Filtrowanie ruchu sieciowego – mechanizm NetFilter

SNAT, DNAT, FILTROWANIE, PORT KNOCKING

Konrad Bryłowski, Aleksander Czerwionka, Michał Krause

31.05.2023

Spis treści

1. Informacje ogólne
2. Schemat sieci
3. SNAT
4. DNAT
5. Filtrowanie ruchu sieciowego
6. Port knocking

Informacje ogólne

ZASADA DZIAŁANIA, ZASTOSOWANIA

Informacje ogólne

Obecnie zalecanym systemem NetFilter jest *nftables*, który stopniowo zastępuje przestarzałe *iptables*, *ip6tables*, itp.

Rozwiązanie to jest częścią jądra Linux od 2014 roku.

Od *iptables* różni się większą uniwersalnością i możliwością dostosowania.

Informacje ogólne

W odróżnieniu od *iptables* administrator sam tworzy tablice i łańcuchy dopasowując je do chronionej sieci/urządzenia.

Główne elementy:

- Zbiór wszystkich reguł (*ruleset*)
  - Tablice (*tables*)
    - Łańcuchy (*chains*)
      - Reguły (*rules)*
    - Zbiory (*sets*)
  - Zmienne

Informacje ogólne

Składnia poleceń *nftables* – wszystkie wymagają uprawnień root:

```
nft {list | flush} {ruleset | table <nazwa> | sets}
nft {add | list | delete | flush} {table | chain} <nazwa>
nft {add | insert} rule <nazwa tablicy i łańcucha> <nazwa> …
nft {add | delete | list | flush } set <tablica> nazwa
nft {add | delete} element <tablica> <zbiór> { element [, …]}
```

Informacje ogólne

Jest też możliwość importowania zbiorów reguł z pliku / plików. Polecenie:

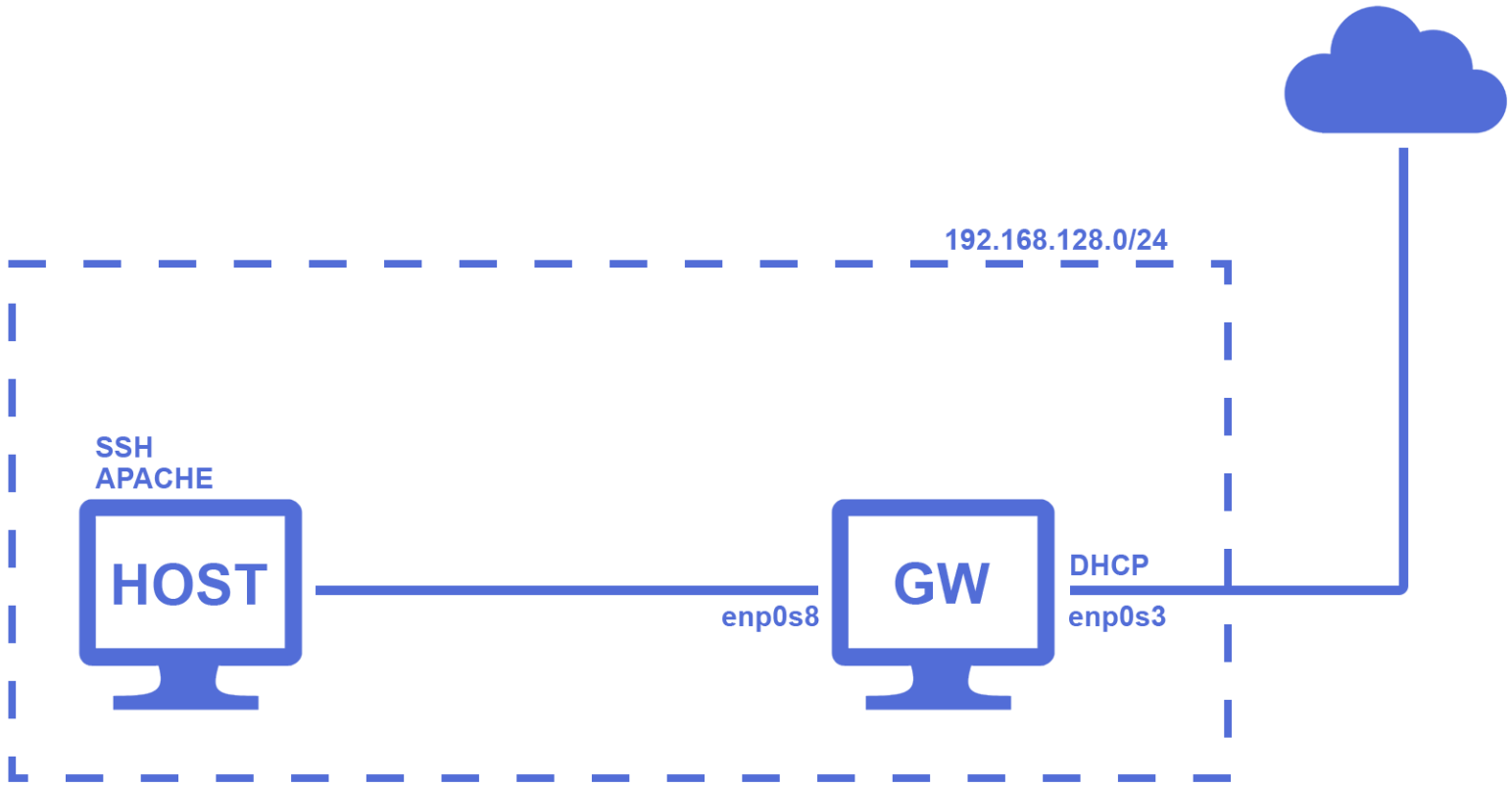
```
sudo nft -f plik
```

Polecenie to dodaje zawartość pliku do obecnego zbioru reguł (chyba, że w pliku jest polecenie `flush ruleset`).

Składnia pliku może być w formie pojedynczych poleceń albo w formie takiej jaką zwraca polecenie `list ruleset`.

Schemat sieci

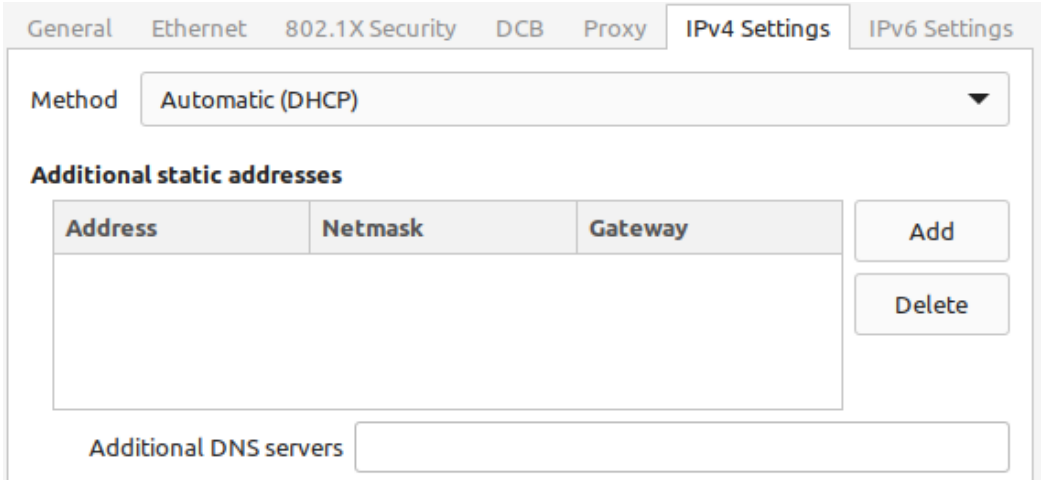
STOSOWANE OZNACZENIA I ADRESACJE



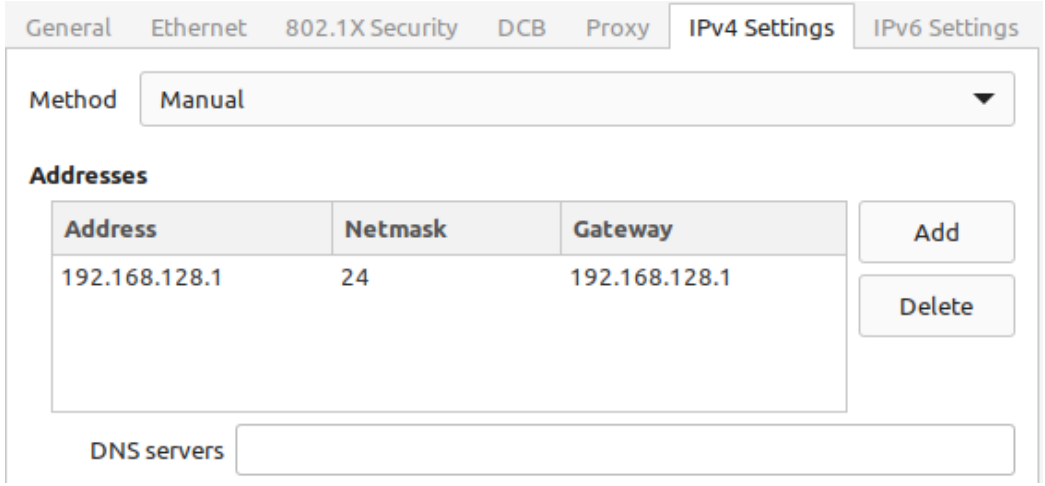
Schemat sieci

Schemat sieci

USTAWIENIA SIECIOWE NA GW – ENP0S3

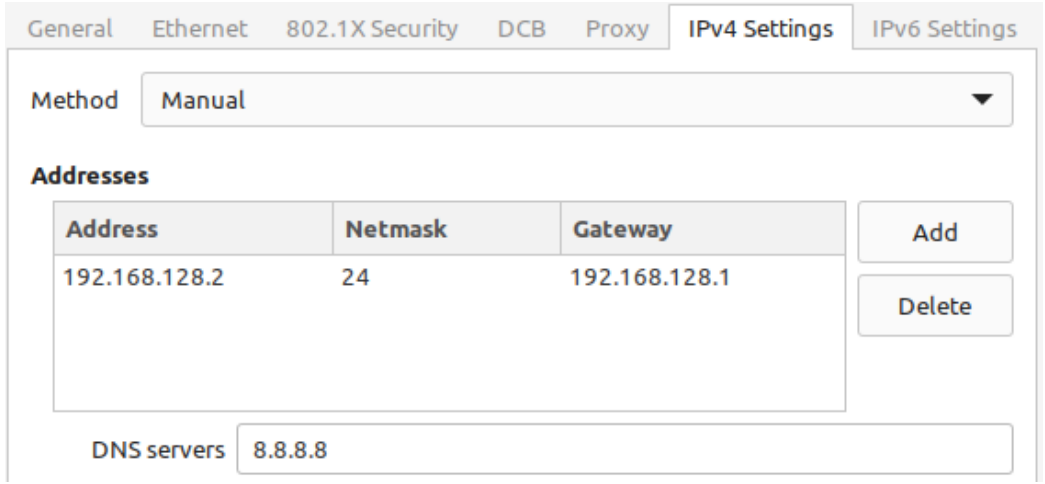


USTAWIENIA SIECIOWE NA GW – ENP0S8



Schemat sieci

USTAWIENIA SIECIOWE NA HOST



SNAT

SOURCE NETWORK ADDRESS TRANSLATION

SNAT

Zmiana adresu źródłowego pakietu na inny.

Najczęstsze zastosowanie – dostęp do Internetu z sieci o adresacji prywatnej.

Szczególny przypadek – urządzenie dokonujące zamiany ma zmienny adres – maskarada (masquerade), czyli zmiana nie na konkretny adres, a na adres interfejsu, przez który pakiet opuszcza urządzenie.

Pakiet pochodzący z zewnątrz będzie przekazany do hosta wewnątrz chronionej sieci tylko jeśli jest odpowiedzią na żądanie z wewnątrz.

SNAT

Łańcuch odpowiedzialny za translację adresów z sieci prywatnej na adres maszyny GW w sieci niechronionej.

```
chain postrouting {
        type nat hook postrouting priority srcnat; policy accept;
        ip saddr 192.168.128.0/24 oifname "enp0s3" masquerade
}
```

DNAT

DESTINATION NETWORK ADDRESS TRANSLATION

DNAT

Zmiana adresu docelowego pakietu na inny, należący do chronionej sieci.

Najczęstsze zastosowanie – serwer, który ma tylko adres prywatny ma udostępniać usługi sieciowe w Internecie. W takich przypadkach pakiety kierowane na określony port są kierowane do innego urządzenia.

Ograniczenie – do każdego portu można przypisać tylko jeden adres docelowy, np. nie jest możliwe udostępnienie na porcie 80 dwóch serwerów WWW działających na różnych maszynach (pod różnymi adresami w sieci prywatnej).

DNAT

Łańcuch odpowiedzialny za przekazywanie pakietów przychodzących na porty 80 i 443 maszyny GW do maszyny HOST, na której działa serwer WWW.

```
chain prerouting {
        type nat hook prerouting priority dstnat; policy accept;
        iif "enp0s3" tcp dport { 80, 443 } dnat to 192.168.128.2
}
```

Filtrowanie ruchu sieciowego

ODFILTROWANIE RUCHU POD KONKRETNY ADRES, DOPUSZCZANIE KONKRETNYCH USŁUG SIECIOWYCH

Filtrowanie ruchu sieciowego

Realizowane przez badanie reguł kolejnych łańcuchów typu *filter* w rosnącej kolejności wartości *priority*.

Reguły mogą decydować o zaakceptowaniu pakietu (*accept*), odrzuceniu (*reject*), porzuceniu (*drop*), przejściu do innego łańcucha (*jump* albo *goto*) albo kontynuowaniu oceny *(continue*).

Możliwość sprawdzania nadawcy, adresata, portu źródłowego, portu docelowego, protokołu i dopasowywania ich do zdefiniowanych przez administratora zbiorów, list i wartości.

Filtrowanie ruchu sieciowego

Możliwość zapisywania do logów (*log)* – domyślnie /var/log/syslog na Linux Mint

Możliwość zliczania dopasowań do danej reguły (*counter)*

Filtrowanie ruchu sieciowego

ŁAŃCUCH BLOKUJĄCY RUCH PRZYCHODZĄCY (Z WYJĄTKAMI)

```
chain input {
        type filter hook input priority filter; policy drop;
        iifname "lo" accept
        ct state invalid drop
        ct state established,related accept
        meta l4proto icmp accept
        jump ports
        jump knock_chain
}
```

ŁAŃCUCH ZEZWALAJĄCY NA RUCH PAKIETÓW PROTOKOŁÓW DNS, HTTP I HTTPS

```
chain ports {
        tcp dport { 53, 80, 443 } accept
        tcp sport { 53, 80, 443 } accept
        udp sport { 53, 80, 443 } accept
        udp dport { 53, 80, 443 } accept
}
```

Filtrowanie ruchu sieciowego

ŁAŃCUCH BLOKUJĄCY RUCH WYCHODZĄCY POD OKREŚLONY ADRES

```
chain banned_addr {
        ip daddr 212.77.98.9 counter packets 0 bytes 0 reject with icmp port-unreachable
        continue
}
```

PRÓBA POŁĄCZENIA Z ZABLOKOWANYM ADRESEM

```
user@host:~$ ping wp.pl
PING wp.pl (212.77.98.9) 56(84) bytes of data.
From _gateway (192.168.128.1) icmp_seq=1 Destination Port Unreachable
```

Port knocking

PORT ZAMKNIĘTY, ALE WYSTARCZY ODPOWIEDNIO ZAPUKAĆ

Port knocking

Możliwość połączenia się z chronionym portem znając odpowiednią sekwencję portów.

Realizowane w łańcuchu typu *filter*.

Nadawca pakietu skierowanego na pierwszy port w sekwencji jest zapamiętywany na określony czas, w którym musi wykonać połączenie do kolejnego portu z sekwencji.

Przy ostatnim elemencie sekwencji nadawca jest zapamiętywany na dłużej i dopuszczany do nawiązania nowego połączenia z chronionym portem.

Najczęstsze zastosowanie – zdalny dostęp do urządzenia.

Port knocking

REGUŁY ODPOWIEDZIALNE ZA KNOCKING

```
tcp dport 123 add @candidates { ip saddr . 234 timeout 5s }
tcp dport 234 ip saddr . tcp dport @candidates add @candidates { ip saddr . 345 timeout 5s }
tcp dport 345 ip saddr . tcp dport @candidates add @clients { ip saddr timeout 20s }
tcp dport 22 counter ip saddr @clients accept
tcp dport 22 ct state established,related accept
tcp dport 22 counter reject with tcp reset
```

UDANA SEKWENCJA

```
mint@mint:~$ knock -v 192.168.43.10 123 234 345 -d 500
hitting tcp 192.168.43.10:123
hitting tcp 192.168.43.10:234
hitting tcp 192.168.43.10:345
mint@mint:~$ ssh user@192.168.43.10
The authenticity of host '192.168.43.10 (192.168.43.10)' can't be established.
ED25519 key fingerprint is SHA256:KrTSYYIIJRelr/6xDwx/XU+FFq7hljOStS+h+Vb7Gig.
This host key is known by the following other names/addresses:
    ~/.ssh/known_hosts:1: [hashed name]
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.43.10' (ED25519) to the list of known hosts.
user@192.168.43.10's password:
Last login: Tue May 30 23:51:19 2023 from 192.168.0.9
user@gw:~$
```

Dziękujemy za uwagę

KONRAD BRYŁOWSKI 188577
ALEKSANDER CZERWIONKA 188659
MICHAŁ KRAUSE 188592