PC WORLD

Wydanie: Extra Linux
Archiwum

Linux jako firewall

PCWorld

Ochrona przed atakami z Internetu to jedno z podstawowych zadań administratora systemu, a coraz częściej także każdego użytkownika komputera. Zintegrowany z jądrem Linuksa filtr pakietów umożliwia budowę wydajnej zapory, a niejako przy okazji pozwala też na pewną kontrolę nad poczynaniami użytkowników w lokalnej sieci.

Gdy nawiązujesz połączenie z Internetem, automatycznie wystawiasz się na potencjalne zagrożenie. Ostatnio praktycznie nie ma tygodnia, by telewizja, radio i prasa nie donosiły o nowych wirusach, robakach, trojanach czy też lukach zabezpieczeń w systemach operacyjnych.

Problem można, na szczęście, stosunkowo łatwo rozwiązać i za naprawdę niewygórowaną sumę kupić router z firewallem, który będzie chronić lokalną sieć. Mimo to jest wiele powodów, aby samemu bliżej zapoznać się z problemem. Jednym z nich mogą być choćby pieniądze zaoszczędzone na zakupie routera.

Nie sposób jednak przecenić - szczególnie w dużych sieciach - znacznie większych możliwości, jakie daje samodzielnie zestawiony firewall pracujący z Linuksem. To nie tylko znacznie bardziej elastyczna konfiguracja, ale także o wiele dokładniejsze protokołowanie wszystkich zdarzeń, co umożliwia z jednej strony łatwiejsze rozpoznawanie ataków, łącznie z śledzeniem wielkości strumienia napływających danych. Firewall taki pozwala również efektywnie ograniczyć połączenia wychodzące - funkcja, która w równym stopniu uniemożliwia użytkownikom niepożądane współdzielenie plików, jak i blokuje komunikację z czyhającymi ewentualnie w sieci trojanami.

1. Konfiguracja - najlepiej ręcznie

Sercem każdego firewalla pracującego z Linuksem jest program iptables. Sam program nie wykonuje funkcji właściwych dla firewalla, a raczej tworzy interfejs do zintegrowanej z jądrem Linuksa architektury filtru sieciowego. Filtr ten, utworzony w ramach projektu Netfilter (www.netfilter.org), zawiera wszystkie funkcje potrzebne do budowy firewalla.


Narzędzie YaST2 umożliwia konfigurację jedynie tych ustawień, które dotyczą lokalnego komputera.

Funkcjami filtru sterują różne opcje, które należy przypisać narzędziu iptables w postaci parametrów. W ten sposób można tworzyć kolejne etapy, przez które musi przejść pakiet danych, aby mógł zostać przesłany dalej, przetworzony czy odrzucony. Tak utworzonymi regułami zarządza "procesor" firewalla, rezydujący w pamięci głównej komputera. Dzięki temu można wprowadzać zmiany również w trakcie pracy.

Niestety, w tym wypadku zalety są jednocześnie wadami: architektura filtru sieciowego jest tak elastyczna, że początkujący użytkownik z dużym trudem poznaje mechanizm jego działania. Oprócz tego nie ma głównego pliku konfiguracyjnego, który stanowiłby zbiór obowiązujących reguł. Dlatego w Internecie można znaleźć wiele programów pomocniczych, które mają ułatwić konfigurację firewalla. W wielu przypadkach są to udane próby, ale mają tę samą wadę - gdy coś nie działa właściwie, użytkownik nadal nie wie, jak zlokalizować błąd.

Ochrona nie pojedynczego komputera, ale na przykład całej sieci przekracza możliwości wielu tych rozwiązań. Dotyczy to również firewalla zintegrowanego z SuSE Linux 9.0. Narzędzie YaST2 pozwala jedynie na konfigurację ustawień lokalnego komputera. Pokonanie tego ograniczenia wymaga ręcznej edycji pliku konfiguracyjnego /etc/sysconfig/SuSEfirewall2. To zadanie praktycznie niewykonalne

dla osoby niewtajemniczonej. Pozostaje zatem wziąć się do nauki. Z czasem okaże się, że nie jest to aż tak trudne, jak się w pierwszej chwili wydawało.

Podstawowe drogi pakietu danych z punktu widzenia routera - wejście, wyjście i przekazanie.

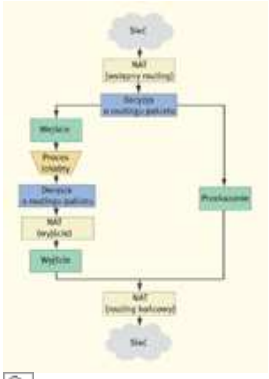
2. Sposób pracy iptables

Na potrzeby naszego przykładu przyjmiemy założenie, że mamy do czynienia z komputerem, który pracuje jako router między Internetem a siecią lokalną. Na początek rozpatrzmy, z jakimi rodzajami transmisji możemy mieć do czynienia. Są trzy możliwości:

- Nadszedł pakiet danych przeznaczony właśnie do tego komputera.
- Dane, które nadeszły, przeznaczone są do innego komputera w sieci i muszą być przekazane dalej.
- Aplikacja działająca na komputerze sama wysyła dane.

Odpowiednio do tego można zdefiniować trzy główne zadania: przyjmowanie (input), przekazywanie (forward) i wysyłanie (output) danych. Filtr sieciowy udostępnia listy reguł o takich samych nazwach; narzędzie iptables może się odwoływać do tych list. Same listy nazywane są często łańcuchami (chains).

Takie podejście jest dużym uproszczeniem, które nie uwzględnia dwóch przypadków specjalnych. Pierwszy to zastosowanie tłumaczenia adresów (Network Address Translation, NAT). Tutaj router "udaje" przed innymi komputerami w Internecie, że jest wyłącznym nadawcą pakietów pochodzących z lokalnej sieci. W tym celu do każdego pakietu wychodzącego wstawia jako adres nadawcy przydzielony mu przez dostawcę usług internetowych adres IP. Aby nadchodzące z Internetu odpowiedzi mogły trafić do właściwego odbiorcy, muszą zostać zaopatrzone we właściwy adres - inny niż adres routera, bo w przeciwnym razie router przetwarzałby pakiety, zamiast przesyłać je do odbiorcy w lokalnej sieci. W przypadku pakietów przychodzących trzeba zmienić adres na właściwy adres odbiorcy w sieci LAN, by mógł on trafić do lokalnego komputera; w przypadku pakietów wychodzących trzeba na koniec procesu przetwarzania zmienić adres nadawcy na adres routera.



Skuteczne tłumaczenie adresów (NAT) wymaga dodatkowych przekształceń pakietów.

Aby umożliwić również i to, iptables zawiera oprócz łańcuchów standardowej tabeli dodatkowo filtry tabeli nat, która z kolei zawiera oprócz łańcuchów input, output i forward także oba łańcuchy prerouting oraz postrouting.

Tłumaczenie adresów (NAT) to nie jedyna możliwość manipulacji pakietów. Procedura o nazwie "mangling" (tzw. wikłanie nazw) pozwala na przypisanie specjalnych atrybutów poszczególnym pakietom. Za pomocą tej funkcji można zrealizować pewne zadania specjalne, na przykład rudymentarną obsługę Quality of Services także w protokole IPv4.

Aby całość zadziałała, wikłanie nazw można przeprowadzać na różnych etapach przetwarzania pakietu: przed wywołaniem funkcji NAT, przed zadziałaniem filtrów wejściowych, a także przed filtrem wyjściowym i przed filtrem dalszego przekazania. Funkcja wikłania pakietów ma jednak niewielkie znaczenie dla pracy firewalla, a więc łańcuchy input, output, forward, prerouting i postrouting, znajdujące się w tabeli mangle można po prostu pozostawić bez zmian.

Nie tylko tabele filtrów mangle i nat mają swoje nazwy. Ma ją również tabela standardowa - filter. Jeżeli więc w wywołaniu iptables nie podasz konkretnej tabeli, nowe reguły zostaną automatycznie umieszczone w tabeli filtrów.

3. Żelazna reguła

Przedstawiliśmy ogólny przebieg filtrowania pakietów, czas zatem na wyjaśnienie, do czego służą poszczególne filtry. Najpierw jednak trzeba wyjaśnić generalną zasadę przetwarzania pakietów. Dla każdego pakietu trzeba odpowiedzieć na pytania: skąd pochodzi pakiet, dokąd zmierza i jakie jest jego zadanie (do czego służy, jaką niesie informację). To właśnie z grubsza podstawowe kryteria, stosowane przez iptables. Na pytanie "skąd", można odpowiedzieć na dwa sposoby: podając interfejs, do którego trafił pakiet, lub adres IP nadawcy. W przypadku "dokąd" można określić, który interfejs ma być wyjściowy lub kto ma być odbiorcą danych. O zadaniu pakietu decyduje port docelowy, pozwalający z reguły zidentyfikować aplikację, do której jest przeznaczony. Dodatkowe informacje o


Wikłanie pakietów wpływa na dane na wielu etapach przetwarzania.

zamiarach pakietu zawarte są w jego statusie.

Inna interpretacja żelaznej reguły może mieć taką postać: "Kto i co może zrobić oraz za pomocą czego?". Właśnie na to podstawowe pytanie trzeba sobie odpowiedzieć w trakcie tworzenia reguł firewalla. Najprostsza odpowiedź brzmi: każdy użytkownik sieci LAN może wysyłać do Internetu dowolne zapytania, odpowiedzi mają trafiać do pytającego, natomiast niezamówione pakiety przychodzące z zewnątrz mają być usuwane. Dodatkowo warto zastosować zasadę, że to, co nie zostało wyraźnie dozwolone, jest zabronione. Biorąc pod uwagę powyższe ustalenia, można już zaproponować pierwszą wersję skryptu firewalla. Zaloguj się jako użytkownik root i załóż plik /root/fwtest1 (listing 1).

Listing 1

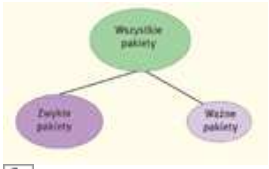
```
#!/bin/bash
LANIF="eth0"
INETIF="ppp0"
IPTABLES=`which iptables`
$IPTABLES -A INPUT -i $INETIF -m state
- -state NEW,INVALID -j DROP
$IPTABLES -A FORWARD -i $INETIF -m state
- -state NEW,INVALID -j DROP
$IPTABLES -A POSTROUTING -t nat -o $INETIF
-j MASQUERADE
echo 1 > /proc/sys/net/ipv4/ip_forward
```

Za pomocą tego skryptu zrealizujesz stosunkowo prosty router NAT. Zakładamy, że interfejs eth0 odpowiada za komunikację z siecią LAN, natomiast ppp0 za komunikację z systemami w Internecie. Rozróżnienie między dozwolonymi a zabronionymi pakietami następuje na podstawie stanu połączenia.

W systemie filtru sieciowego pakiet może się znajdować w jednym z czterech stanów: nowy (new), znany (established), pokrewny (related) i nieważny (invalid). Ostatni typ można bez zastanowienia odrzucić. To samo przeważnie dotyczy pakietów o statusie nowy, ponieważ stanowią nowe zapytanie, a więc nie zostały zamówione przez klienta z sieci lokalnej. Pozostają pakiety znane, czyli należące do bieżącej sesji, oraz pokrewne, czyli związane z bieżącą sesją. Pakiety danych o takim statusie są przyjmowane i przetwarzane w ramach powyższych ustawień zarówno przez lokalny komputer (-A INPUT), jak i przez inne komputery w sieci (-A FORWARD). Wiersz z poleceniem -A POSTROUTING -t nat powoduje, że filtr sieciowy nadaje wszystkim pakietom wychodzącym przez ppp0 (-o $INETIF) adres IP routera. To jest właśnie Network Address Translation.

4. Ostrzejszy wariant

Opisany powyżej skrypt kieruje się zasadą, że dozwolone jest to, co nie zostało zabronione. W przypadku firewalla, który ma stanowić ochronę sieci lokalnej, z pewnością nie jest to optymalne rozwiązanie. Odwrotne podejście nie jest zasadniczo trudniejsze w realizacji, czego dowodzi skrypt fwtest2 (listing 2).



Za pomocą Hierarchical Token Bucket można podzielić pasmo na kanały o różnej szybkości.

Różnica polega przede wszystkim na tym, że reguła działania łańcuchów input i forward została zmieniona z "zezwalaj na wszystko" na "odrzucaj wszystko" (-P ... DROP). W rezultacie oba łańcuchy filtrów trzeba nie tylko poinformować o tym, że mają akceptować przychodzące z Internetu pakiety o statusie established lub related. Trzeba też wyraźnie zezwolić na przyjmowanie pakietów pochodzących z lokalnej sieci (-i $LANIF). O reguły wyjściowe nie trzeba się martwić, ponieważ w tym względzie standardowe zasady postępowania pozostały niezmienione. Ten łańcuch reguł akceptuje więc wszystko, co zostało mu dostarczone.

Listing 2

```
#!/bin/bash
LANIF="eth0"
INETIF="ppp0"
IPTABLES=`which iptables`
$IPTABLES -P INPUT DROP
$IPTABLES -P FORWARD DROP
$IPTABLES -A INPUT -i $INETIF -m state
- -state ESTABLISHED,RELATED -j ACCEPT
$IPTABLES -A INPUT -i $LANIF -j ACCEPT
$IPTABLES -A FORWARD -i $INETIF -m state
- -state ESTABLISHED,RELATED -j ACCEPT
$IPTABLES -A FORWARD -i $LANIF -j ACCEPT
```

```
$IPTABLES -A POSTROUTING -t nat
-o $INETIF -j MASQUERADE
echo 1 > /proc/sys/net/ipv4/ip_forward
```

5. Własne serwery

Opisana do tej pory konfiguracja umożliwia komunikację komputerów w sieci LAN z maszynami w Internecie. Ale czasami również komputery z Internetu muszą się komunikować z komputerami w sieci lokalnej. Najprostszy przykład to witryna internetowa z ofertą firmy lub serwer FTP do wymiany danych. Aby możliwy był dostęp do tych serwerów, trzeba skierować dane przeznaczone do odpowiednich usług do właściwych komputerów w lokalnej sieci. Innymi słowy, trzeba zrobić dziury w firewallu. Wymaga to zmian w routingu NAT oraz w przesyłaniu (forwarding). Widać to wyraźnie w przykładowym skrypcie do serwera internetowego i serwera FTP (listing 3).

Listing 3

```
#!/bin/bash
LANIF="eth0"
INETIF="ppp0"
IPTABLES=`which iptables`
FTP_SRV="192.168.27.100"
HTTP_SRV="192.168.27.101"
$IPTABLES -P INPUT DROP
$IPTABLES -P FORWARD DROP
$IPTABLES -A PREROUTING -t nat -i $INETIF -p tcp
- -dport 21 -j DNAT - -to-destination $FTP_SRV
$IPTABLES -A PREROUTING -t nat -i $INETIF -p tcp
- -dport 80 -j DNAT - -to-destination $HTTP_SRV
$IPTABLES -A INPUT -i $INETIF -m state
- -state ESTABLISHED,RELATED -j ACCEPT
$IPTABLES -A INPUT -i $LANIF -j ACCEPT
$IPTABLES -A FORWARD -i $INETIF -d $FTP_SRV
-p tcp - -dport 21 -j ACCEPT
$IPTABLES -A FORWARD -i $INETIF -d $HTTP_SRV
-p tcp - -dport 80 -j ACCEPT
$IPTABLES -A FORWARD -i $INETIF -m state
- -state ESTABLISHED,RELATED -j ACCEPT
$IPTABLES -A FORWARD -i $LANIF -j ACCEPT
$IPTABLES -A POSTROUTING -t nat -o $INETIF
-j MASQUERADE
echo 1 > /proc/sys/net/ipv4/ip_forward
```

Powyższy skrypt nie tylko działa, ale sygnalizuje od razu problem wymagający rozwiązania przez każdego użytkownika firewalla: jeżeli zezwalasz na dostęp dla komputerów z zewnątrz, musisz z góry zdecydować, na którym z lokalnych komputerów ma działać dana usługa. Drogie firewalle obchodzą ten problem, stosując specjalne filtry.

Umożliwiają one w razie potrzeby otwarcie firewalla dla klienta, a po upływie określonego czasu dostęp zostaje ponownie zamknięty. Taki firewall obserwuje porty docelowe wychodzących danych. Jeżeli klient adresuje pakiet do określonego portu, firewall w oczekiwaniu na odpowiedź tworzy odpowiednie reguły przekazywania danych w innym porcie, określonym przez administratora. Jeżeli port odpowiedzi nie zostanie wykorzystany w określonym czasie, firewall zamyka wejście do sieci lokalnej. Takiej funkcji nie da się obecnie zrealizować za pomocą iptables.

6. Definiowanie własnych łańcuchów

Można natomiast bez żadnego problemu zrealizować własne łańcuchy reguł w tabeli filtrów. Jest to praktyczne, np. gdy ruch sieciowy ma być rejestrowany w dziennikach. Poniższy skrypt pozwoli sprawdzić, z jakich usług oprócz HTTP korzystają użytkownicy w sieci lokalnej (listiing 4).

Prostota rozwiązania została okupiona pewnym brakiem funkcjonalności, na przykład komunikacja z zewnętrznym serwerem FTP powoduje bardzo wiele wpisów do dziennika. Powyższy przykład doskonale jednak ilustruje inną właściwość iptables: to, że pakiety można przekierować bez ich utraty. W przykładowym skrypcie założono tylko jeden łańcuch reguł log_it, składający się tylko z jednej reguły - nakazu protokołowania. Gdy pakiet przejdzie przez ten łańcuch, filtr sieciowy kieruje go z powrotem do łańcucha, w którym rozpoczął "objazd". Problem nadmiaru wpisów można rozwiązać na dwa sposoby: dodając kolejną regułę, która wyklucza FTP z obowiązku protokołowania, lub ustalając limit wpisów. Można też zastosować kombinację obu tych reguł (listing 5).

Listing 4

```
#!/bin/bash
LANIF="eth0"
INETIF="ppp0"
IPTABLES=`which iptables`
FTP_SRV="192.168.27.100"
HTTP_SRV="192.168.27.101"
$IPTABLES -P INPUT DROP
$IPTABLES -P FORWARD DROP
$IPTABLES -N log_it
$IPTABLES -A log_it -j LOG
- -log-prefix "Nie HTTP: "
$IPTABLES -A PREROUTING -t nat -i $INETIF -p tcp
- -dport 21 -j DNAT - -to-destination $FTP_SRV
$IPTABLES -A PREROUTING -t nat -i $INETIF -p tcp
- -dport 80 -j DNAT - -to-destination $HTTP_SRV
$IPTABLES -A INPUT -i $INETIF -m state
- -state ESTABLISHED,RELATED -j ACCEPT
$IPTABLES -A INPUT -i $LANIF -j ACCEPT
$IPTABLES -A FORWARD -p tcp ! - -dport 80
-j log_it
$IPTABLES -A FORWARD -i $INETIF -d $FTP_SRV
-p tcp - -dport 21 -j ACCEPT
$IPTABLES -A FORWARD -i $INETIF -d $HTTP_SRV
-p tcp - -dport 80 -j ACCEPT
$IPTABLES -A FORWARD -i $INETIF -m state
- -state ESTABLISHED,RELATED -j ACCEPT
$IPTABLES -A FORWARD -i $LANIF -j ACCEPT
$IPTABLES -A POSTROUTING -t nat -o $INETIF
-j MASQUERADE
echo 1 > /proc/sys/net/ipv4/ip_forward
```

Listing 5

```
$IPTABLES -A log_it -p tcp - -dport 21 -m limit
- -limit 3/minute -j LOG - -log-prefix "SSH :"
$IPTABLES -A log_it -p tcp ! - -dport 21 -j LOG
- -log-prefix "Nie HTTP: "
```

7. Ograniczanie połączeń wychodzących

Sprawa nieco się komplikuje, gdy chcesz objąć restrykcjami ruch wychodzący. W tym celu można ustawić regułę łańcucha wyjściowego na "drop" i zezwolić na przykład tylko na pakiety do serwerów internetowych (listing 6).

Listing 6

```
#!/bin/bash
LANIF="eth0"
INETIF="ppp0"
IPTABLES='which iptables'
FTP_SRV="192.168.27.100"
HTTP_SRV="192.168.27.101"
$IPTABLES -P INPUT DROP
$IPTABLES -P FORWARD DROP
$IPTABLES -N log_it
$IPTABLES -A log_it -j LOG
- -log-prefix "Nie HTTP: "
$IPTABLES -A PREROUTING -t nat -i $INETIF -p tcp
- -dport 21 -j DNAT - -to-destination $FTP_SRV
$IPTABLES -A PREROUTING -t nat -i $INETIF -p tcp
- -dport 80 -j DNAT - -to-destination $HTTP_SRV
$IPTABLES -A INPUT -i $INETIF -m state
- -state ESTABLISHED,RELATED -j ACCEPT
$IPTABLES -A INPUT -i $LANIF -j ACCEPT
$IPTABLES -A FORWARD -p tcp ! - -dport 80 -j log_it
$IPTABLES -A FORWARD -i $INETIF -d $FTP_SRV
-p tcp - -dport 21 -j ACCEPT
$IPTABLES -A FORWARD -i $INETIF -d $HTTP_SRV
-p tcp - -dport 80 -j ACCEPT
$IPTABLES -A FORWARD -i $INETIF -m state
- -state ESTABLISHED,RELATED -j ACCEPT
```

```
$IPTABLES -A FORWARD -i $LANIF -p tcp
- -dport 80 -j ACCEPT
$IPTABLES -A POSTROUTING -t nat -o $INETIF
-j MASQUERADE
echo 1 > /proc/sys/net/ipv4/ip_forward
```

Powyższe rozwiązanie ma jednak dodatkowe niepożądane skutki, ponieważ uniemożliwia jednocześnie dostęp do zewnętrznych serwerów pocztowych, a zatem kompletnie odcina tę drogę komunikacji. W zależności od tego, jak odbierasz i wysyłasz pocztę, musisz zastosować odpowiednie rozwiązania. Najprostszy jest wariant, w którym serwer pocztowy w sieci LAN odbiera pocztę z zewnętrznego systemu i tam ją również wysyła. W takim przypadku obaj partnerzy komunikacji są znani i można ich uwzględnić bezpośrednio w konfiguracji (listing 7).

Listing 7

```
MAIL_INT="192.168.27.200"
MAIL_EXT="62.146.34.194"
$IPTABLES -A FORWARD -i $LANIF -s $MAIL_INT
-d $MAIL__EXT -p tcp - -dport 110 -j ACCEPT
$IPTABLES -A FORWARD -i $LANIF -s $MAIL_INT
-d $MAIL_EXT -p tcp - -dport 25 -j ACCEPT
```

Jak wyraźnie widać, restrykcyjny firewall wymaga znajomości wszystkich aplikacji i używanych przez nie portów. Tylko pod takim warunkiem możesz utworzyć sprawnie działający system. Należy też uwzględnić różnorodne sposoby pracy aplikacji. Jeżeli na przykład twój wewnętrzny serwer pocztowy wysyła wiadomości wychodzące bezpośrednio do odpowiedniego serwera pocztowego odbiorcy, musisz nieco zmienić czwarty wiersz (listing 8).

Listing 8

```
$IPTABLES -A FORWARD -i $LANIF -s $MAIL_INT
-p tcp - -dport 25 -j ACCEPT
```

8. Optymalizacja ruchu

Filtr sieciowy można również wykorzystać do przyspieszenia ruchu danych. Zasadniczo w obecnie stosowanej wersji IPv4 żaden rodzaj pakietów nie jest uprzywilejowany. W dodatku przepustowość łącza do Internetu jest z reguły mniejsza od przepustowości sieci LAN. Jeżeli wielu użytkowników jednocześnie wysyła dane, tworzy się kolejka, w której pakiety ustawiają się w kolejności nadchodzenia. W takiej sytuacji jest bardzo prawdopodobne, że tuż przed wysłaniem krótkiego potwierdzenia nadejścia pakietu (ACK) w kolejce znajdzie się wiadomość pocztowa z bardzo dużym załącznikiem. Zanim pakiet ACK zostanie w końcu wysłany, komputer po drugiej stronie przestanie czekać na potwierdzenie i ponownie wyśle dane. W skrajnie niekorzystnej sytuacji kolejny pakiet ACK może spotkać ten sam los, a użytkownik zaczyna mieć wrażenie, że połączenie internetowe przestało działać albo pracuje z dużymi zakłóceniami.

Rozwiązaniem tego problemu jest Hierarchical Token Bucket (HTB, luxik.cdi.cz/~devik/qos/htb/), już zintegrowany z nową wersją jądra. Za jego pomocą można niemal dowolnie podzielić szerokość pasma wychodzącego na kanały. Do zarządzania kanałami, HTB wykorzystuje strukturę drzewa, w której każdy podkanał ma jednoznacznie przypisany kanał nadrzędny.

HTB nie tylko przydziela kanałom gwarantowaną minimalną szerokość pasma, ale jednocześnie zwalnia niewykorzystaną przepustowość jednych kanałów na rzecz innych. Aby podzielić standardowe łącze DSL o przepustowości 128 kb/s na dwa kanały, potrzebne są polecenia zamieszczone w listingu 9.

Listing 9

```
tc qdisc add dev eth1 root handle 1: htb default 10
tc class add dev eth1 parent 1: classid 1:1 htb
rate 126kbps
tc class add dev eth1 parent 1:1 classid 1:10 htb
rate 110kbps ceil 126kbps
tc class add dev eth1 parent 1:1 classid 1:11 htb
rate 16kbps ceil 126kbps
```

Polecenia te tworzą tak zwany Qdisc do interfejsu sieciowego eth1, do którego dołączony jest modem DSL. Tworzona jest ponadto klasa źródłowa, o szybkości transferu danych tuż poniżej przepustowości łącza. Dzięki temu wąskim gardłem staje się nie modem DSL, a karta sieciowa. Na karcie można jednak utworzyć odpowiednie kanały i kontrolować kolejność wysyłania pakietów do modemu DSL. Polecenia tworzą jeszcze dwie podklasy, z których jedna standardowo otrzymuje przepustowość 110 kb/s, zaś druga 16 kb/s. Znajdujący się na końcu parametr ceil 126kbps informuje system, że każda z podklas może korzystać z pełnej przepustowości pasma, jeśli jest ona dostępna.

Teraz pozostaje jeszcze zadbać o to, żeby pakiety danych przechodzące przez router otrzymywały odpowiednią klasę. Wstępne sortowanie zapewnia już powyższa definicja klas. Parametr default 10 w pierwszym wierszu polecenia stanowi, że wszystkie nieoznaczone inaczej pakiety trafiają do klasy 10 kanału należącego do Qdisc 1 - czyli do kanału, do którego przypisana jest większa szerokość pasma. Wystarczy zatem odfiltrować pakiety, które mają trafić do drugiego kanału. W tym celu należy najpierw poinformować filtr HTB, w jaki sposób ma je rozpoznać (listing 10).

Listing 10

```
tc filter add dev eth1 parent 1: protocol ip prio 1
handle 1 fw classid 1:11
```

Ważna jest tu przede wszystkim liczba znajdująca się za parametrem handle, w tym przypadku 1. W przypadku pakietów wychodzących, które mają przejść przez specjalną kolejkę, konieczne jest nadanie im stosownego znacznika. Oto potrzebne polecenia dla filtru sieciowego znajdziesz w listingu 11.

Listing 11

```
$IPTABLES -A FORWARD -t mangle -p tcp
- -dport 21 -j MARK - -set-mark 1
$IPTABLES -A FORWARD -t mangle -p tcp
- -dport 22 -j MARK - -set-mark 1
$IPTABLES -A FORWARD -t mangle -p tcp
- -tcp-flags ACK ACK -j MARK - -set-mark 1
```

Dzięki temu wszystkie pakiety połączeń SSH, informacje sterujące FTP i potwierdzenia otrzymanych danych trafią do specjalnie na to zarezerwowanej klasy, ponieważ w wyniku działania parametru -set-mark 1 otrzymają znacznik, którego oczekuje HTB.

9. Ostrzeżenie na koniec

Postawienie działającego firewalla nie jest ostatecznie trudne - problem w tym, czy jest on bezpieczny. Informacje zawarte w tej części to zaledwie podstawy wiedzy na temat filtru pakietów Linuksa i sposobu jego działania. Jeżeli nie przestudiujesz gruntownie dodatkowej literatury, bardzo możliwe, że twój firewall będzie mniej skuteczny od urządzenia kupionego w sklepiku z elektroniką za rogiem.

Przetestuj gruntownie swój firewall - i to nie w swojej sieci, w której pracujesz na co dzień (tzw. środowisko produkcyjne), lecz w środowisku testowym. Wszystkie powyższe przykłady zostały tak przygotowane, że nie powinno być niespodzianek, jednak nie możemy udzielić żadnej gwarancji. W razie kłopotów właśnie ty będziesz najprawdopodobniej kozłem ofiarnym. Pomyśl o tym i działaj z rozwagą.

1234dalej »
Spis treści: Extra Linux

Komentarze

- **dodaj komentarz** |

Ten artykuł nie ma jeszcze żadnych komentarzy. Twój może być pierwszy...