Linux 2.4 NAT HOWTO

Rusty Russell, lista pocztowa `netfilter@lists.samba.org`

Wersja oryginalna: 1.15, 2001/07/29 03:59:37

Oryginał tego dokumentu znajduje się pod adresem: http://netfilter.samba.org/unreliable-guides/NAT-HOWTO/index.html

Tłumaczenie: Łukasz Bromirski, `l.bromirski@mr0vka.eu.org`

Wersja tłumaczenia: 2.1, $Date: 2002/03/19 22:34:11 $

Oryginał tłumaczenia znajduje się pod adresem: http://mr0vka.eu.org/tlumaczenia/linux24-nat.html

---

Dokument ten opisuje jak wykonać maskaradę, transparentne proxy, przekazywanie portów i inne formy NAT w pracy z linuksowymi kernelami 2.4.

---

1. Wprowadzenie

Witam, szanowny czytelniku.

Właśnie zanurzasz się w fascynujący (choć czasami koszmarny) świat NAT: Translacji Adresów Sieciowych, a ten dokument HOWTO będzie dla Ciebie w miarę dokładnym przewodnikiem, do tego i nie tylko, tematu dla kernela linuksa 2.4.

W linuksie 2.4 dodano infrastrukturę do manipulowania pakietami nazwaną 'netfilter'. Warstwa zbudowana na jej fundamentach zapewnia NAT, kompletnie przepisany w porównaniu do poprzednich kerneli.

2. Gdzie znajduje się oficjalna strona WWW i lista?

Są trzy oficjalne strony:

- Dzięki Filewatcher.
- Dzięki Zespołowi Samba i SGI.
- Dzięki Harald'owi Welte.

By odnaleźć oficjalną listę pocztową netfilter, zajrzyj na Serwer list Samba.

2.1 Co to jest Translacja Adresów Sieciowych (NAT)?

Zwykle, pakiety podróżują od źródła (takiego jak twój komputer) do swojego przeznaczenia (takiego jak www.gnumonks.org) przez wiele różnych połączeń: około 19 stąd gdzie jestem, w Australii. Żaden z tych węzłów nie zmienia twojego pakietu: po prostu przesyła go dalej.

Jeśli jeden z tych węzłów miałby wykonać NAT, zmieniłby adres źródłowy lub przeznaczenia w trakcie jak pakiet przechodziłby przez niego. Jak sobie pewnie wyobrażasz, nie jest to sposób w jaki system miał działać, i w związku z tym używanie NAT jest zawsze trochę ryzykowne. Zwykle węzeł który wykonuje NAT zapamięta sposób w jaki zmienił pakiet i kiedy nadejdzie odpowiedź na niego, wykona odwrotne czynności tak, by wszystko działało.

2.2 Po co miałbym stosować NAT?

W idealnym świecie, nie byłoby powodu. Ale, głównymi powodami są:

Modemowe połączenie do Internetu

Większość ISP (dostawców internetowych) daje ci jeden IP gdy się do nich wdzwaniasz. Możesz wysyłać pakiety z dowolnym adresem źródłowym, ale tylko odpowiedzi przeznaczone do tego konkretnego IP który został Ci przydzielony dotrą z powrotem do Ciebie. Jeśli chcesz używać wielu komputerów (jeśli posiadasz na przykład sieć w domu) i zapewnić im dostęp do Internetu, będziesz potrzebował NAT.

Aktualnie jest to najczęstsze zastosowanie NAT, znane pod nazwą 'masquerading' w świecie linuksa. Nazywam to SNAT, ponieważ zmieniasz adres źródłowy pierwszego pakietu (*Source-NAT*, czyli *Źródłowa Translacja Adresów* - przypomnienie tłum.).

Wiele serwerów

Czasami chcesz zmienić sposób w jaki pakiety rozprowadzane są w twojej sieci. Zwykle dzieje się tak ponieważ masz jeden adres IP, ale chciałbyś by ludzie mogli się dostać do maszyn w środku sieci tak jakby miały 'prawdziwe' adresy IP. Jeśli zmienisz adres przeznaczenia przychodzących pakietów, możesz to osiągnąć. Ten typ NAT nazywany był w dotychczasowych wersjach Linuksa **przekazywaniem portów** (ang. *port-forwarding*).

Najczęstszym rodzajem tego sposobu jest **rozkładanie obciążenia** (ang. *load-sharing*), mapowanie wskazuje wtedy na listę maszyn które dzielą się nadchodzącymi wywołaniami. Jeśli zamierzasz robić coś takiego na poważnie, warto zajrzeć pod adres linuksowego wirtualnego serwera.

Transparentne Proxy

Czasami chcesz udawać że każdy pakiet który przechodzi przez twój komputer z Linuksem, przeznaczony jest do jakiejś aplikacji uruchomionej właśnie na nim. Używa się tego do tworzenia transparentnych proxy: proxy to program który pośredniczy w wymianie informacji między twoją siecią a światem zewnętrznym. Transparentnych dlatego, że twoja sieć nie wie nawet że rozmawia z proxy, dopóki oczywiście proxy działa.

Można tak skonfigurować Squida, i nazywa się to **przekierowywaniem** (ang. *redirection*) lub transparentnym proxy we wcześniejszych wersjach Linuksa.

3. Dwa typy NAT

Dzielę NAT na dwa rodzaje: **Źródłowy NAT** (SNAT) i **Docelowy NAT** (DNAT).

SNAT ma miejsce wtedy, gdy zmieniasz adres źródłowy pierwszego pakietu: tzn. kiedy zmieniasz adres maszyny z której inicjowane jest połączenie. SNAT wykonywany jest zawsze po routingu (ang. *post-routing*), tuż przed tym jak pakiet opuści maszynę po kablu. Masquerading jest specjalizowaną formą SNAT.

DNAT ma miejsce wtedy, gdy zmieniasz adres docelowy pierwszego pakietu: tzn. kiedy zmieniasz adres maszyny do której ma dotrzeć połączenia. DNAT wykonywany jest zawsze przed routingiem (ang. *pre-routing*), kiedy pakiet właśnie został odebrany z kabla. Przekazywanie portów, rozkładanie obciążenia i transparentne proxy to wszystko różne rodzaje DNAT.

4. Szybkie przejście z kerneli 2.0 i 2.2

Przepraszam wszystkich nadal zszokowanych po przejściu z 2.0 (ipfwadm) do 2.2 (ipchains). Są dobre i złe wiadomości

Przede wszystkim, możesz używać dalej ipchains i ipfwadm tak jak kiedyś. By to zrobić, musisz włączyć do kernela (`insmod`) moduły 'ipchains.o' lub 'ipfwadm.o' które znajdują się w najnowszej dystrybucji pakietu netfilter. Moduły mogą być załadowane tylko rozłącznie (zostałeś ostrzeżony) i nie powinny być łączone z innymi modułami netfilter.

Kiedy już zainstalujesz jeden z tych modułów, możesz używać ipchains i ipfwadm tak jak do tej pory, z

następującymi różnicami:

- Ustawienie czasu wygaśnięcia dla połączeń maskaradowanych opcjami `ipchains -M -S` lub `ipfwadm -M -s` nic nie robi. Ponieważ jednak czasy wygasania dla nowej infrastruktury NAT są dużo dłuższe, nie powinno mieć to znaczenia.
- Pola `init_seq`, `delta` i `previous_delta` w listingu szczegółowym maskarady są zawsze ustawione na zero.
- Zerowanie i listowanie liczników w tym samym momencie '`-Z -L`' nie działa: liczniki nie zostaną wyzerowane.
- Warstwa zapewniająca kompatybilność wstecz nie skaluje się zbyt dobrze dla dużych ilości połączeń: nie używaj jej na firmowej bramie!

Hakerzey mogą również zauważyć, że:

- Możesz się teraz bindować do portów 61000-65095 nawet jeśli działa maskarada. Jej kod zakładał do tej pory że w tym zakresie portów mógł się rządzić, więc programy nie mogły go używać.
- (Nieudokumentowana) opcja '`getsockname`', której programy pracujące jako transparentne proxy mogły użyć by uzyskać prawdziwe adresy przeznaczenia połączeń już nie działa.
- (Nieudokumentowana) opcja '`bind-to-foreign-address`' również nie została zaimplementowana; była używana to stworzenia kompletnej iluzji transparentnego proxy.

4.1 Ja chcę tylko maskarady! Ratunku!

Większość ludzi chce. Jeśli masz dynamicznie przydzielany IP podczas wdzwaniania się po PPP (jeśli nie wiesz o co chodzi to zapewne tak właśnie jest), możesz po prostu powiedzieć swojej maszynie że wszystkie pakiety wychodzące z twojej sieci wewnętrznej powinny być zmieniane tak, by wyglądały jak pakiety wychodzące z komputera wdzwaniającego się.

```
# Załaduj moduł NAT (usuwa inne).
modprobe iptable_nat

# W tabeli NAT (-t nat), dodaj regułę (-A) po routingu
# (POSTROUTING) dla wszystkich pakietów wychodzących przez ppp0 (-o ppp0)
# która mówi by prowadzić maskaradowanie połączenia (-j MASQUERADE).
iptables -t nat -A POSTROUTING -o ppp0 -j MASQUERADE

# Włącz przekazywanie IP
echo 1 > /proc/sys/net/ipv4/ip_forward
```

Zauważ że nie prowadzimy tutaj żadnego filtrowania pakietów: jeśli chcesz je zastosować, przeczytaj Filtrowanie Pakietów HOWTO a w szczególności sekcję: 'Łączenie NAT i filtrowania pakietów'.

4.2 A co z ipmasqadm?

Istnieje dużo mniejsza liczba użytkowników, więc specjalnie nie przejmowałem się wsteczną kompatybilnością. Możesz po prostu użyć '`iptables -t nat`' by wykonywać przekazywanie portów. Na przykład, dla Linuksa 2.2 mógłbyś zrobić:

```
# Linux 2.2
# Przekaż pakiety TCP kierowane do portu 8080 na 1.2.3.4 do 192.168.1.1 na port 80
ipmasqadm portfw -a -P tcp -L 1.2.3.4 8080 -R 192.168.1.1 80
```

A teraz zrobisz:

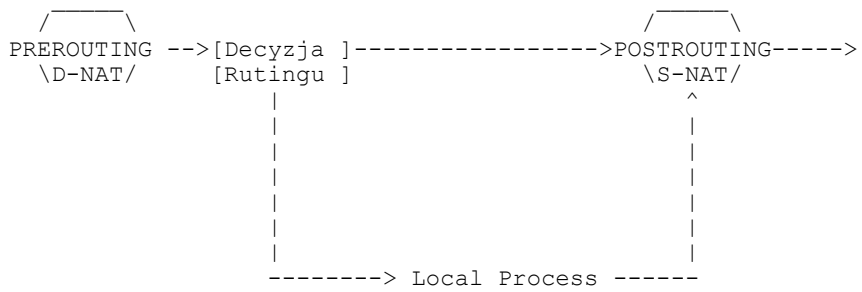
```
# Linux 2.4
# Dodaj regułę przed routingiem (-A PREROUTING) do tabeli NAT (-t nat) mówiącą, że
# pakiety TCP (-p tcp) kierowane do 1.2.3.4 (-d 1.2.3.4) na port 8080 (--dport 8080
# mają być zmapowane (-j DNAT) na 192.168.1.1, port 80 (--to 192.168.1.1:80).
iptables -A PREROUTING -t nat -p tcp -d 1.2.3.4 --dport 8080 \
        -j DNAT --to 192.168.1.1:80
```

5. Kontrolowanie tego, co poddawać NAT

Musisz stworzyć takie reguły NAT, by kernel wiedział które połączenia poddawać modyfikacjom i jak to robić. By tego dokonać, używamy bardzo szczegółowego narzędzia `iptables` i wskazujemy kernelowi jak zmienić tabelę NAT przez podanie opcji '`-t nat`'.

Tabela NAT zawiera dwie listy zwane 'łańcuchami' (ang. *chains*): każda reguła w takiej liście jest sprawdzana dopóki któraś nie pasuje. Te dwa łańcuchy nazywają się: PREROUTING (dla DNAT, ponieważ pakiety najpierw do nas docierają), POSTROUTING (dla SNAT, po którym pakiety opuszczają nasz system) i OUTPUT (dla DNAT, dla pakietów generowanych lokalnie).

Poniższy rysunek ilustrowałby to dosyć dobrze gdybym miał choć trochę talentu artystycznego:



W każdym z węzłów powyżej, gdy dociera do niego pakiet, sprawdzane jest z jakim połączeniem jest skojarzony. Jeśli jest to nowe połączenie, sprawdzamy odpowiedni łańcuch w tabeli NAT by sprawdzić co mamy z nim zrobić. Odpowiedź której udziela tabela dotyczyć będzie wszystkich przyszłych pakietów dla tego połączenia.

5.1 Proste przykłady z użyciem iptables

`iptables` pobiera pewną liczbę standardowych opcji tak jak podano to niżej. Wszystkie opcje z podwójnym znakiem minus mogą być skrócone, tak długo jak `iptables` może je rozróżnić od innych możliwych znaczeń. Jeśli twój kernel ma wkompilowaną obsługę iptables jako moduł, musisz go najpierw załadować: '`insmod ip_tables`'.

Najważniejszą opcją jest wybierająca tabelę, '`-t`'. Dla wszystkich operacji które dotyczą NAT, używać będziesz opcji '`-t nat`'. Drugą ważną opcją jest '`-A`', która służy do dodawania nowej reguły na koniec łańcucha (np. '`-A POSTROUTING`'), oraz '`-I`', która służy do dodawania jej na początku (np. '`-I PREROUTING`').

Możesz podawać adresy źródłowe ('`-s`' lub '`--source`') i docelowe ('`-d`' lub '`--destination`') pakietów których ma dotyczyć NAT. Opcjom tym może towarzyszyć pojedyńczy adres IP (np. 192.168.1.1), nazwa (np. www.gnumonks.org) lub adres sieci (np. 192.168.1.0/24 albo 192.168.1.0/255.255.255.0).

Możesz również wskazać interfejs wejściowy ('`-i`' lub '`--in-interface`') lub wyjściowy ('`-o`' lub '`--out-interface`'), ale który konkretnie zależy również od tego który łańcuch wskażesz: w łańcuchu PREROUTING możesz wskazać tylko interfejs wejściowy, a w łańcuchu POSTROUTING możesz wskazać tylko interfejs wyjściowy. Jeśli użyjesz niewłaściwego, `iptables` zwrócą błąd.

5.2 Dokładniejsze informacje o wyborze pakietów do modyfikowania

Powiedziałem wyżej, że możesz wskazać adresy źródłowe i docelowe. Jeśli ominiesz adres źródłowy, pasować będzie każdy adres źródłowy. Jeśli pominiesz adres docelowy, pasować będzie każdy adres docelowy.

Możesz również wskazać konkretny protokół ('`-p`' lub '`--protocol`'), taki jak TCP czy UDP; pasować będą pakiety tylko z tego protokołu. Głównym powodem dla którego chciałbyś wskazać konkretny protokół jest to, że z każdym z nich skojarzone są dodatkowe opcje: w szczególności '`--source-port`' i '`--destination-port`' (które można skrócić do '`--sport`' i '`--dport`').

Opcje te pozwalają ci określić o jakie pakiety chodzi z dokładnością do portu źródłowego i docelowego. Jest to użyteczne w przekazywaniu na przykład wywołań do serwerów WWW (port TCP 80 lub 8080) i jednocześnie pomijaniu innych pakietów.

Opcjom tym musi towarzyszyć opcja '-p' (która ma ten skutek uboczny, że doładowywuje współdzielone rozszerzenie biblioteczne dla konkretnego protokołu). Do wskazania numeru portu możesz używać numerów, lub nazw z pliku /etc/services.

Wszystkie sposoby na które możesz wskazać określony typ pakietów wyszczególnione zostały w podręczniku (man iptables).

6. Określanie jak zmieniać pakiety

Wiesz już jak wybrać pakiety do modyfikacji. By skompletować naszą regułę, musimy poinformować jeszcze kernel co zrobić z pasującymi pakietami.

6.1 Źródłowy NAT

Chcesz prowadzić Źródłowy NAT, czyli zmieniać adresy źródłowe połączeń na coś innego. Dzieje się to w łańcuchu POSTROUTING, tuż przed tym jak pakiety zostają wysłane; jest to bardzo ważny szczegół, ponieważ oznacza że cokolwiek innego na tej maszynie oglądając ten pakiet (routing, filtrowanie pakietów) będzie widziało go w stanie jeszcze niezmienionym. Oznacza to, że można również używać opcji '-o'.

Źródłowy NAT wykonuje się przez wpisanie '-j SNAT', i dodanie opcji '--to-source', która określa adres lub grupę docelowych adresów IP; opcjonalnie można również wskazać numer lub zakres numerów portów (ale tylko dla protokołów TCP i UDP).

```
## Zmień adres źródłowy na 1.2.3.4.
# iptables -t nat -A POSTROUTING -o eth0 -j SNAT --to 1.2.3.4

## Zmień adres źródłowy na 1.2.3.4, 1.2.3.5 lub 1.2.3.6
# iptables -t nat -A POSTROUTING -o eth0 -j SNAT --to 1.2.3.4-1.2.3.6

## Zmień adresy źródłowy 1.2.3.4, porty z zakresu 1-1023
# iptables -t nat -A POSTROUTING -p tcp -o eth0 -j SNAT --to 1.2.3.4:1-1023
```

Maskarada

Jest to specjalizowana odmiana Źródłowego NATu: powinna być używana tylko jeśli masz dynamicznie przydzielany adres IP, tak jak podczas wdzwaniania się (jeśli masz statycznie przydzielony IP, użyj SNAT tak jak opisano to powyżej).

Nie musisz podawać adresów źródłowych by wykonać maskaradę: użyje ona adresu źródłowego interfejsu przez który pakiety będą opuszczały maszynę. Ale co ważniejsze, jeśli połączenie wdzwaniane zostanie zamknięte, połączenia (które i tak już zostały zerwane) zostaną zapomniane, co oznacza mniej zgrzytów w momencie gdy ponownie się wdzwonisz z nowym IP.

```
## Maskaraduj wszystko wychodzące przez ppp0.
# iptables -t nat -A POSTROUTING -o ppp0 -j MASQUERADE
```

6.2 Docelowy NAT

Wykonywany jest w łańcuchu PREROUTING, zaraz po tym jak pakiet zostaje odebrany, co oznacza że cokolwiek innego na tej maszynie będzie pracować z pakietem (routing, filtrowanie pakietów), zobaczy go już skierowanego do 'prawdziwego' adresu docelowego. Oznacza to również, że można również używać opcji '-i' (interfejs wejściowy).

Docelowy NAT wskazuje się przez podanie opcji '-j DNAT' i '--to-destination' która określa pojedyńczy adres lub zakres adresów IP; opcjonalnie można wskazać port lub zakres portów (ale tylko dla protokołów TCP i UDP).

```
## Zmień adresy docelowe na 5.6.7.8
# iptables -t nat -A PREROUTING -i eth0 -j DNAT --to 5.6.7.8

## Zmień adresy docelowe 5.6.7.8, 5.6.7.9 lub 5.6.7.10
# iptables -t nat -A PREROUTING -i eth0 -j DNAT --to 5.6.7.8-5.6.7.10
```

```
## Zmień adresy docelowe ruchu WWW na 5.6.7.8, port 8080
# iptables -t nat -A PREROUTING -p tcp --dport 80 -i eth0 \
        -j DNAT --to 5.6.7.8:8080
```

Przekierowywanie (ang. *redirection*)

Jest to specjalizowany przypadek DNAT: dodano go dla wygodny, ponieważ jest to dokładnie to samo co wykonanie DNAT na adres interfejsu wejściowego.

```
## Wyślij przychodzący ruch WWW na port 80 do naszego transparentnego proxy (squida
# iptables -t nat -A PREROUTING -i eth1 -p tcp --dport 80 \
        -j REDIRECT --to-port 3128
```

Zwróć uwagę na to, że squid musi zostać skonfigurowany tak by wiedział że jest transparentnym proxy!

6.3 Więcej o mapowaniach

Jest trochę różnych subtelności w NAT z którymi większość użytkowników nigdy nie będzie miała do czynienia. Są one tu udokumentowane dla ciekawskich.

Wybór wielu adresów z zakresu

W momencie gdy poda się zakres adresów IP, są one wybierane na podstawie ostatnio używanego IP dla połączeń o których kod wie. Daje to coś na kształt prymitywnego rozkładania obciążenia.

Wykonywanie pustych mapowań NAT

Możesz użyć celu '`-j ACCEPT`' by pozwolić na przejście połączenia bez wykonania NAT.

Standardowe zachowanie NAT

Domyślnym zachowaniem jest zmiana połączenia tylko na tyle ile potrzeba, w granicach reguły podanej przez użytkownika. Oznacza to że nie będziemy remapować portów jeśli wprost tego nie zażądano, chyba że będzie to konieczne.

Mapowanie portów źródłowych wynikające z innych połączeń

Nawet gdy dla danego połączenia nie wykonuje się NAT, może się okazać że zostanie wykonana translacja adresu źródłowego, w związku z tym że jest już jakieś połączenie używające tego portu. Rozważmy następujący przykład maskarady, który jest dosyć powszechny:

1. Nawiązywane jest połączenie przez maszynę 192.1.1.1 z portu 1024 do maszyny www.netscape.com na port 80.
2. Połączenie jest maskaradowane przez maszynę prowadzącą NAT i używa jej adresu IP (1.2.3.4).
3. Maszyna ta stara się wykonać połączenie do www.netscape.com na port 80 z 1.2.3.4 (swojego adresu zewnętrznego) z portu 1024.
4. Kod NAT zmieni port źródłowy drugiego połączenia na 1025 tak by te dwa porty nie kolidowały ze sobą.

Kiedy odbywa się takie mapowanie, porty dzielone są na trzy klasy:

- porty poniżej 512
- porty pomiędzy 512 a 1023
- porty 1024 i wyżzsze

Port nie zostanie nigdy zmapowany w tym przypadku do innej klasy.

Co dzieje się gdy NAT zawiedzie

Jeśli nie istnieje sposób by unikalnie zmapować połączenie na żądanie użytkownika, zostanie ono odrzucone. Dotyczy to również pakietów nie sklasyfikowanych jako część istniejącego połączenia, ponieważ są zniekształcone, maszyna wyczerpała pamięć itd.

Wiele mapowań, nakładanie się i konflikty

Możesz stworzyć reguły NAT które mapują pakiety do tego samego zakresu; kod NAT jest na tyle mądry by uniknąć konfliktów. Oznacza to że dwie reguły mapujące adresy źródłowe 192.168.1.1 i 192.168.1.2 na 1.2.3.4 będzą działać poprawnie.

Co więcej, możesz mapować nawet na realne, używane adresy IP dopóki przechodzą one przez maszynę mapującą. Jeśli masz zatem przydzieloną sieć (1.2.3.0/24), ale masz jedną sieć wewnętrzną używającą tych adresów i jedną używającą zakresu adresów prywatnych 192.168.1.0/24, możesz po prostu wykonać NAT na adresy źródłowe 192.168.1.0/24 do sieci 1.2.3.0, bez strachu że coś rozsypie:

```
# iptables -t nat -A POSTROUTING -s 192.168.1.0/24 -o eth1 \
        -j SNAT --to 1.2.3.0/24
```

Ta sama logika dotyczy adresów używanych przez samą maszynę prowadzącą NAT: tak właśnie działa maskarada (współdzieląc adres interfejsu pomiędzy pakiety maskaradowane a 'prawdziwe' pakiety wychodzące z tej maszyny).

Co więcej, możesz mapować te same pakiety na wiele maszyn docelowych, i będą one współdzielone. Na przykład, jeśli nie chcesz mapować niczego na 1.2.3.5 możesz napisać:

```
# iptables -t nat -A POSTROUTING -s 192.168.1.0/24 -o eth1 \
        -j SNAT --to 1.2.3.0-1.2.3.4 --to 1.2.3.6-1.2.3.254
```

Zmiana adresu docelowego dla lokalnie generowanych połączeń

Kod wykonujący NAT pozwala na wstawianie reguł wykorzystujących cel DNAT do łańcucha OUTPUT, ale nie jest to w pełni obsługiwane w wersji 2.4 (może być, ale wymaga to nowych opcji konfigurujących, testowania i trochę kodu do dopisania, więc jeśli ktoś nie zleci tego Rusty'emu nie spodziewałbym się tego szybko).

Aktualne ograniczenie polega na tym, że możesz zmieniać tylko adres docelowy na maszynę lokalną (tzn. '-j DNAT --to 127.0.0.1'), a nie na dowolną inną, ponieważ odpowiedzi nie będą prawidłowo tłumaczone.

7. Protokoły specjalne

Niektóre protokoły nie lubią być poddawane NAT. Dla każdego z tych protokołów muszą zostać napisane dwa rozszerzenia: jedno dla śledzenia połączeń samego protokołu, drugie dla wykonania na nim NAT.

W dystrybucji netfilter znajdują się aktualnie moduły dla ftp: `ip_conntrack_ftp.o` i `ip_nat_ftp.o`. Możesz użyć `insmod` by dołączyć je do kernela (lub skompilować na stałe) i od tego momentu połączenia ftp przez NAT powinny działać poprawnie. Jeśli tego nie zrobisz, będziesz mógł używać tylko pasywnego ftp, a nawet wtedy mogą wystąpić problemy jeśli wykonujesz coś więcej niż tylko prosy SNAT.

8. Problemy z NAT

Jeśli wykonujesz NAT na konkretnym połączeniu, wszystkie pakiety podróżujące w **obu** kierunkach (do i z sieci) muszą przechodzić przez maszynę wykonującą NAT, w innym przypadku nie będzie to działało dobrze. W szczególności, kod śledzenia połączeń składaja fragmenty, co oznacza że nie tylko nie będzie on działał poprawnie, ale również i to, że twoje pakiety mogą w ogóle nigdzie nie dotrzeć, ponieważ jako fragmenty zostaną zatrzymane.

9. Źródłowy NAT (SNAT) i routing

Jeśli wykonujesz SNAT, musisz upewnić się że każda maszyna do której docierają pakiety poddane Źródłowemu NAT odpowie na nie kierując pakiety właśnie do maszyny wykonującej SNAT. Na przykład, jeśli mapujesz pakiety

wychodzące na adres źródłowy 1.2.3.4, zewnętrzny router musi wiedzieć że pakiety z odpowiedziami (które przeznaczone są dla **1.2.3.4**) muszą trafić z powrotem do tego komputera na którym odbywa się mapowanie. Można to wykonać na następujące sposoby:

1. Jeśli robisz SNAT na adres własny maszyny która go wykonuje (a routing dla niej już działa), nie musisz robić nic.
2. Jeśli robisz SNAT na nieużywany adres w lokalnej sieci LAN (na przykład, mapujesz na 1.2.3.99, wolny IP w twojejsieci 1.2.3.0/24), twój komputer wykonujący NAT będzie musiał odpowiadać na zapytania ARP do tego adresu, oprócz zapytań do jego własnego adresu: najprostszym sposobem by to osiągnąć jest stworzenie aliasu IP, np.:

```
# ip address add 1.2.3.99 dev eth0
```

3. Jeśli robisz SNAT na kompletnie inny adres, będziesz się musiał upewnić że maszyny do których będą docierały pakiety SNAT będą routować odpowiedzi z powrotem do maszyny wykonującej NAT. Tak się dzieje jeśli komputer wykonujący NAT jest domyślną bramką (gateway), w innym przypadku będzie trzeba rozgłosić taką trasę (jeśli wykorzystujesz jakiś protokół routowania) lub ręcznie dodać trasy do każdej maszyny której dotyczy ten przypadek.

10. Docelowy NAT (DNAT) do tej samej sieci

Jeśli wykonujesz przekazywanie portów (ang. *portforwarding*) do tej samej sieci, musisz upewnić się że zarówno przyszłe pakiety jak i odpowiedzi na nie przejdą przez maszynę prowadzącą NAT (by mogły być zmodyfikowane). Kod odpowiedzialny za NAT blokuje obecnie (od 2.4.0-test6) wychodzące pakiety przekierowujące ICMP (ang. *ICMP redirect*), które produkowane są w momencie gdy pakiety po przejściu przez NAT wychodzą tym samym interfejsem do którego dotarły, ale maszyna do której są skierowane nadal stara się odpowiedzieć bezpośrednio do klienta (który nie rozpozna odpowiedzi).

Klasycznym przykładem jest próba dostępu przez personel wewnętrzny do 'publicznego' serwera WWW, który tak naprawdę przesłonięty jest przez DNAT z publicznego adresu (1.2.3.4) na adres sieci wewnętrznej (192.168.1.1), tak jak poniżej:

```
# iptables -t nat -A PREROUTING -d 1.2.3.4 \
        -p tcp --dport 80 -j DNAT --to 192.168.1.1
```

Jednym ze sposobów jest utrzymywanie wewnętrznego serwera DNS który zna prawdziwe (wewnętrzne) adresy IP publicznego serwera WWW i przekazuje wszystkie inne wywołania do zewnętrznego serwera DNS. Oznacza to że logowanie dostępu na twoim serwerze WWW wskaże adresy IP z sieci wewnętrznej poprawnie.

Innym sposobem jest zapewnienie, by maszyna prowadząca NAT mapowała również źródłowy adres IP na jej własny, dla połączeń tego typu, ogłupiając serwer i sprawiając by odpowiadał przez nią. W tym przykładzie, wykonujemy co następuje (zakładając że wewnętrzny adres IP komputera prowadzącego NAT to 192.168.1.250):

```
# iptables -t nat -A POSTROUTING -d 192.168.1.1 -s 192.168.1.0/24 \
        -p tcp --dport 80 -j SNAT --to 192.168.1.250
```

Ponieważ reguła **PREROUTING** jest sprawdzana pierwsza, pakiety będą od razu skierowane do wewnętrznego serwera WWW: możemy stwierdzić które wywołania pochodzą z sieci wewnętrznej sprawdzając źródłowe adresy IP.

11. Podziękowania

Dziękuje przede wszystkim WatchGuard, oraz Davidowi Bonn, który wierzył na tyle w ideę netfilter by wsierać mnie kiedy nad nią pracowałem.

Oraz dla wszystkich, którzy podnieśli moje zorientowanie w temacie, gdy uczyłem się obrzydliwych stron NAT, szczególnie dla tych którzy czytali mój dziennik.

Rusty.

12. Uwagi od tłumacza

Chciałbym podziękować następującym osobom za uwagi co do tłumaczenia i zasugerowanie poprawek:

Piotr Mierzwiński administrator (at) kutno.mediacom.pl

- literówka