanie magistralą. Do takich należą karty Intel EtherExpress Pro, a także karty 3Com z serii 3c9xx. Dla uproszczenia konfiguracji firewalla pożytecznym okazać się może posiadanie kart dwóch różnych producentów (korzystających z innych sterowników), by łatwiej było odróżnić interfejs podłączony do rutera od interfejsu połączonego z siecią wewnętrzną.

2.1. Konfigurowanie jądra

Pierwsza z metod jest starsza, lecz sprawdzona. Na początek należy dodać następujące wiersze do pliku konfiguracyjnego jądra:

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

Pierwszy wiersz włącza do jądra obsługę mostu, drugi obsługę firewalla, a trzeci funkcje rejestrujące firewalla.

Teraz trzeba skompilować i zainstalować nowe jądro. Szczegółowy opis tych czynności znaleźć można w Podręczniku FreeBSD, w części "Building and Installing a Custom Kernel (../../books/handbook/kernelconfig-building.html)".

2.2. Ładowanie modułów

Ta metoda instalacji jest nowsza i prostsza, polega jedynie na dodaniu poniższego wiersza do `/boot/loader.conf`:

```
bridge_load="YES"
```

W efekcie podczas ładowania systemu wraz z jądrem zostanie załadowany moduł `bridge.ko`. Nie trzeba dodawać analogicznego wiersza dla modułu `ipfw.ko`, gdyż zostanie on załadowany automatycznie po wykonaniu czynności opisanych w następnej części.

3. Przygotowanie do pracy

Przed ponownym uruchomieniem systemu oraz załadowaniem nowego jądra lub modułów (w zależności od wybranej metody instalacji), trzeba jeszcze dokonać kilku zmian w pliku konfiguracyjnym `/etc/rc.conf`. Domyślną regułą firewalla jest zatrzymywanie wszystkich pakietów IP. Zaczniemy od skonfigurowania firewalla `otwartego`, by sprawdzić jego działanie przy wyłączonym filtrowaniu (dzięki temu maszyna będzie mogła utrzymać połączenie z siecią, co jest niezbędne w przypadku, gdy konfiguracja przeprowadzana jest poprzez sieć). W pliku `/etc/rc.conf` należy umieścić poniższe wpisy:

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

Pierwszy wiersz powoduje uruchomienie firewalla (ładowany jest moduł `ipfw.ko`, jeśli nie został wkompilowany do jądra), w drugim ustawiany jest `otwarty` tryb jego pracy (zgodnie z opisem w `/etc/rc.firewall`). Następny wiersz nakazuje nie pokazywać ładowanych reguł, a w ostatnim włączane jest rejestrowanie.

Interfejsy sieciowe są najczęściej skonfigurowane tak, że tylko jedna z kart sieciowych ma przypisany adres IP, jednakże most działa tak samo również wtedy, gdy adresy przypisane są do obu kart lub nie są przypisane do żadnej z nich. W tym ostatnim przypadku (brak IP) maszyna pełniąca rolę mostu będzie jeszcze bardziej ukryta, gdyż nie będzie dostępna z sieci; dostęp do niej możliwy będzie poprzez konsolę lub trzeci interfejs sieciowy odseparowany od mostu. Niekiedy dostęp do sieci potrzebny jest programom uruchamianym podczas ładowania systemu, na przykład do określenia nazwy domeny. W takiej sytuacji adres IP należy przydzielić interfejsowi zewnętrznemu (czyli temu, który połączony jest z Internetem, gdzie znajduje się serwer DNS), ponieważ most będzie uruchomiony dopiero w końcowej fazie uruchamiania systemu. Oznacza to, że interfejs `fxp0` (jak w przykładzie) musi być uwzględniony w sekcji ifconfig pliku `/etc/rc.conf`, w przeciwieństwie do interfejsu `xl0`. Przydzielanie adresów IP obu kartom sieciowym nie ma raczej sensu, chyba, że podczas uruchamiania systemu programy potrzebują dostępu do obu segmentów sieci.

Należy mieć na uwadze, że w sieci IP opartej na Ethernecie działają w rzeczywistości dwa protokoły: jednym jest oczywiście IP, drugim jest ARP. Zadaniem protokołu ARP jest przekształcenie adresu IP stacji na jej adres ethernetowy (adres MAC). By możliwa była komunikacja między dwoma stacjami znajdującymi się po dwóch stronach mostu, pakiety ARP muszą być przekazywane przez most. Protokół ARP nie jest składnikiem warstwy IP, ponieważ jest używany tylko wtedy, gdy IP działa w sieci Ethernet. Filtrowanie pakietów przez firewall w FreeBSD dotyczy warstwy IP, więc pakiety innego typu (w tym także ARP) będą przekazywane dalej bez filtrowania, nawet jeśli konfiguracja firewalla nakazuje blokowanie wszystkiego.

Można już uruchomić system ponownie i korzystać z niego jak dotychczas. Pojawią się nowe komunikaty dotyczące mostu i firewalla, most jednak nie będzie jeszcze pracować, natomiast firewall będzie działać w trybie `otwartym`, bez jakiegokolwiek blokowania.

Jeśli pojawią się jakiekolwiek problemy, należy się z nimi uporać przed przystąpieniem do kolejnego etapu pracy.

4. Uruchamianie mostu

Uruchomienie mostu polega na wykonaniu następującej sekwencji poleceń (nazwy przykładowych interfejsów sieciowych `fxp0` i `xl0` należy zastąpić nazwami własnych interfejsów):

```
# sysctl net.link.ether.bridge_cfg=fxp0:0,xl0:0
# sysctl net.link.ether.bridge_ipfw=1
# sysctl net.link.ether.bridge=1
```

Pierwsze polecenie wskazuje interfejsy obsługiwane przez most, drugie włącza firewalla, wreszcie trzecie polecenie włącza sam most.

Tak skonfigurowana maszyna może zostać włączona między dwie grupy połączonych w sieć komputerów bez zakłócania ich wzajemnej komunikacji. Jeśli to się powiedzie, można dodać do pliku `/etc/sysctl.conf` wpisy `net.link.ether.`*`[coś]=[coś]`* zgodne z powyższymi, by zostały uwzględnione przy ładowaniu systemu.

5. Konfiguracja firewalla

Następnym krokiem jest przygotowanie reguł firewalla, zabezpieczających sieć wewnętrzną. Wiąże się to z pewnymi utrudnieniami, gdyż nie wszystkie możliwości firewalla mogą być wykorzystywane w przypadku pakietów

przechodzących przez most. Trzeba też wiedzieć, że między pakietami przekazywanymi, a odbieranymi przez maszynę lokalną jest pewna różnica. Pakiety przychodzące przechodzą przez firewall tylko raz, a nie dwa razy jak w zwykłych warunkach. Mówiąc dokładniej, są one filtrowane tylko przy odbiorze, tak więc reguły zawierające `out` lub `xmit` będą bezużyteczne. Ja osobiście używam starszej składni `in via`, którą rozsądniej się czyta. Trzeba również pamiętać, że filtrując pakiety przechodzące przez most, można używać tylko poleceń `pass` lub `drop`. Bardziej wymyślne polecenia, jak `divert`, `forward` czy `reject` są niedozwolone. Można z nich korzystać tylko w odniesieniu do pakietów wysyłanych przez maszynę mostu lub do niej przychodzących (jeśli oczywiście ma ona adres IP).

Nowością w FreeBSD 4.0 jest filtrowanie z utrzymywaniem stanu. Jest ono znacznym ułatwieniem obsługi komunikacji przez UDP, polegającej najczęściej na wysłaniu żądania, a za chwilę odebraniu odpowiedzi, z takimi samymi adresami IP i numerami portów (przy czym nadawca i odbiorca są oczywiście zamienieni miejscami). Praktycznie nie da się potraktować takiej wymiany jako pojedynczej sesji, posługując się firewallem nie przechowującym informacji o stanie połączenia. Jednakże gdy firewall potrafi „zapamiętać” wychodzący pakiet UDP i zezwolić na odpowiedź w ciągu kilku następnych minut, wówczas zarządzanie komunikacją UDP staje się dziecinnie proste. Jak to zrobić, pokazuje poniższy przykład. Podobnie można traktować pakiety TCP, chroni to przed niektórymi atakami przez uniemożliwienie działania oraz innymi figlami, prowadzi jednak do szybkiego rozrastania się tablicy stanów.

Spójrzmy na przykładową konfigurację. Zwróćmy uwagę, że na początku pliku `/etc/rc.firewall` umieszczono domyślne reguły dla interfejsu pseudosieci `lo0`, nie trzeba się więc już nimi przejmować. Inne reguły powinny być umieszczone w oddzielnym pliku (np. `/etc/rc.firewall.local`), który byłby dołączany podczas ładowania systemu dzięki zmianie w pliku `/etc/rc.conf` tego wiersza, w którym typ firewalla był zdefiniowany jako `otwarty`:

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

> **WAŻNE:** Należy tu podać *pełną* ścieżkę, w przeciwnym razie plik nie zostanie załadowany, co grozi utratą dostępu do sieci.

Na potrzeby przykładu przyjmujemy, że interfejs `fxp0` połączony jest z Internetem, natomiast `xl0` z siecią wewnętrzną (LAN). Adres IP maszyny mostu to `1.2.3.4` (w rzeczywistości dostawca usług interenetowych nie mógłby przydzielić adresu klasy A, jednak świetnie nadaje się on jako przykład).

```
# Szybkie przepuszczanie pakietów, których stan został zapamiętany
add check-state

# Blokada sieci z RFC 1918
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

# Maszyna będąca mostem może wysyłać co tylko zechce
# (jeśli maszyna nie ma adresu IP, pomiń poniższe wiersze)
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

# Stacje sieci wewnętrznej mogą wysyłać co tylko zechcą
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

# Protokół TCP
# Przepuszczanie SSH
add pass tcp from any to any 22 in via fxp0 setup keep-state
# Przepuszczanie SMTP jedynie do serwera poczty
add pass tcp from any to relay 25 in via fxp0 setup keep-state
# Informacje o obszarach mogą być przesyłane tylko przez podrzędny
# serwer nazw [dns2.nic.it]
add pass tcp from 193.205.245.8 to ns 53 in via fxp0 setup keep-state
# Przepuszczanie zapytań ident - takie rozwiązanie jest lepsze
# niż oczekiwanie na przekroczenie czasu
add pass tcp from any to any 113 in via fxp0 setup keep-state
# Przepuszczenie zakresu portów dynamicznych
add pass tcp from any to any 49152-65535 in via fxp0 setup keep-state

# Protokół UDP
# Przepuszczanie zapytań DNS jedynie do serwera DNS
add pass udp from any to ns 53 in via fxp0 keep-state
# Przepuszczenie zakresu portów dynamicznych
add pass udp from any to any 49152-65535 in via fxp0 keep-state

# Protokół ICMP
# Przepuszczanie 'pingów'
add pass icmp from any to any icmptypes 8 keep-state
# Przepuszczanie komunikatów o błędach generowanych przez 'traceroute'
add pass icmp from any to any icmptypes 3
add pass icmp from any to any icmptypes 11

# Wszystko inne jest podejrzane
add drop log all from any to any
```

Czytelnicy mający już doświadczenie z konfiguracją firewalla mogą zwrócić uwagę na brak pewnych rzeczy. W szczególności, brakuje reguł zapobiegających podszywaniu się. Rzeczywiście, wśród powyższych reguł *nie było* takiej:

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

Nakazuje ona odrzucanie pakietów, które nadchodzą z zewnątrz, a udają, że są z sieci wewnętrznej. Jest to zwykle stosowane w celu zabezpieczenia się przed próbami prześliznięcia się przez filtrowanie, polegającymi na tworzeniu fałszywych pakietów, wyglądających jak wysłane z sieci wewnętrznej. Kłopot w tym, że jest *co najmniej* jedna stacja połączona z interfejsem zewnętrznym, której nie można zignorować: jest nią ruter. Zwykle jednak ochrona przed podszywaniem się realizowana jest przez dostawcę usług internetowych na jego ruterze, nie trzeba się więc tym przejmować.

Ostatnia reguła jest bardzo podobna do reguły przyjmowanej domyślnie, czyli odrzucania wszystkiego, co nie jest ściśle dozwolone. Jest jednak różnica: wszystko, co jest podejrzane, jest rejestrowane.

Dwie reguły odpowiedzialne są za przekazywanie pakietów SMTP i DNS do serwera poczty i serwera nazw, o ile takowe serwery są. Zestaw reguł powinien być oczywiście dostosowany do własnych potrzeb, tutaj pokazany jest tylko pewien przykład (składnia reguł jest dokładnie opisana w dokumentacji systemowej ipfw(8)). Trzeba mieć na uwadze, że aby poprawnie działały „relay” i „ns”, wymagane jest, by zapytania DNS działały *zanim* pracę

rozpoczyna most. Dzięki temu można też się przekonać, czy adres IP został przypisany do właściwiej karty sieciowej. Alternatywnym rozwiązaniem jest wpisanie adresu IP zamiast nazwy stacji (jest to jedyna możliwość w przypadku, gdy maszyna nie ma adresu IP).

Osoby, które już kiedyś miały pewne doświadczenia z konfiguracją firewalla, przyzwyczajone są zapewne do reguł `reset` lub `forward` dla pakietów ident (port TCP o numerze 113). Niestety, w przypadku mostu takie rozwiązanie nie wchodzi w grę, najlepiej jest po prostu przepuścić owe pakiety do ich adresata. Jest to stosunkowo niegroźne, gdy adresat ma wyłączoną usługę ident. Można także blokować połączenia z portem 113, co powoduje pewne problemy np. z usługą IRC (ponieważ zapytanie ident musi przekroczyć czas oczekiwania).

Niezrozumiała może wydać się obecność oddzielnych reguł, z których jedne zezwalaja na wysyłanie maszynie będącej mostem, drugie natomiast stacjom sieci wewnętrznej. Jest tak dlatego, że pakiety wysyłane przez maszynę lokalną docierają do filtra inną drogą niż pakiety wysłane z sieci wewnętrznej. Te ostatnie muszą przejść przez most, natomiast pakiety wysłane lokalnie trafiają na stos IP maszyny. Osobne zestawy reguł obsługują obydwa przypadki. Z kolei reguły zawierające `in via fxp0` odnoszą się do obu rodzajów pakietów. Mówiąc ogólnie, pisząc reguły `in via` trzeba zrobić wyjątek dla pakietów wysyłanych z lokalnej maszyny, ponieważ one nie przyszły przez żaden z interfejsów.

6. Podziękowania

Duża część niniejszego artykułu zaczerpnięta została ze starego dokumentu o mostach autorstwa Nicka Sayera. Inspiracją było również wprowadzenie do tematyki mostów napisane przez Steve'a Petersona.

Bardzo dziękuję Luigiemu Rizzo za implementację kodu mostu w FreeBSD, jak również za czas poświęcony na odpowiedzi na moje pytania.

Dziękuję też Tomowi Rhodesowi, który zechciał przyjrzeć się mojemu tłumaczeniu tego artykułu z włoskiego (w takim języku napisany był oryginał) na angielski.