Sieciowe systemy operacyjne – laboratorium – Filtracja datagramów

Filtracja datagramów wykorzystuje mechanizm Netfilter dostępny w jądrze systemu operacyjnego. Umożliwia ona budowę zapór sieciowych, realizację translacji adresów NAT oraz swobodną inspekcję i dowolną modyfikację zawartości datagramów. Najpopularniejszym narzędziem do konfiguracji filtracji datagramów jest program iptables. iptables obecnie, wraz z towarzyszącymi ip6tables, arptables i ebtables , jest zastępowany nowym programem nft lepiej dopasowanym do mechanizmu Netfilter. Ćwiczenie zostanie zrealizowane z wykorzystaniem narzędzia nft.

Celem ćwiczenia jest zdobycie umiejętności konfiguracji filtracji datagramów i budowy zapory sieciowej. Ćwiczenie będzie realizowane w SO Linux w wersji Live DVD - Knoppix. Materiały dotyczące filtracji datagramów i jej konfiguracji były podane na wykładzie.

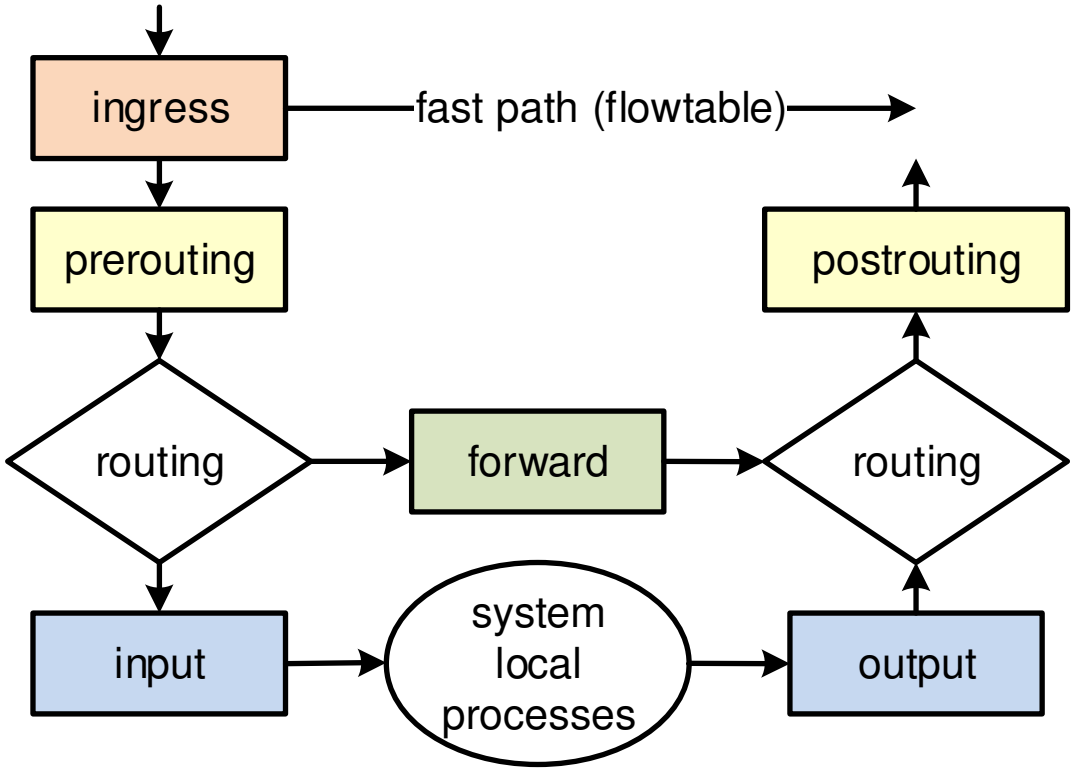
Do realizacji ćwiczenia przydatne jest zapoznanie się z informacjami umieszczonymi w Wiki nftables https://wiki.nftables.org/.

Ćwiczenie należy wykonać w dwuosobowych grupach. Numerem grupy jest parzysty numer komputera podzielony przez dwa.

eNauczanie: Do samodzielnej realizacji ćwiczenia potrzebne jest środowisko wirtualne z dwoma maszynami wirtualnymi. Jako numer grupy należy użyć ostatnią cyfrę numeru albumu.

1. Trasy datagramów

Mechanizm Netfilter zarządzany narzędziem nft nie definiuje żadnych standardowych tablic, filtrów ani łańcuchów filtracji. Umożliwia pełną kontrolę nad definicjami zasad inspekcji ruchu przy zachowaniu znanej od dwóch dekad architektury tras datagramów. Rysunek 1 przedstawia trasy datagramów w systemie operacyjnym z zaznaczeniem miejsc przechwytywania (ang. hook), w których datagramy mogą zostać poddane inspekcji.



Rysunek 1 Trasy datagramów w systemie operacyjnym i możliwe miejsca inspekcji.

Typowo można wyróżnić trzy drogi, którymi przemieszczają się datagramy:

1. Datagramy **docierające do procesów lokalnego systemu** operacyjnego wędrują trasą prowadzącą przez *ingres* → *prerouting* → *input*.

Hook *input* – jest tu głównym miejscem klasyfikacji i filtracji przychodzących datagramów.

Hook *prerouting* – jest wykorzystywany w mechanizmie translacji adresów docelowych DNAT umożliwiając zmianę adresu (albo portu) docelowego datagramu.

Hook *ingres* – jest nowym mechanizmem umożliwiającym inspekcję ramki warstwy drugiej zanim jeszcze trafi do stosu IP. Pozwala na filtrację z wykorzystaniem adresów MAC interfejsów sieciowych oraz realizację szybkiej ścieżki (ang. fast path) pozwalającej na skrócenie trasy datagramów należących do wcześniej zaakceptowanych strumieni danych.

2. Datagramy **wychodzące z procesów lokalnego systemu** operacyjnego wędrują trasą *output* → *postrouting*.

Hook *output* – jest głównym miejscem filtracji datagramów wychodzących z systemu.

Hook *postrouting* – jest wykorzystywany w translacji adresów źródłowych SNAT umożliwiając zmianę adresu źródłowego datagramu i np. współdzielenie jednego łącza internetowego przez wiele urządzeń w sieci lokalnej.

3. Datagramy **w systemie pełniącym funkcję routera** wędrują trasą *ingress* → *prerouting* → *forward* → *postrouting*.

Hook *forward* – jest głównym miejscem filtracji ruchu w routerze. Ze względu na to, że przez to miejsce przechodzą datagramy w obu kierunkach transmisji często zachodzi potrzeba umieszczania dwóch reguł do klasyfikacji jednego strumienia ruchu.

W celu przeprowadzenia klasyfikacji i ewentualnej filtracji ruchu wykorzystuje się tablice (ang. table) do realizacji celów np. filtracji, translacji adresów itp. oraz łańcuchy filtrów (ang. chain) do klasyfikacji i wykonywania akcji.

Najprostsze działanie zapory sieciowej polega na klasyfikacji docierających datagramów i podejmowaniu decyzji o ich przepuszczeniu (*accept*) albo odrzuceniu (*drop, reject*). Typowo zapora sieciowa odrzuca wszystkie datagramy przychodzące i przepuszcza wszystkie wychodzące z systemu. Polityka łańcucha jest decyzją dla tych datagramów, dla których nie pasuje żaden ze zdefiniowanych klasyfikatorów. Polityka jest wykonywana jako ostatnia. Polityką może być wyłącznie akcja *accept* albo *drop*.

Więcej szczegółów o klasyfikacji i możliwych akcjach można znaleźć w dokumentacji https://wiki.nftables.org/.

2. Zapora sieciowa dla stacji roboczej

Listing 1 prezentuje konfigurację typowej zapory sieciowej dla stacji roboczej w formacie pliku konfiguracyjnego polecenia nft. Zapora ta pracuje w trybie *stateful* i przepuszcza wyłącznie odpowiedzi na żądania wysłane z lokalnego systemu. Dodatkowo dopuszcza obsługę protokołu ICMPv6 w zakresie niezbędnym do działania klienta usługi Neighbor Discovery wymaganego do poprawnego działania w sieci IPv6 oraz dowolny ruch wewnątrz systemu przez interfejs lokalny lo.

```
#!/usr/sbin/nft -f

flush ruleset

table inet firewall {
    chain incoming {
        type filter hook input priority 0; policy drop;

        ct state established,related accept
        ct state invalid drop
        icmpv6 type { nd-neighbor-solicit, nd-router-advert } accept
        iif lo accept
    }
}
```

Listing 1 Konfiguracja prostej zapory sieciowej

Wyjaśnienie poszczególnych poleceń definicji zapory z listingu 1:

```
nft flush ruleset
```

Usuwa wszystkie reguły mechanizmu Netfilter.

```
nft add table inet firewall
```

Dodaje nową tablicę o nazwie *firewall*, która będzie zawierała reguły typu *inet*, czyli dotyczące zarówno IPv4 jak i IPv6.

```
nft add chain inet firewall incoming \
      { type filter hook input priority 0 \; policy drop \; }
```

Dodaje łańcuch o nazwie *incoming* do tablicy o nazwie *firewall* służący do filtracji datagramów typu *inet* (IPv4 i IPv6 równocześnie), których klasyfikacja będzie wykonywana w punkcie przechwytywania *input* ze standardowym priorytetem i polityką odrzucania przychodzących datagramów.

```
ntf add rule inet firewall incoming ct state established,related accept
ntf add rule inet firewall incoming ct state invalid drop
```

Dodaje reguły filtracji typu *inet* (IPv4 i IPv6 równocześnie) do łańcucha *incoming* w tablicy *firewall*, których klasyfikatorem jest badanie stanu połączenia (*ct state*) a działaniem przepuszczanie (*accept)*, gdy stan połączenia oznacza wcześniej nawiązane połączenie (*established*) i odrzucanie (*drop*) jeśli stan jest niewłaściwy albo nieokreślony (*invalid*).

```
ntf add rule inet firewall incoming \
      icmpv6 type { nd-neighbor-solicit, nd-router-advert } accept
```

Dodaje regułę filtracji do łańcucha *incoming* w tablicy *firewall* przepuszczającą (*accept*) datagramy protokołu ICMPv6 o typie ze zbioru *{nd-neighbor-solicit, nd-router-advert}* niezbędne do poprawnej pracy w sieci IPv6.

```
ntf add rule inet firewall incoming iif lo accept
```

Dodaje regułę przepuszczającą (*accept*) wszystkie datagramy docierające do interfejsu loopback *lo*.

Zadanie 2.1

- Wyłączyć interfejs eth0 tylko w komputerze K1.

```
sudo ip link set eth0 down
sudo ip addr flush dev eth0
```

- Na obu komputerach uruchomić usługi SSH i WWW.

```
sudo service ssh start
sudo service apache2 start
```

- Do interfejsów eth1 obu komputerów przypisać adresy zgodnie z planem 192.168.G.K/24, gdzie G jest numerem grupy, a K numerem komputera (patrz rysunek 2).

```
sudo ip addr add 192.168.G.K/24 dev eth1
```

- Sprawdzić poleceniem ping łączność między komputerami.

```
ping -c 4 192.168.G.K
```

- Uruchomić obsługę routingu tylko na komputerze K2.

```
sudo sysctl -w net.ipv4.ip_forward=1
sudo sysctl -w net.ipv6.conf.all.forwarding=1
```

- Dodać trasę domyślną przez router K2 tylko na hoście K1.

```
ip route add default via 192.168.G.K2
```

Zadanie 2.2

- Konfigurację zapory sieciowej z listingu 1 należy zapisać w pliku np. *fw.conf* a następnie uruchomić:

```
sudo nft -f fw.conf
```

Uwaga! Każda zmiana zawartości pliku konfiguracyjnego wymaga ponownego uruchomienia.

Zadanie 2.3 do samodzielnej realizacji

Uzupełnić konfigurację zapory sieciowej z listingu 1 o następujące funkcje:

2.3.1. Zliczanie ruchu (akcja *counter*).
   - Wszystkie przychodzące datagramy i bajty.
   - Wszystkie datagramy i bajty dla każdej z reguł.
   - Wszystkie odrzucone poza regułami datagramy i bajty.

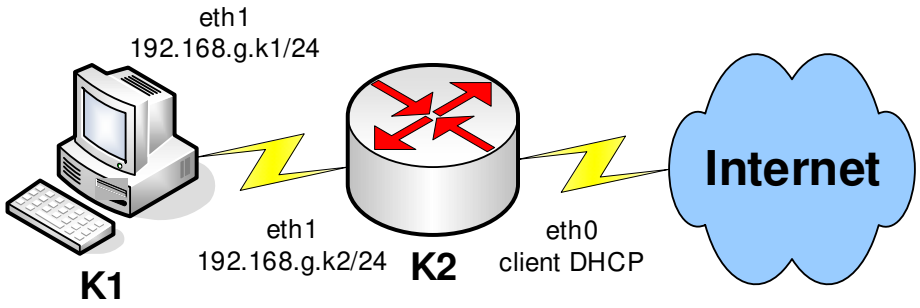
   Wykonać polecenie ping do sąsiada i sprawdzić działanie liczników.

```
ping -c4 192.168.G.K
sudo nft list table inet firewall
```

2.3.2. Zezwolenie na przychodzące datagramy polecenia ping (ICMP type echo-request) tylko z hostów sieci własnej grupy (192.168.G.0/24).
2.3.3. Zezwolenie na przychodzące nowe (ct state new) połączenia WWW (TCP port 80).
2.3.4. Zezwolenie na przychodzące nowe połączenia SSH tylko z sieci własnej grupy.
2.3.5. Dodanie do zapory w tablicy *firewall* łańcucha o nazwie *outgoing* z punktem przechwytywania *output* i polityką *accept*.
2.3.6. Zakaz korzystania z dowolnie wybranego serwisu WWW np. www.gdansk.pl z licznikiem prób połączenia.

3. Zapora dla routera z translacją adresów

W schemacie połączeń jak na rysunku 2 host K2 pełni funkcję routera z dostępem do Internetu. Mechanizm Netfilter może być wykorzystany do współdzielenia pojedynczego łącza i pojedynczego adresu IP routera przez wszystkie hosty w sieci lokalnej. Do realizacji współdzielenia łącza używana jest translacja adresów źródłowych SNAT (ang. Source Network Address Translation).



Rysunek 2 Konfiguracja połączeń i plan adresacji komputerów

```
table ip nat {
    chain shareinet {
        type nat hook postrouting priority 100; policy accept;

        oif eth0 masquerade
    }
}
```

Listing 2 Konfiguracja translacji adresów źródłowych SNAT w celu udostępnienia łącza internetowego

Zadanie 3.1

- Uzupełnić konfigurację zapory na hoście K2 o zawartość listingu 2 zapewniając współdzielenie łącza internetowego routera K2 dla hosta K1.
- Sprawdzić dostęp do Internetu z hosta K1.

```
ping -c4 8.8.8.8
```

Zadanie 3.2 do samodzielnej realizacji

Uzupełnić konfigurację zapory sieciowej o następujące funkcje:

3.2.1. Dodanie do zapory w tablicy *firewall* łańcucha o nazwie *forwarding* z punktem przechwytywania *forward* i polityką *drop*.

3.2.2. Zapewnienie nieograniczonego dostępu do Internetu dla hostów z sieci własnej grupy z realizacją dla całej sieci zapory w trybie *stateful* (*…ct state established…*). **Uwaga!** Nie wolno zmienić polityki *drop* łańcucha *forwarding*. Podpowiedź potrzebne są przynajmniej dwie reguły: zgoda na wysyłanie wszystkiego z sieci do Internetu i zgoda na powrót tylko nawiązanych połączeń z Internetu do sieci.

3.2.3. Ograniczenie dostępu do wybranego (innego niż w 2.3.6) serwisu WWW np. trojmiasto.pl dla wszystkich w sieci własnej grupy.

4. Zadania dodatkowe

Realizacja zadań dodatkowych wymaga dostępu do routera z zewnątrz i dostępu do sieci IPv6. Możliwe jest to w laboratoriach. Zadań dodatkowych nie trzeba realizować w trybie eNauczania.

Zadanie 4.1 dodatkowe

Uzupełnić konfigurację zapory sieciowej o następujące funkcje:

4.1.1. Dodanie przekierowania przychodzących połączeń TCP port 2022 routera do hosta K1 na port 22 (DNAT).

4.1.2. Dodanie przekierowania przychodzących połączeń TCP port 2080 routera do hosta K1 na port 80 (DNAT).

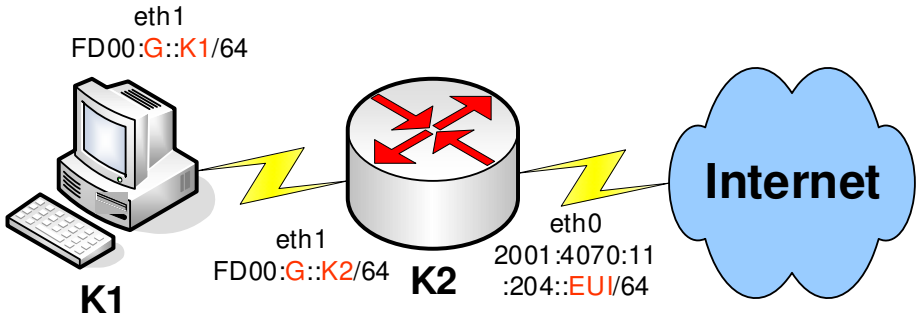
4.1.3. Wdrożenie mechanizmu *fast path*.

4.1.4. Wdrożenie adresacji IPv6 FD00:G::K/64 w sieci grupy oraz bramy domyślnej dla K1.

```
ip -6 addr add FD00:G::K/64 dev eth1
ip -6 route add 2000::/3 via FD00:G::K2 # tylko K1
```

4.1.5. Zapewnienie dostępu do Internetu IPv6 dla komputerów z sieci grupy z wykorzystaniem mechanizmu IPv6-to-IPv6 Network Prefix Translation.

4.1.6. Analogicznie do zadania 3.2.3 wprowadzić do łańcucha *forwarding* ograniczenie dla serwisu WWW dostępnego zarówno w sieci IPv4 jak i IPv6 np. facebook.com. Sprawdzić skuteczność ograniczenia.



Rysunek 3 Konfiguracja połączeń i plan adresacji dla sieci IPv6

eNauczanie: Jako sprawozdanie proszę wgrać wytworzony plik konfiguracyjne fw.conf do portalu https://enauczanie.pg.edu.pl