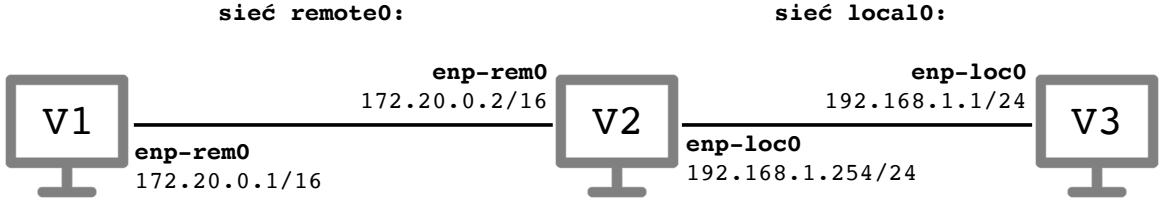
# Warsztaty z Sieci komputerowych
# Lista 8

## Konfiguracja początkowa

Przygotuj sieć jak na rysunku poniżej wykonując poniższe polecenia. Warto myśleć, że maszyna *Virbian3* jest lokalnym komputerem, maszyna *Virbian2* jest routerem, zaś maszyna *Virbian1* jest serwerem w Internecie, którego konfiguracji nie możemy bezpośrednio zmieniać.




- Uruchom trzy maszyny wirtualne *Virbian1* – *Virbian3* połączone za pomocą sieci wirtualnych `local0` i `remote0`. Odpowiednie pary interfejsów powinny mieć nazwy `enp-loc0` i `enp-rem0` jak na rysunku powyżej. Aktywuj wszystkie interfejsy i uruchom Wiresharka na wszystkich maszynach wirtualnych.
- Przypisz interfejsom adresy IP jak na powyższym rysunku. Sprawdź, że z maszyny *Virbian2* możesz z powodzeniem pingnąć obie sąsiednie maszyny.
- Na maszynie *Virbian3* ustaw bramę domyślną na adres interfejsu `enp-loc0` maszyny *Virbian2*. Sprawdź, że z maszyny *Virbian3* możesz z powodzeniem pingnąć oba adresy IP maszyny *Virbian2*.
- Sprawdź, co dzieje się, jeśli z maszyny *Virbian3* pingasz maszynę *Virbian1*. Za pomocą Wiresharka sprawdź, że komunikaty *ICMP echo request* dochodzą do celu, ale odpowiedzi nie wracają do nadawcy. Dlaczego tak się dzieje?

## Tutorial #1

- Na maszynie *Virbian3* poleceniem

  ```
  V3$> telnet 192.168.1.254 7
  ```

  połącz się z serwerem echa maszyny *Virbian2*. Obejrzyj przesyłane pakiety w Wiresharku. Obejrzyj całą komunikację klikając prawym przyciskiem myszy jeden z pakietów należących do połączenia `telnet` i następnie wybierając z menu kontekstowego Wiresharka opcję *Follow* | *TCP stream*.

  Program `telnet` możesz zakończyć naciskając kombinację `Ctrl` + `]` i następnie wpisując `quit`.

- Na maszynie *Virbian2* włącz serwer SSH poleceniem

  ```
  V2#> systemctl start ssh
  ```

  a następnie połącz się z maszyny *Virbian3* z tym serwerem poleceniem

  ```
  V3$> ssh 192.168.1.254
  ```

  podając `user` jako hasło użytkownika `user`. Z jakim portem zostało nawiązane połączenie? Zauważ, że podczas pracy na zdalnej maszynie znak zachęty zawiera czerwony napis `[REMOTE]`.

- Będąc zalogowanym(-ą) na maszynie *Virbian2* przez SSH wykonaj jakieś polecenie, np. wyświetl zawartość katalogu domowego poleceniem `ls`. Obejrzyj całą komunikację za pomocą opcji *Follow | TCP stream* Wiresharka. Czy potrafisz odczytać przesyłane dane? Zamknij połączenie SSH.

- Skonfigurujemy teraz SSH, tak aby możliwe było łączenie się z maszyny *Virbian3* do maszyny *Virbian2* bez podawania hasła. Wygeneruj klucz publiczny i prywatny poleceniem

  ```
  V3$> ssh-keygen
  ```

  Zapisz te klucze w domyślnych plikach (odpowiednio `.ssh/id_rsa.pub` oraz `.ssh/id_rsa`). Hasło zabezpieczające klucz pozostaw puste. (Zazwyczaj pozostawianie klucza prywatnego niezabezpieczonego hasłem to zły pomysł). Obejrzyj właśnie wygenerowane pliki z kluczami.

- Teraz wystarczy dopisać wygenerowany klucz publiczny do pliku `.ssh/authorized_keys` na serwerze SSH (maszynie *Virbian2*). W tym celu skopiuj go poleceniem

  ```
  V3$> scp .ssh/id_rsa.pub 192.168.1.254:keyfile
  ```

  Na maszynie *Virbian2* dopisz skopiowany właśnie klucz publiczny do pliku `.ssh/authorized_keys` poleceniami

  ```
  V2$> mkdir -p .ssh
  V2$> cat keyfile >> .ssh/authorized_keys
  V2$> rm keyfile
  ```

- Sprawdź, czy działania odniosły skutek, tj. czy możesz zalogować się teraz z maszyny *Virbian3* na maszynę *Virbian2* bez podawania hasła. Polecenie

  ```
  V3$> ssh -v 192.168.1.254
  ```

  wyświetli kolejne etapy nawiązywania połączenia. Obejrzyj je również w Wiresharku. Na końcu zamknij sesję SSH.

- Prostym sposobem zaszyfrowania połączenia jest wykorzystanie tunelowania strumienia danych w danych protokołu SSH. Na maszynie *Virbian3* utworzymy tunel SSH łączący port `7777` lokalnej maszyny (*Virbian3*) z portem 7 maszyny *Virbian2*. W tym celu wykonaj polecenie

```
V3$> ssh -4 -N -L 7777:localhost:7 user@192.168.1.254
```

i pozostaw je uruchomione. Sprawdź, że po wpisaniu na maszynie *Virbian3* polecenia

```
V2$> telnet localhost 7777
```

odpowiada serwer echa maszyny *Virbian2*.

- Na podstawie Wiresharka i wykorzystując polecenie `sudo netstat -tanp4` sprawdź, że strumień danych od programu `telnet` na maszynie *Virbian3* do serwera echa na maszynie *Virbian2* jest dzielony na trzy etapy:
  - Połączenie między portem (jakim?) programu `telnet` na maszynie *Virbian3* z portem `7777` klienta ssh na maszynie *Virbian3*.
  - Połączenie między klientem ssh na maszynie *Virbian3* a serwerem ssh na maszynie *Virbian2*.
  - Połączenie między portem (jakim?) serwera ssh na maszynie *Virbian2* a portem `7` serwera echa na maszynie *Virbian2*.

  W Wiresharku sprawdź, że drugie z tych połączeń jest szyfrowane, zaś pierwsze i trzecie nie są.
- Zamknij program `telnet` i sesję SSH tunelującą połączenie.

# Tutorial #2

W tej części skonfigurujemy zaporę na maszynie *Virbian2*, wykorzystując moduł `nftables` jądra konfigurowany przez polecenie `nft`. Zaporę można konfigurować interaktywnie za pomocą tego programu, lecz wygodniej jest edytować plik konfiguracyjny `/etc/nftables.conf`.

- Upewnij się, że zawartość pliku `/etc/nftables.conf` na maszynie *Virbian2* jest (z dokładnością do białych znaków) taka, jak poniżej.[1]

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

flush ruleset

table inet my_table {
  chain my_input_rules {
    type filter hook input priority filter;
  }
  chain my_forward_rules {
    type filter hook forward priority filter;
  }
  chain my_output_rules {
    type filter hook output priority filter;
```

[1] Obecne wersje `nftables` definiują nazwy priorytetów takie jak `filter = 0` czy `srcnat = 100`, które warto wykorzystywać przy definiowaniu reguł filtrujących lub definiujących źródłowy NAT. Więcej informacji: https://wiki.nftables.org/wiki-nftables/index.php/Netfilter_hooks#Priority_within_hook.

```
  }
}
```

- Wykonaj teraz polecenie

  ```
  V2#> /etc/nftables.conf
  ```

  Powyższe polecenie należy wykonywać na maszynie *Virbian2 po każdej* edycji pliku `/etc/nftables.conf` (nie będzie to zaznaczone poniżej). Spowoduje to skonfigurowanie zapory zgodnie z instrukcjami z tego pliku.

  Obecnie powyższe instrukcje usuwają wszystkie istniejące reguły a następnie implementują (pustą) konfigurację z sekcji `table inet my_table {...}`. Aktualną konfigurację zapory możesz wyświetlić poleceniem

  ```
  V2#> nft list ruleset
  ```

- Zmodyfikujemy teraz plik `/etc/nftables.conf` na maszynie *Virbian2* tak, żeby pakiety przychodzące do maszyny *Virbian2* i przechodzące przez nią były odrzucane, zaś pakiety wychodzące przepuszczane. Dodatkowo będziemy rejestrować wszystkie odrzucone w ten sposób pakiety do pliku dziennika. W tym celu sekcja `table inet my_table {...}` powinna wyglądać następująco:

  ```
  chain my_input_rules {
    type filter hook input priority filter;
    log
    drop
  }
  chain my_forward_rules {
    type filter hook forward priority filter;
    log
    drop
  }
  chain my_output_rules {
    type filter hook forward priority filter;
    accept
  }
  ```

- Wyświetl bieżące ustawienia zapory. Wyświetlone reguły powinny odpowiadać zapisanym w pliku `/etc/nftables.conf`.

- W osobnym terminalu uruchom polecenie

  ```
  V2#> journalctl -f -k
  ```

  wyświetlające bieżące komunikaty jądra pliku dziennika maszyny *Virbian2* (czyli w szczególności odrzucone pakiety).

- Sprawdź, co zapisuje się do pliku dziennika i co wyświetlane jest w Wiresharku, gdy pingasz z maszyny *Virbian3* maszynę *Virbian2*, a co, gdy z maszyny *Virbian2* pingasz maszynę *Virbian3*. W którym przypadku zatrzymywane są komunikaty *ICMP echo request*, a w którym komunikaty *ICMP echo reply*?

- Zaktualizuj konfigurację zapory dopisując do sekcji `chain my_input_rules {...}` regułę wpuszczającą pakiety należące do już nawiązanych połączeń. Odpowiednia część pliku konfiguracyjnego powinna teraz wyglądać następująco:

  ```
  chain my_input_rules {
    type filter hook input priority filter;
    ct state established,related accept
    log
    drop
  }
  ```

  Zwróć uwagę na kolejność reguł: zaakceptowane pakiety nie będą rejestrowane. Zaobserwuj, że teraz pinganie maszyny *Virbian3* z maszyny *Virbian2* będzie już działać.

- Sprawdź, że nadal nie jest możliwe pingnięcie maszyny *Virbian2* z maszyny *Virbian3*. Co więcej, nie jest nawet możliwe pingnięcie maszyny *Virbian2* z niej samej:

  ```
  V2$> ping 127.0.0.1
  ```

  ani też połączenie z portem echa lokalnej maszyny:

  ```
  V2$> telnet 127.0.0.1 7
  ```

  Aby to naprawić, wpuść wszystkie połączenia lokalne oraz połączenia ICMP *echo request* z zewnątrz, zmieniając sekcję `chain my_input_rules {...}` na następującą:

  ```
  chain my_input_rules {
    type filter hook input priority filter;
    ct state established,related accept
    iif lo accept
    ip protocol icmp icmp type echo-request accept
    log
    drop
  }
  ```

  W razie potrzeby składnię poleceń możesz sprawdzić w manualu, albo przeczytać jeden z dostępnych tutoriali dla `nftables`, np. https://wiki.archlinux.org/index.php/Nftables.

  Sprawdź, że pinganie maszyny *Virbian2* z maszyny *Virbian3* jest obecnie możliwe. Sprawdź też, że możliwe stało się połączenie z lokalnym portem echa na maszynie *Virbian2*.

- Za pomocą polecenia

  ```
  V3#> nmap -A -T4 192.168.1.254
  ```

sprawdź, jakie porty są dostępne do komunikacji na maszynie *Virbian2*. Przeczytaj uważnie wyświetlane informacje. Oglądając pakiety w Wiresharku i komunikaty o zablokowanych pakietach w pliku dziennika sprawdź, z jakimi portami usiłował połączyć się `nmap`.

Uwaga: powyższe polecenie będzie miało inny efekt, jeśli zostanie wywołane z uprawnieniami zwykłego użytkownika.

- Włącz teraz w zaporze możliwość łączenia się z portem SSH zmieniając sekcję `chain my_input_rules {...}` na następującą:

```
chain my_input_rules {
  type filter hook input priority filter;
  ct state established,related accept
  iif lo accept
  ip protocol icmp icmp type echo-request accept
  ct state new tcp dport 22 accept
  log
  drop
}
```

Sprawdź, że połączenie SSH z maszyny *Virbian3* do maszyny *Virbian2* jest teraz możliwe. Ponownie wykonaj skan portów poleceniem `nmap` i porównaj wyniki.

# Wyzwanie #1

W tym zadaniu skonfigurujemy mechanizm NAT na maszynie *Virbian2*, żeby umożliwić maszynie *Virbian3* komunikację z maszyną *Virbian1* (i innymi potencjalnymi maszynami osiągalnymi z *Virbian2* za pomocą interfejsu `enp-rem0`).

- Z maszyny *Virbian3* pingnij maszynę *Virbian1*. Na początku zajęć to polecenie powodowało dojście pakietów *ICMP echo* do maszyny *Virbian1*, lecz teraz te pakiety te dochodzą tylko do maszyny *Virbian2* (sprawdź to w Wiresharku).

  Sprawdź, co pojawia się w pliku dziennika maszyny *Virbian2*. Okazuje, się że winne jest blokowanie ruchu przechodzącego przez tę maszynę. Napraw to zmieniając sekcję `chain my_forward_rules {...}` zapory na następującą:

```
chain my_forward_rules {
  type filter hook forward priority filter;
  ct state established,related accept
  iif enp-loc0 oif enp-rem0 ct state new accept
  log
  drop
}
```

Zmiany powodują przepuszczanie całego ruchu pochodzącego od maszyny *Virbian3* do maszyny *Virbian1* i przepuszczanie pakietów należących do już nawiązanych połączeń w drugą stronę. Sprawdź, że jeśli pingasz maszynę *Virbian1* z maszyny *Virbian3*, to pakiety *ICMP echo request* już docierają, ale wciąż nie wracają odpowiedzi *ICMP echo reply* (dlaczego?).

- Moglibyśmy naprawić sytuację definiując odpowiednio routing na maszynie *Virbian1*. Ale w tym zadaniu będziemy zakładać, że nie mamy takiej możliwości i poradzimy sobie włączając funkcję źródłowego NAT na maszynie *Virbian2*. W tym celu na końcu pliku `/etc/nftables.conf` dopisz następujące wiersze

  ```
  table inet my_nat_table {
    chain my_source_nat_rules {
      type nat hook postrouting priority srcnat;
      ip saddr 192.168.1.0/24 oif enp-rem0 snat 172.20.0.2
    }
  }
  ```

  Spowodują one, że pakiety z oryginalnym adresem źródłowym pochodzącym z sieci `192.168.1.0/24` i wychodzące przez interfejs `enp-rem0` będą otrzymywały adres IP tego interfejsu.

- Pingnij maszynę *Virbian1* z maszyny *Virbian2* i z maszyny *Virbian3*. Porównaj w Wiresharku na maszynie *Virbian1* komunikaty ICMP *echo-request* dochodzące do maszyny *Virbian1* w obu powyższych przypadkach. W czym są podobne, a co je odróżnia? (Zwróć uwagę na źródłowe i docelowe adresy IP oraz na pole TTL).

  Obejrzyj też te komunikaty w Wiresharku uruchomionym na maszynie *Virbian2*: wszystkie pakiety przechodzące będą rejestrowane dwukrotnie, tj. przed podmianą źródłowego adresu IP i po niej.

- Uruchom na maszynie *Virbian1* usługę `ssh` i sprawdź, czy możesz się z nią połączyć z maszyny *Virbian3*. Sprawdź, jaki jest lokalny port tego połączenia na maszynie *Virbian3* i jaki jest port przypisany przez NAT na maszynie *Virbian2*.

- Zdekonfiguruj interfejsy sieciowe i wyłącz maszyny wirtualne.

Materiały do kursu znajdują się w systemie SKOS: https://skos.ii.uni.wroc.pl/.

*Marcin Bieńkowski*