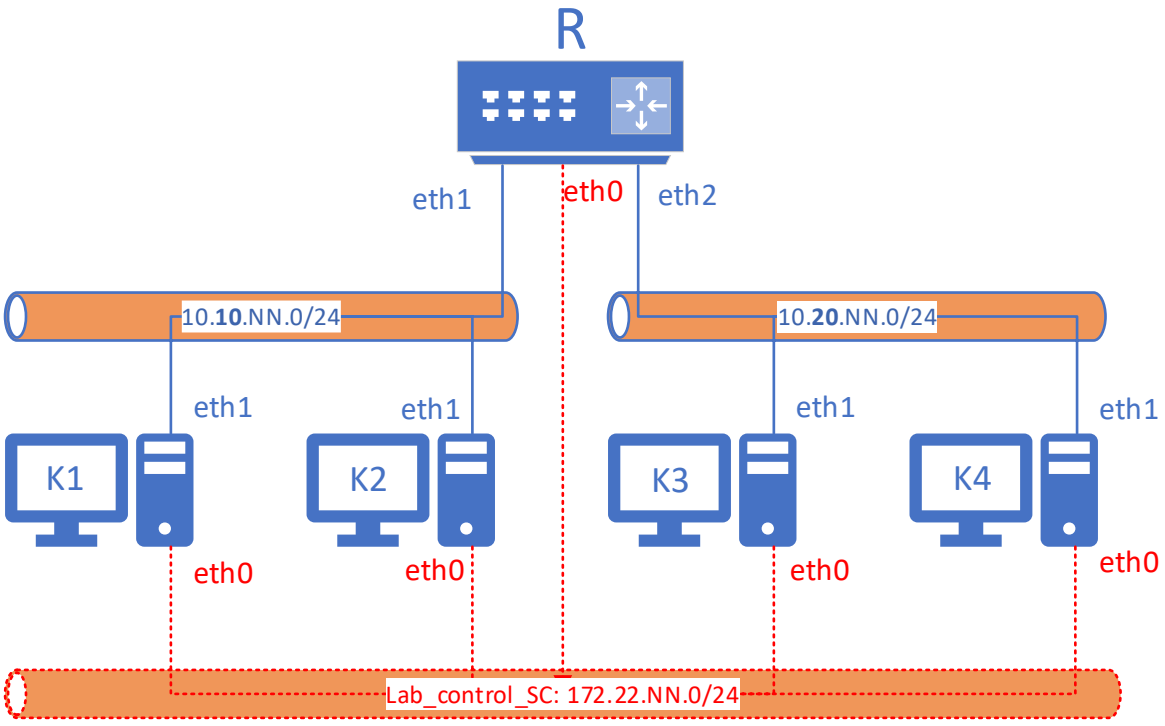
1 Cel ćwiczenia

Celem ćwiczenia jest zapoznanie studentów z podstawami adresacji oraz działania IP.

2 Środowisko laboratoryjne

Dla każdej grupy przygotowano środowisko laboratoryjne, które składa się z 5 kontenerów Docker: K1, K2, K3, K4, R, tak jak przedstawiono na rysunku 1.



K1 oraz K2 to dwa „Komputery" wpięte do sieci lokalnej o adresie 10.10.NN.0/24.
K3 oraz K4 to dwa „Komputery" wpięte do sieci lokalnej o adresie 10.20.NN.0/24.
„Ruter" R jest wpięty do obu tych sieci.
Wszystkie kontenery korzystają z systemu Ubuntu Server 20.04 i posiadają doinstalowane dodatkowe paczki programów wykorzystywane w trakcie ćwiczenia.

Wartość NN to numer grupy zadany przez prowadzącego. Oznaczenia grup są dwucyfrowe co ma znaczenie przy wyznaczeniu używanego portu do Logowania (patrz kolejny rozdział). W adresie IP postać 0N zostanie zmieniona na N (np. 10.10.02.0/24 -> 10.10.2.0/24).

Interfejs eth0 służy do podłączenia z siecią zarządzania (dostęp przez SSH / Internet), natomiast interfejsy eth1 i eth2 do przekazu danych między urządzeniami laboratorium. Interfejs eth0 pozwala na dowolne manipulowanie interfejsami eth1 i eth2 bez ryzyka zerwania połączenia z kontenerem. **Interfejs eth0 nie powinien być w żaden sposób rekonfigurowany** (~~`ifconfig eth0 <adres_ip>`~~) w trakcie ćwiczenia

Adresy na interfejsach eth1 i eth2 zostały nadane automatycznie przez Docker. **Należy zwrócić uwagę, że pierwszy adres z każdej sieci jest używany przez serwer/host.** Określenie adresów urządzeń jest częścią ćwiczenia.

2.1 Logowanie

Serwer SSH na kontenerach został wystawiony na konkretny port serwera/hosta o adresie **10.140.0.129** zgodnie z zasadą:

- K1 – 2NN11
- K2 – 2NN12
- K3 – 2NN13
- K4 – 2NN14
- R – 2NN15

Serwer znajduje się w sieci PL-LAB i powinien być osiągalny po połączeniu do VPN PL-LAB.

Logując się przez SSH należy użyć **użytkownika „root" i hasła „rootNN".**

Przykładowo, grupa NN=88 łączy się na Komputer K1 poleceniem:

ssh root@10.140.0.129 -p 28811

a na Ruter R poleceniem:

ssh root@10.140.0.129 -p 28815

UWAGA: Komputer za pomocą którego wykonywane jest ćwiczenie, będzie w tej instrukcji nazywany jako: **Komputer Studenta**

2.2 Przydatne komendy

2.2.1 Podstawowe komendy

`ps -au`	wypisanie listy działających procesów
`history`	wypisuje listę wszystkich wpisanych komend
`<tab>`	Wciśnięcie klawisza Tabulacji powoduje uzupełnienie wpisywanego tekstu (np. komendy, folderu, czy nazwy pliku)
`<Ctrl+c>`	Wciśnięcie kombinacji `<Ctrl+c>` powoduje zatrzymanie działającego procesu
`<strzałki>`	Strzałki w górę i w dół przewijają ostatnio używane komendy.

Komendy niepotrzebne do wykonania ćwiczenia (ale tak podstawowe, że musiały się tu znaleźć):	
`ls`	wypisanie zawartości katalogu.
`ls -lh`	wypisanie zawartości katalogu w postaci listy wraz z właściwościami plików
`pwd`	Wypisanie ścieżki bezwzględnej katalogu, w którym aktualnie się znajdujemy
`cat <plik>`	wypisanie zawartości pliku <plik>
`cp <plik1> <plik2>`	Skopiowanie <plik1> do <plik2>
`cd <path>`	przejście do katalogu opisanego w ścieżce <path>, ścieżka może być bezwzględna (rozpoczęta znakiem „/”) lub względna. • ścieżka bezwzględna to cała ścieżka, od poziomu „/”, np. komenda `cd /root/my_data` spowoduje przejście do katalogu /root/my_data, niezależnie od tego, w jakim katalogu komenda została wykonana. • Ścieżka względna to ścieżka wyznaczona od poziomu katalogu, w którym aktualnie się znajdujemy, np. `cd my_data` wywołane w katalogu /root/ spowoduje przejście do katalogu /root/my_data, ale wywołana w innym katalogu zwróci błąd „No such file or directory”. `cd ..` - przejście poziom wyżej w strukturze katalogów

2.2.2 ping

Program „ping” pozwala na wysłanie zapytania ICMP Echo Request na podany adres IP. Jeśli zapytanie zostanie odebrane przez wywoływaną maszynę, powinna ona odpowiedzieć przy pomocy wiadomości ICMP Echo Reply.

- `ping <adres_IP>` - wysłanie pinga na <adres_IP>

W poleceniu ping nie podajemy maski adresu IP.

2.2.3 ifconfig

Polecenie „ifconfig” służy do manipulowania interfejsami sieciowymi.

- `ifconfig` – wypisuje informacje o aktywnych interfejsach
  - w systemie Windows analogiczna komenda to **`ipconfig`**
- `ifconfig <nazwa_interfejsu> <adres_ip>/<maska>` - ustawienie adresu `<adres_ip>/<maska>` na interfejsie `<nazwa_interfejsu>`
  - podawanie maski nie jest konieczne, ale zalecane, gdyż brak maski prowadzi do niejednoznaczności i błędów (co przetestujemy na laboratorium)

2.2.4 route

Polecenie „route” służy do manipulowania tablicą rutingu danej maszyny. Tablica rutingu informuje jakim interfejsem należy wysłać dany pakiet IP. W szczególności może również definiować jaki jest adres bramy. Wpisy w tej tablicy są dodawane automatycznie na podstawie adresów nadanych interfejsom, wynikają z konfiguracji, lub mogą być dodawane ręcznie.

- `route` - wypisuje tablicę rutingu
- `route -n` - wypisuje tablicę rutingu bez nazw (zalecana opcja)
- W systemie Windows, analogiczna komenda do wypisania ścieżek rutingu to **`route print`**
- `route add -net <adres_sieci> netmask <maska> gw <adres_bramy>` - dodaje wpis do tablicy rutingu, który oznacza: „Pakiety kierowane do sieci `<adres_sieci>/<maska>` przesyłaj na adres `<adres_bramy>`”
  - maskę należy podać w postaci 4 oktetów
- `route del -net <adres_sieci> netmask <maska> gw <adres_bramy>`
  - usuwa podany wpis z tablicy rutingu
  - ta sama składnia co w „route add …”, ale zamiast „add” jest „del”

2.2.5 tcpdump

Polecenie „tcpdump” służy do obserwacji ruchu na interfejsie sieciowym.

- `tcpdump -i <nazwa_interfejsu>` - przechwytuje ramki odbierane na interfejsie o nazwie <nazwa_interfejsu> i wyświetla na ekran podstawowe informacje o tych ramkach
  - dodanie flagi -n -jak wyżej, bez nazw
  - dodanie flagi -v -jak wyżej, wyświetla bardziej szczegółowe dane
  - flagi można łączyć, np. `tcpdump -i eth0 -n -v`

2.2.6 arp

Polecenie „arp” wyświetla tablicę ARP – odwzorowanie adresów L2/L3.

- `arp -n` - tablica ARP bez nazw (zalecana opcja)

2.2.7 netstat

Program „netstat” wyświetla listę połączeń sieciowych.

- `netstat -n` - służy do wyświetlania aktywnych połączeń protokołu TCP. Adresy i numery portów są wyrażane numerycznie i nie zostaną zmienione na nazwy

3 Zadania do wykonania

W tej części przedstawione zostaną zadania do wykonania.

3.1 Zespół projektowy

Imię	Nazwisko	nr indeksu	adres otrzymany z VPN
Stanisław	Kwiatkowski	321050	10.141.6.3
Bartosz	Ziemba	324952	

Adres otrzymany z VPN może się zmieniać po restarcie podłączenia. Otrzymany adres można zobaczyć przy pomocy opcji „pokaż status" programu OpenVPN, lub bezpośrednio w terminalu swojego komputera.

3.2 Połączenie z siecią PL-LAB

W celu zapewnienia połączenia z siecią laboratoryjną wykorzystano dostęp przez bramkę VPN. Po podłączeniu do bramki VPN, Komputerowi Studenta powinien zostać przyznany adres z puli 10.141.6.0-10.141.6.255 z maską 9, który umożliwi dostęp do sieci 10.128.0.0/9.

3.2.1.1 Jaki zakres adresów obejmują ww. sieci? Proszę uzupełnić tabelkę:

Sieć 10.141.6.0/24	
adres sieci	10.141.6.0
najmniejszy możliwy adres dostępny dla hosta	10.141.6.1
największy możliwy adres dostępny dla hosta	10.141.6.254
adres broadcast	10.141.6.255

Sieć 10.128.0.0/9	
adres sieci	10.128.0.0
najmniejszy możliwy adres dostępny dla hosta	10.128.0.1
największy możliwy adres dostępny dla hosta	10.255.255.254
adres broadcast	10.255.255.255

3.2.1.2 Do czego służy adres broadcast?

Adres broadcast to specjalny adres IP w sieci, który umożliwia wysłanie danych do wszystkich urządzeń w danej podsieci. Pozwala na komunikacje z wszystkimi hostami, bez znajomości ich adresów IP.

3.2.2 Weryfikacja otrzymanych adresów i tablicy rutingu

Proszę zamieścić zrzuty ekranu z terminala Komputera Studenta, które potwierdzają otrzymanie adresu z puli 10.141.6.0-10.141.6.255 oraz wpisu do tablicy rutingu do sieci 10.128.0.0/9. W przypadku większej ilości tekstu na zrzucie ekranu, proszę zaznaczyć fragment bezpośrednio odnoszący się do polecenia.

```
Unknown adapter OpenVPN TAP-Windows6:

   Connection-specific DNS Suffix  . :
   Link-local IPv6 Address . . . . . : fe80::ec45:9fdd:97c0:a896%54
   IPv4 Address. . . . . . . . . . . : 10.141.6.10
   Subnet Mask . . . . . . . . . . . : 255.128.0.0
   Default Gateway . . . . . . . . . :
```

```
IPv4 Route Table
===========================================================================
Active Routes:
Network Destination        Netmask          Gateway       Interface  Metric
          0.0.0.0          0.0.0.0      192.168.1.1    192.168.1.101     55
       10.128.0.0      255.128.0.0         On-link       10.141.6.10    281
```

Wyświetlenie tablicy routingu w Windows: *route print*

3.2.3 Połączenie na kontenery oraz między kontenerami

Należy zalogować się na kontenery K1 i K3, oraz sprawdzić ich konfigurację sieciową.

	K1	K3
eth0:	172.22.50.3/24	172.22.50.5/24
eth1:	10.10.50.3/24	10.20.50.2/24

Logowanie na kontenery zgodnie z pkt 2.1, w skrócie:

ssh root@10.140.0.129 -p port_num

port_num:

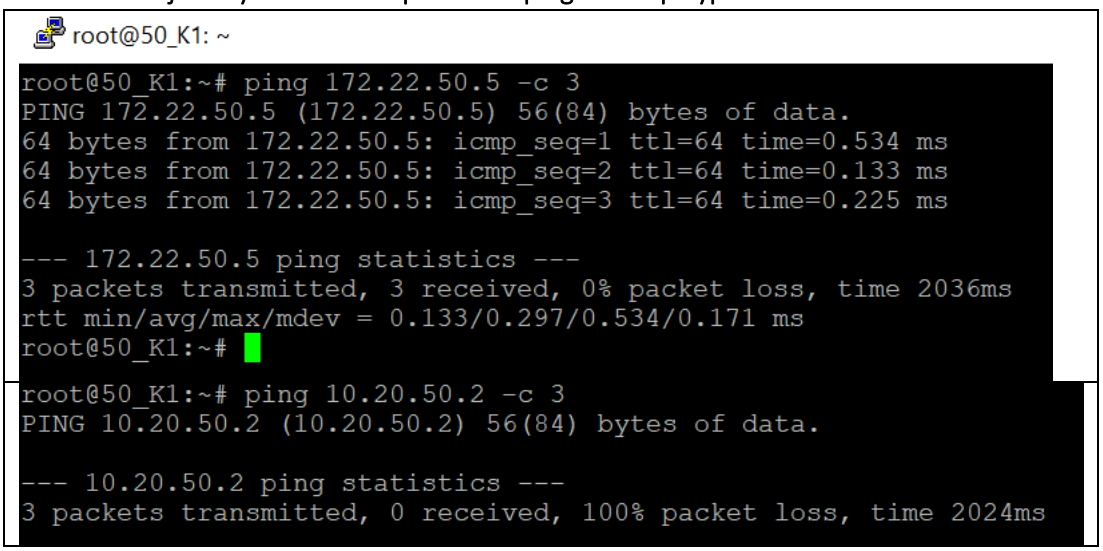
- K1 – 2NN11
- K2 – 2NN12
- K3 – 2NN13
- K4 – 2NN14
- R – 2NN15

Należy spróbować spingować K3 z K1 używając jako adresu docelowego

a) adresu K3/eth0
b) adresu K3/eth1

ping <adres_IP> - wysłanie pinga na <adres_IP>

3.2.3.1 Jaki jest wynik działania polecenia ping w obu przypadkach?



3.2.3.2 Dlaczego otrzymano inny wynik?

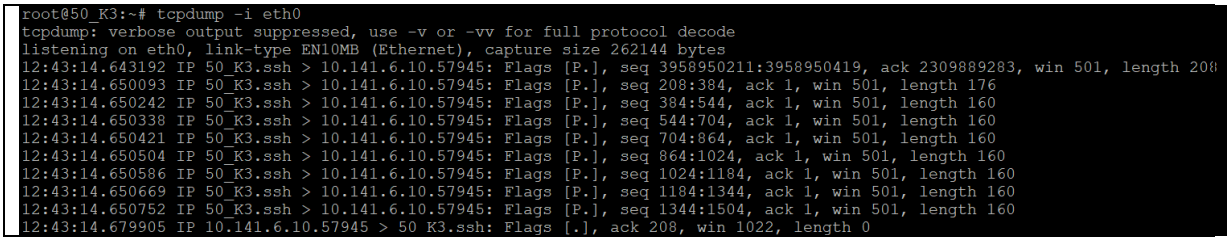
Odp: Z poziomu interfejsów eth0 kontenery znajdują się w tej samej podsieci i mogą bezpośrednio komunikować się między sobą. W przypadku interfejsu eth1, urządzenia połączone są za pomocą routera, który nie posiada jeszcze tablic kierowania pakietów, które pozwalałyby na przesyłanie pakietów ICMP z kontenera 1 do kontenera 2 i na odwrót.

3.3 Obserwowanie ruchu

To zadanie ma na celu zapoznanie Studentów z komendą tcpdump.

Na kontenerze K3 należy uruchomić obserwowanie ruchu na interfejsie **eth0**. Wystarczy kilka sekund obserwacji.

3.3.1.1 Wynik obserwacji:



tcpdump -i <nazwa_interfejsu>

3.3.1.2 Co to za ruch i skąd to wiemy?

Odp: Jest to ruch związany z połączeniem SSH z kontenerem 3. Mówi nam o tym adres IP kontenera 3 na połączeniu Komputer Studenta <-> K3, który wyrażony jest za pomocą “50_K3.ssh”.

3.4 Podstawowa konfiguracja rutingu

To zadanie ma na celu zapoznanie Studentów z podstawową ideą rutingu.

Z jednego komputera należy mieć uruchomione 3 połączenia SSH na K1 oraz 2 połączenia SSH na K3. Na 4 terminalach należy uruchomić obserwowanie ruchu na interfejsie eth0 i eth1 kontenerów K1 i K3 przy pomocy tcpdump.

Uwaga: na interfejsach eth0 będzie bardzo dużo ruchu związanego z połączeniem SSH. W zadaniu interesuje nas obserwacja ping'a, więc w tcpdump można użyć filtru na icmp: **tcpdump -i eth0 icmp**

Z piątego terminala uruchomić ping z K1 na K3 (używając jako adresu docelowego adresu K3/eth1).

3.4.1.1 Czy jakiś ruch jest widoczny na którymkolwiek z interfejsów?

Interfejsy eth0 i eth1 dla **K1**

```
root@50_K1:~# tcpdump -i eth0 icmp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
13:12:06.371334 IP 50_K1 > 10.20.50.2: ICMP echo request, id 4169, seq 1, len
gth 64
13:12:07.401196 IP 50_K1 > 10.20.50.2: ICMP echo request, id 4169, seq 2, len
gth 64
13:12:08.429136 IP 50_K1 > 10.20.50.2: ICMP echo request, id 4169, seq 3, len
gth 64
13:12:09.453152 IP 50_K1 > 10.20.50.2: ICMP echo request, id 4169, seq 4, len
```

root@50_K1: ~

```
root@50_K1:~# tcpdump -i eth1 icmp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth1, link-type EN10MB (Ethernet), capture size 262144 bytes
```

Interfejsy eth0 i eth1 dla **K3**

root@50_K3: ~

```
root@50_K3:~# tcpdump -i eth0 icmp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
```

root@50_K3: ~

```
root@50_K3:~# tcpdump -i eth1 icmp
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth1, link-type EN10MB (Ethernet), capture size 262144 bytes
```

Odp: Ruch widoczny jest tylko na interfejsie eth0 kontenera 1 (pingującego)

Zatrzymać program ping (Ctrl+c).

3.4.1.2 Sprawdzić tablicę rutingu na K1.

```
root@50_K1:~# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         172.22.50.1     0.0.0.0         UG    0      0        0 eth0
10.10.50.0      0.0.0.0         255.255.255.0   U     0      0        0 eth1
172.22.50.0     0.0.0.0         255.255.255.0   U     0      0        0 eth0
```

`route -n` - wypisuje tablicę rutingu bez nazw (zalecana opcja)

3.4.1.3 Na podstawie jakiego pola w nagłówku pakietu IP, pakiet jest przesyłany w odpowiednie miejsce?

Odp: Na podstawie adresu docelowego IP (destination adress).

Z otrzymanej tablicy rutingu wynika, że:

- ruch kierowany do sieci 172.22.NN.0/24* jest kierowany na interfejs eth0 (brak bramy – Gateway=0.0.0.0),
- ruch kierowany do sieci 10.10.NN.0/24 jest kierowany na interfejs eth1 (brak bramy – Gateway=0.0.0.0),
- Ruch do sieci 0.0.0.0/0 jest kierowany na interfejs eth0 na bramę 172.22.NN.1.

Sieć 0.0.0.0/0 oznacza „wszystkie inne adresy".

**maska jest w tym przypadku zapisana w postaci 4 oktetów i wynosi 255.255.255.0, co odpowiada masce o długości 24 bitów.*

3.4.1.4 Dlaczego jedne wpisy posiadają bramę a inne nie?

Odp: Jeśli pakiet ma być przesłany pod adres, którego dane urządzenie nie rozpoznaje (do innej podsieci), pakiet musi zostać wysłany do bramy, która jest pewnego rodzaju pośrednikiem i zna przestrzeń adresową obcej podsieci (brama znajduje się w tej podsieci, więc dla niej nie jest obca). Adresu bramy nie musimy podawać, jeśli chcemy wysyłać pakiety w sieci lokalnej.

3.4.1.5 Gdzie kierowany jest ruch adresowany na adres K3/eth1?

Odp: na interfejs eth0 bramy pod adresem 172.22.50.1 **(brama nie jest równoważna z R)**

Na K1 należy dodać wpis do tablicy rutingu, który skieruje pakiety adresowane do sieci 10.20.NN.0/24 na adres R/eth1.

`route add -net <adres_sieci> netmask <maska> gw <adres_bramy>` - dodaje wpis do tablicy rutingu, który oznacza: „Pakiety kierowane do sieci `<adres_sieci>/<maska>` przesyłaj na adres `<adres_bramy>`"

- maskę należy podać w postaci 4 oktetów

3.4.1.6 Po dodaniu odpowiedniego wpisu, należy udokumentować zmiany w tablicy rutingu.

Zrzut ekranu – tablica rutingu K1

```
root@50_K1:~# route add -net 10.20.50.0 netmask 255.255.255.0 gw 10.10.50.254
root@50_K1:~# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         172.22.50.1     0.0.0.0         UG    0      0        0 eth0
10.10.50.0      0.0.0.0         255.255.255.0   U     0      0        0 eth1
10.20.50.0      10.10.50.254    255.255.255.0   UG    0      0        0 eth1
172.22.50.0     0.0.0.0         255.255.255.0   U     0      0        0 eth0
```

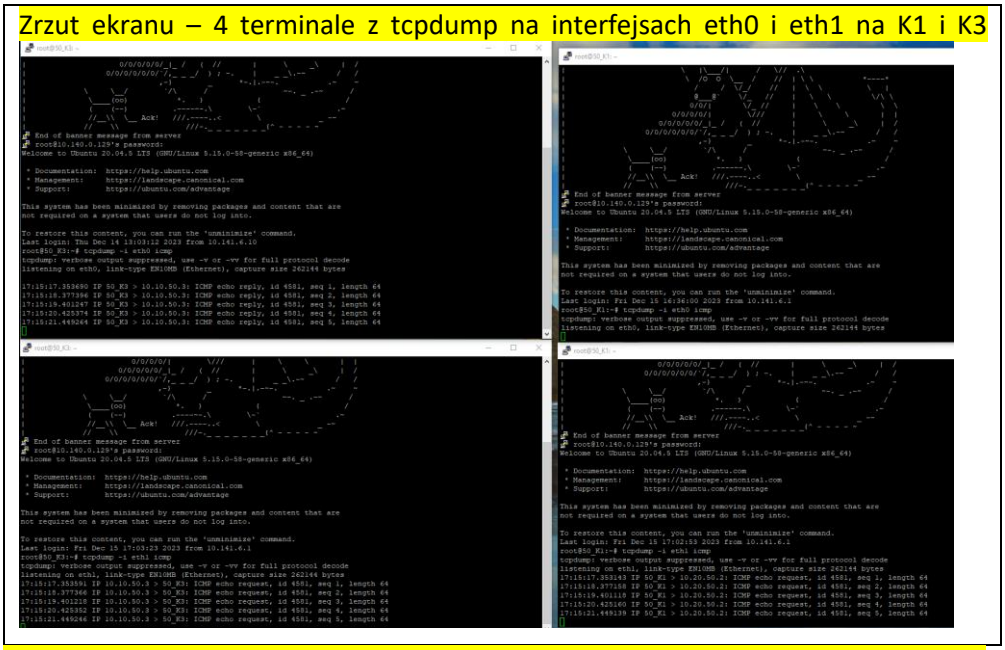
`route -n` - wypisuje tablicę rutingu bez nazw (zalecana opcja)

3.4.1.7 Czemu bramą dla K1 jest R/eth1 a nie R/eth2?

Odp: Ponieważ interfejs eth1 znajduje się w podsieci, w której znajduje się również K1. Dzięki temu K1 rozpoznaje adres bramy i może tam wysyłać pakiety. Gdyby podać adres interfejsu eth2, K1 nie znajdowałby się w sieci razem z tym adresem.

3.4.1.8 Czy po dodaniu na K1 wpisu do tablicy rutingu jakiś ruch jest widoczny na którymkolwiek z interfejsów?

Zrzut ekranu – 4 terminale z tcpdump na interfejsach eth0 i eth1 na K1 i K3



Odp: Tak, ruch widać na obydwu interfejsach eth1, ale tylko po stronie K3 na eth0.

tcpdump -i <nazwa_interfejsu>

3.4.1.9 Dlaczego, skoro do K3 docierają zapytania ICMP echo request, to nie odpowiada on na nie? [podchwytliwe] *(podpowiedź: K3 odpowiada, ale czy kieruje odpowiedzi w dobre miejsce?)*

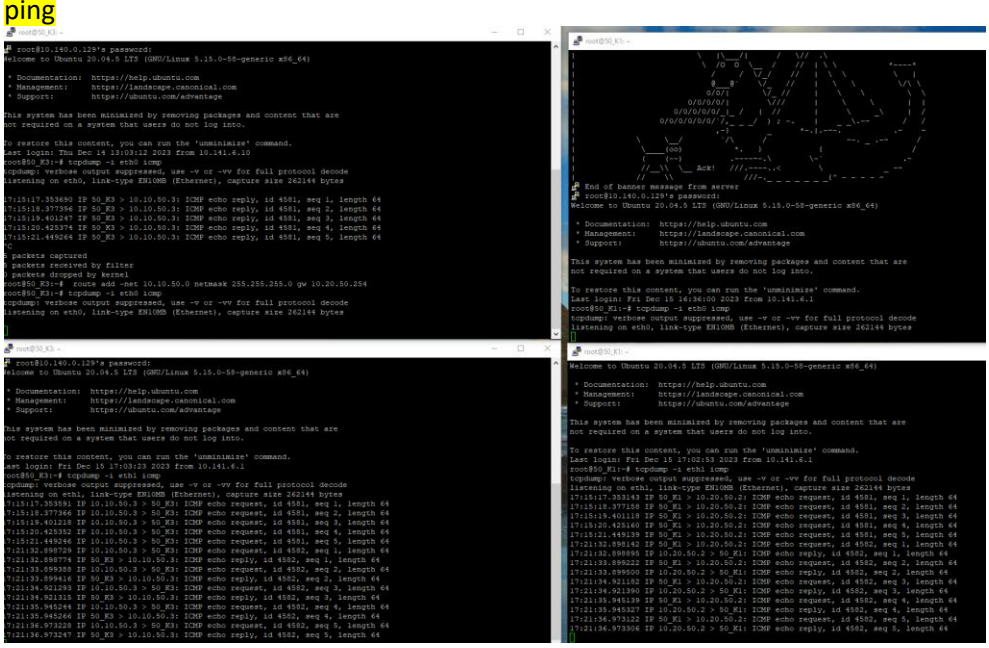
Odp: Odpowiedzi wysyłane z K3 z obydwu interfejsów jako adres docelowy wpisany mają adres interfejsu eth1. Pakiety zwrotne są więc kierowane na adres bramy, a nie na adres R. Aby to rozwiązać, należałoby dodać wpis do tablicy routingu K3, aby pakiety na adres eth1 K1 kierować pod adres eth0 R.

Należy dokonać odpowiedniej zmiany w konfiguracji K3, tak by ping zaczął poprawnie działać.

3.4.1.10 Udokumentować zrzutem ekranu.

```
root@50_K3:~#  route add -net 10.10.50.0 netmask 255.255.255.0 gw 10.20.50.254
```

Zrzut ekranu – 4 terminale z tcpdump na interfejsach eth0 i eth1 na K1 i K3 oraz terminal z programem ping



```
root@50_K1:~# ping 10.20.50.2 -c 5
PING 10.20.50.2 (10.20.50.2) 56(84) bytes of data.
64 bytes from 10.20.50.2: icmp_seq=1 ttl=63 time=0.775 ms
64 bytes from 10.20.50.2: icmp_seq=2 ttl=63 time=0.310 ms
64 bytes from 10.20.50.2: icmp_seq=3 ttl=63 time=0.237 ms
64 bytes from 10.20.50.2: icmp_seq=4 ttl=63 time=0.214 ms
64 bytes from 10.20.50.2: icmp_seq=5 ttl=63 time=0.208 ms

--- 10.20.50.2 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4075ms
rtt min/avg/max/mdev = 0.208/0.348/0.775/0.216 ms
```

`ping <adres_IP>` - wysłanie pinga na <adres_IP>

3.4.1.11 Czy oprócz *ICMP echo request* są widoczne jeszcze jakieś inne ramki? Jeśli tak, jakie?
Odp: tak, widoczne są również ramki ICMP echo reply, jako pakiety zapytań idące do K3.

Nagłówek IP zawiera w sobie informację TTL (*Time to live*). TTL to licznik, który przy każdorazowym przejściu przez interfejs jest zmniejszany o 1. Gdy licznik spadnie do 0, pakiet jest odrzucany a urządzenie, które taki pakiet odrzuciło, odsyła do źródła pakietu odpowiedni komunikat ICMP. Mechanizm ten pozwala ograniczyć negatywne skutki wystąpienia pętli rutingu.

3.4.1.12 Jakiej wartości TTL jest domyślnie przesyłany w ping *(podpowiedź: taka informacja jest szczegółowa)*. Udokumentuj zrzutem ekranu.
Odp: Domyślnie przesyłany TTL wynosi 64 przejścia przez interfejsy. W naszym przypadku otrzymujemy pakiety z TTL wynoszącym 63, ponieważ każdorazowo pakiet musi przejść przez router (R). Inaczej byłoby w przypadku, gdybyśmy komunikowali ze sobą K1 i K3 w sieci 172.22.50.0, otrzymywalibyśmy wtedy 64, ponieważ pakiety trafiałyby do interfejsów bezpośrednio.

Zrzut ekranu

```
root@50_K1:~# ping 10.20.50.2 -c 5
PING 10.20.50.2 (10.20.50.2) 56(84) bytes of data.
64 bytes from 10.20.50.2: icmp_seq=1 ttl=63 time=0.775 ms
64 bytes from 10.20.50.2: icmp_seq=2 ttl=63 time=0.310 ms
64 bytes from 10.20.50.2: icmp_seq=3 ttl=63 time=0.237 ms
64 bytes from 10.20.50.2: icmp_seq=4 ttl=63 time=0.214 ms
64 bytes from 10.20.50.2: icmp_seq=5 ttl=63 time=0.208 ms

--- 10.20.50.2 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4075ms
rtt min/avg/max/mdev = 0.208/0.348/0.775/0.216 ms
```

`tcpdump -i <nazwa_interfejsu>` - przechwytuje ramki odbierane na interfejsie o nazwie <nazwa_interfejsu> i wyświetla na ekran podstawowe informacje o tych ramkach

- dodanie flagi -n -jak wyżej, bez nazw
- dodanie flagi -v -jak wyżej, wyświetla bardziej szczegółowe dane
- flagi można łączyć, np. `tcpdump -i eth0 -n -v`

Jeśli ping przesyła się poprawnie, można zamknąć nadmiarowe terminale.

3.5 ARP

To zadanie ma na celu prezentację zachowania protokołu ARP.

Protokół ARP służy do określania jaka maszyna fizyczna (adres MAC) kryje się za adresem IP.

Należy zalogować się na K1, K2, K3, R

3.5.1.1 Sprawdzić na tych maszynach tablice ARP.

Zrzut ekranu – arp, 4 terminale

root@50_K1: ~

```
root@50_K1:~# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
172.22.50.1              ether   02:42:98:a2:bf:5f   C                     eth0
root@50_K1:~#
```

root@50_K3: ~

```
root@50_K3:~# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
172.22.50.1              ether   02:42:98:a2:bf:5f   C                     eth0
root@50_K3:~#
```

root@50_K2: ~

```
root@50_K2:~# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
172.22.50.1              ether   02:42:98:a2:bf:5f   C                     eth0
root@50_K2:~#
```

root@50_R: ~

```
root@50_R:~# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
172.22.50.1              ether   02:42:98:a2:bf:5f   C                     eth0
root@50_R:~#
```

`arp -n` - tablica ARP bez nazw (zalecana opcja)

Należy otworzyć jeszcze jedno połączenie do K1, tak by w sumie widzieć 5 terminali (2 razy K1, oraz K2, K3, R). Na K1, K2, K3, R należy uruchomić nasłuchiwanie ruchu na interfejsie eth1

Na wolnym terminalu K1 należy uruchomić ping na adres 10.10.NN.100.

3.5.1.2 Jaki ruch jest obserwowany na K1, K2, K3, R? Udokumentować zrzutem ekranu.

Zrzut ekranu – 4 terminale

```
root@50_K1:~# ping 10.10.50.100 -c 5
PING 10.10.50.100 (10.10.50.100) 56(84) bytes of data.
From 10.10.50.3 icmp_seq=1 Destination Host Unreachable
From 10.10.50.3 icmp_seq=2 Destination Host Unreachable
From 10.10.50.3 icmp_seq=5 Destination Host Unreachable

--- 10.10.50.100 ping statistics ---
5 packets transmitted, 0 received, +3 errors, 100% packet l
pipe 4
```

root@50_K1: ~

```
root@50_K1:~# tcpdump -i eth1
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth1, link-type EN10MB (Ethernet), capture size 262144 bytes
18:07:57.955982 ARP, Request who-has 10.10.50.100 tell 50_K1, length 28
18:07:58.985113 ARP, Request who-has 10.10.50.100 tell 50_K1, length 28
18:08:00.009106 ARP, Request who-has 10.10.50.100 tell 50_K1, length 28
18:08:01.033146 ARP, Request who-has 10.10.50.100 tell 50_K1, length 28
18:08:02.057089 ARP, Request who-has 10.10.50.100 tell 50_K1, length 28
18:08:03.081098 ARP, Request who-has 10.10.50.100 tell 50_K1, length 28
```

root@50_K3: ~

```
root@50_K3:~# tcpdump -i eth1
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth1, link-type EN10MB (Ethernet), capture size 262144 bytes
```

root@50_K2: ~

```
root@50_K2:~# tcpdump -i eth1
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth1, link-type EN10MB (Ethernet), capture size 262144 bytes
18:07:57.956012 ARP, Request who-has 10.10.50.100 tell ipv4_50_K1_1.ipv4_50_net_SC1_4, length 28
18:07:58.985219 ARP, Request who-has 10.10.50.100 tell ipv4_50_K1_1.ipv4_50_net_SC1_4, length 28
18:08:00.009176 ARP, Request who-has 10.10.50.100 tell ipv4_50_K1_1.ipv4_50_net_SC1_4, length 28
18:08:01.033170 ARP, Request who-has 10.10.50.100 tell ipv4_50_K1_1.ipv4_50_net_SC1_4, length 28
18:08:02.057132 ARP, Request who-has 10.10.50.100 tell ipv4_50_K1_1.ipv4_50_net_SC1_4, length 28
18:08:03.081130 ARP, Request who-has 10.10.50.100 tell ipv4_50_K1_1.ipv4_50_net_SC1_4, length 28
```

root@50_R: ~

```
root@50_R:~# tcpdump -i eth1
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth1, link-type EN10MB (Ethernet), capture size 262144 bytes
18:07:57.956014 ARP, Request who-has 10.10.50.100 tell ipv4_50_K1_1.ipv4_50_net_SC1_4, length 28
18:07:58.985222 ARP, Request who-has 10.10.50.100 tell ipv4_50_K1_1.ipv4_50_net_SC1_4, length 28
18:08:00.009179 ARP, Request who-has 10.10.50.100 tell ipv4_50_K1_1.ipv4_50_net_SC1_4, length 28
18:08:01.033173 ARP, Request who-has 10.10.50.100 tell ipv4_50_K1_1.ipv4_50_net_SC1_4, length 28
18:08:02.057137 ARP, Request who-has 10.10.50.100 tell ipv4_50_K1_1.ipv4_50_net_SC1_4, length 28
18:08:03.081133 ARP, Request who-has 10.10.50.100 tell ipv4_50_K1_1.ipv4_50_net_SC1_4, length 28
```

3.5.1.3 Dlaczego *ARP request* jest widoczny na K1, K2, R? Dlaczego nie jest widoczny na K3? Dlaczego nie obserwujemy *ARP reply*?

Odp: Ponieważ teoretyczne urządzenie pod adresem 10. 10. 50. 100 znajduje się w tej samej sieci lokalnej co K1. Zapytanie ARP jest wobec tego wysyłane pod adres broadcast sieci lokalnej, czyli 10.10.50.255. Zapytanie jest wysyłane do wszystkich hostów w tej sieci, czyli K1, K2 oraz R (nie ma ustawionych reguł, które przekazałyby zapytanie ARP na broadcast drugiej sieci)

Można zakończyć nasłuchiwanie ruchu na K1, K2, K3, R

3.6 Aktywne połączenia

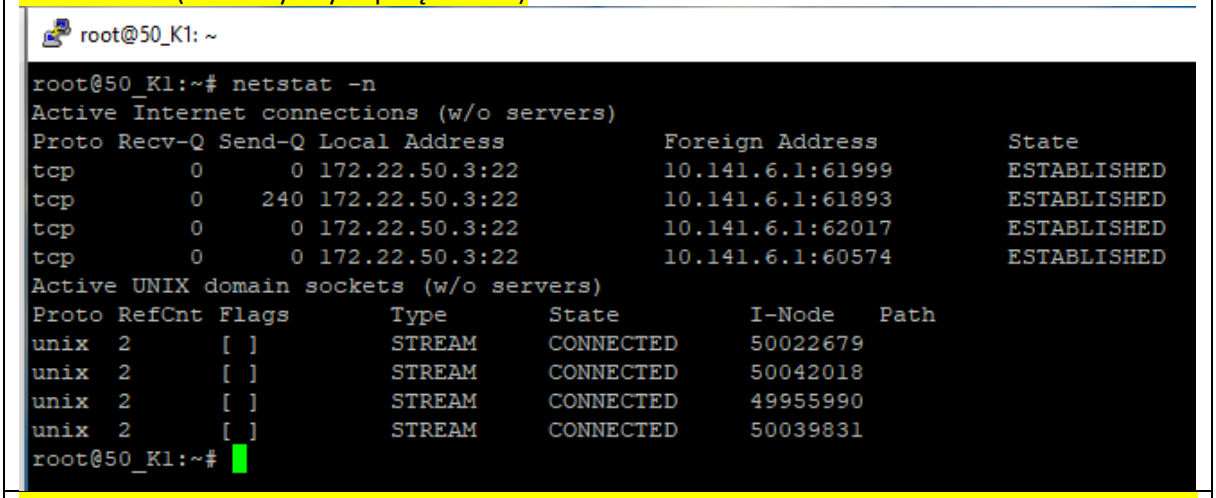
To zadanie ma na celu zapoznanie Studentów z komendą netstat.

Należy zestawić kilka (min. 3) połączeń SSH do K1 a następnie sprawdzić na nim listę aktywnych połączeń sieciowych. Podobną operację wykonać dla K2.

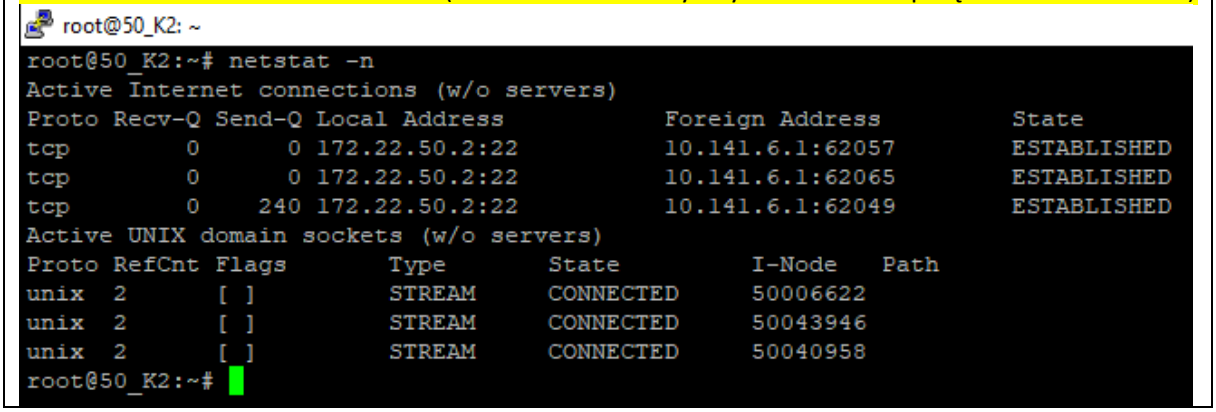
`netstat -n` - służy do wyświetlania aktywnych połączeń protokołu TCP. Adresy i numery portów są wyrażane numerycznie i nie zostaną zmienione na nazwy

3.6.1.1 Udokumentować:

Zrzut ekranu (lista aktywnych połączeń K1)



Zrzut ekranu (lista aktywnych połączeń K2)



Po wykonaniu zrzutów ekranu można zamknąć nadmiarowe połączenia SSH.

3.6.1.2 Proszę uzupełnić tekst:

Obserwowane w sekcjach Local/Foreign Address adresy składają się z dwóch części oddzielonych znakiem „:”. Pierwsza część to IP hosta, druga to port, z którego łączymy się przez SSH. Przykładowo, serwer SSH na K1 oraz K2 nasłuchuje na tym samym porcie nr 22, dlatego w sekcji Local Address widnieją wpisy 172.22.50.2:22 i 172.22.50.3:22 (podać dla dwóch Komputerów). Z serwerem SSH na K1 nawiązanych jest 4 połączeń. Wartości Foreign Address identyfikują klientów SSH – ich adresy IPv4 i używane przez nich porty do komunikacji SSH.

3.6.1.3 Dlaczego serwer SSH jest zawsze* uruchamiany na tym samym porcie, natomiast poszczególni klienci SSH korzystają z losowych (wysokich) numerów portów? *(podpowiedź: kto inicjuje połączenie, jakie informacje są przesyłane w nagłówku IP? Także* https://en.wikipedia.org/wiki/List_of_TCP_and_UDP_port_numbers *)*

Odp: dlatego, że serwer musi nasłuchiwać z tego portu na nadchodzące połączenie i port musi być znany klientowi, przez co powinien być stały. Serwer natomiast łączy się z dowolnym przychodzącym połączeniem, bez względu na jego port. Wybór wysokich numerów portów ma na celu zmniejszenie ryzyka zajęcia już używanego portu i wybrania nieużywanego.

**serwer SSH można uruchomić na dowolnym porcie, ale wymaga to zmiany domyślnej konfiguracji. W przypadku użycia portu innego niż domyślny, klient musi podać ten port (flaga -p).*

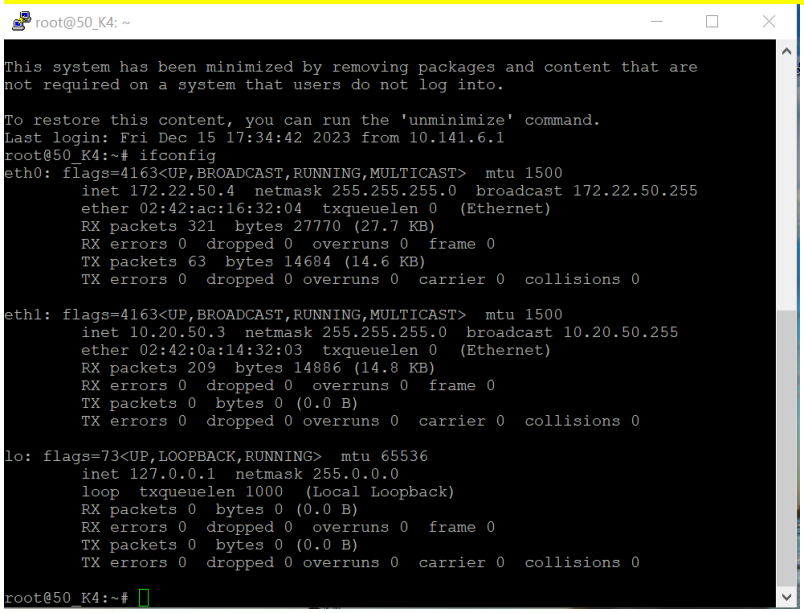
Proszę zauważyć, że łączymy się na adres 10.140.0.129 i porty 2NN11, 2NN12, …, i także korzystamy z flagi -p. Wszystkie kontenery mają uruchomione serwery SSH na porcie 22. Na serwerze 10.140.0.129 ustawione jest przekierowanie portów, tzn. dane kierowane na adres np. 10.140.0.129:2NN11 są przesyłane na adres 172.22.NN.XX:22 (XX jest przydzielane automatycznie przez program docker). Mechanizm przekierowania portów nie będzie badany w ramach tego ćwiczenia.

3.7 Zmiana adresu IP (awaria i odzyskiwanie)

To zadanie ma na celu wywołanie awarii dostępu spowodowanej błędną konfiguracją interfejsu sieciowego. Następnie zaprezentowany zostanie sposób odzyskania kontroli nad sytuacją.

3.7.1.1 Należy sprawdzić konfigurację adresów sieciowych na K4.

Zrzut ekranu



`ifconfig` – wypisuje informacje o aktywnych interfejsach

`ifconfig <nazwa_interfejsu>`

3.7.1.2 Zmienić adres K4/eth1 na 192.168.NN.100/24 bez podawania maski w poleceniu ifconfig. (polecenie: ifconfig eth1 192.168.NN.100)

Zrzut ekranu - konfiguracja K4/eth1

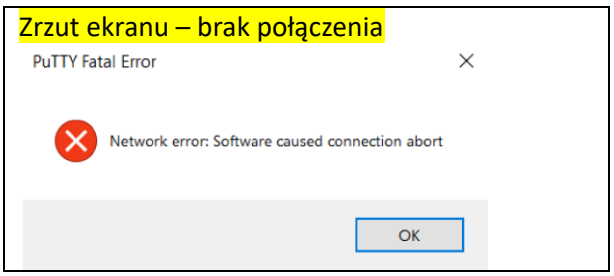
```
root@50_K4:~# ifconfig eth1 192.168.50.100
root@50_K4:~# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 172.22.50.4  netmask 255.255.255.0  broadcast 172.22.50.255
        ether 02:42:ac:16:32:04  txqueuelen 0  (Ethernet)
        RX packets 519  bytes 46302 (46.3 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 199  bytes 32368 (32.3 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.50.100  netmask 255.255.255.0  broadcast 192.168.50.255
        ether 02:42:0a:14:32:03  txqueuelen 0  (Ethernet)
        RX packets 209  bytes 14886 (14.8 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

3.7.1.3 Czy została ustawiona maska 24?

Odp: Tak, 255.255.255.0 -> /24

3.7.1.4 Zmienić adres K4/eth1 na 10.20.NN.100/24 bez podawania maski w poleceniu ifconfig. (polecenie: ifconfig eth1 10.20.NN.100)

Zrzut ekranu – brak połączenia



Jeśli nie zostało utracone połączenie, proszę to udokumentować i ustawić adres ręcznie na 10.20.NN.100/8 (`ifconfig eth1 10.20.NN.100/8`). W takim przypadku należy pominąć punkt 3.7.1.5, 3.7.1.6, 3.7.1.7

Po ustawieniu adresu w ten sposób powinna zostać utracona łączność z K4. Kolejne próby łączenia się na adres 10.140.0.129 -p 2NN14 się nie udają. Studenci przypominają sobie jednak, że wszystkie kontenery są podłączone do sieci *Lab_control_SC* (interfejsami eth0).

Należy z któregokolwiek Komputera lub Rutera (K1, K2, K3, R) zalogować się przez SSH na adres K4/eth0.

Polecenie: `ssh root@172.22.NN.X`

X należy odczytać z wykonanego na początku tego zadania zrzutu ekranu konfiguracji adresów sieciowych na K4

3.7.1.5 Po zalogowaniu na K4 należy sprawdzić jaki adres został ustawiony na eth1.

Zrzut ekranu - konfiguracja K4/eth1

```
root@50_K4:~# ifconfig eth1
eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.20.50.100  netmask 255.0.0.0  broadcast 10.255.255.255
        ether 02:42:0a:14:32:03  txqueuelen 0  (Ethernet)
        RX packets 209  bytes 14886 (14.8 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 27  bytes 1134 (1.1 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

root@50_K4:~#
```

`ifconfig <nazwa_interfejsu>`

3.7.1.6 Jaka została ustawiona maska sieci dla adresu 10.20.NN.100, gdy nie podaliśmy wprost, że chcemy użyć 24 bitowej maski?

Odp: 255.0.0.0, czyli maska /8 (8 bitowej).

3.7.1.7 Dlaczego podając do komendy **`ifconfig`** adresy 192.68.NN.100 oraz 10.20.NN.100 (bez maski) domyślnie zostały przyjęte maski o różnej długości?

Odp: nie precyzując maski, użyta zostaje ta ze (starej), domyślnej klasy adresowej IP, tzn. w przypadku adresu 192.68.50.100 jest to klasa C o masce 24 bitowej, a w przypadku 10.20.50.100 jest to klasa A o masce 8 bitowej.

3.7.1.8 Należy sprawdzić tablicę rutingu na K4

Zrzut ekranu

```
root@50_K4:~# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         172.22.50.1     0.0.0.0         UG    0      0        0 eth0
10.0.0.0        0.0.0.0         255.0.0.0       U     0      0        0 eth1
172.22.50.0     0.0.0.0         255.255.255.0   U     0      0        0 eth0
root@50_K4:~#
```

`route -n` - wypisuje tablicę rutingu bez nazw (zalecana opcja)

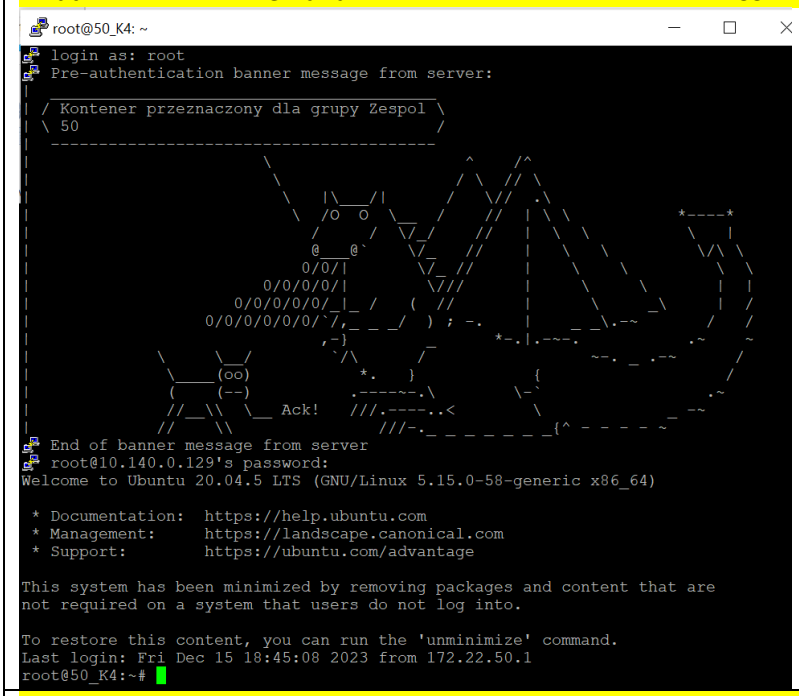
3.7.1.9 Na podstawie tablicy rutingu należy odpowiedzieć, gdzie kierowany jest ruch adresowany do Komputera Studenta?

Odp: Ruch do Komputera Studenta o adresie 10.141.6.3 należy do sieci 10.0.0.0/8 i kierowany jest na interfejs eth1, bez określonej bramy.

Należy we właściwy sposób ustawić adres 10.20.NN.100/24 na K4/eth1. Polecenie: `ifconfig eth1 10.20.NN.100/24`

3.7.1.10 Należy udokumentować: Możliwość połączenia do K4 z Komputera Studenta, konfigurację K4/eth1, tablicę rutingu na K4.

Zrzut ekranu – SSH na K4



Zrzut ekranu - konfiguracja K4/eth1

```
root@50_K4:~# ifconfig eth1
eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.20.50.100  netmask 255.255.255.0  broadcast 10.20.50.255
        ether 02:42:0a:14:32:03  txqueuelen 0  (Ethernet)
        RX packets 209  bytes 14886 (14.8 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 27  bytes 1134 (1.1 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Zrzut ekranu – tablica rutingu K4

```
root@50_K4:~# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         172.22.50.1     0.0.0.0         UG    0      0        0 eth0
10.20.50.0      0.0.0.0         255.255.255.0   U     0      0        0 eth1
172.22.50.0     0.0.0.0         255.255.255.0   U     0      0        0 eth0
```