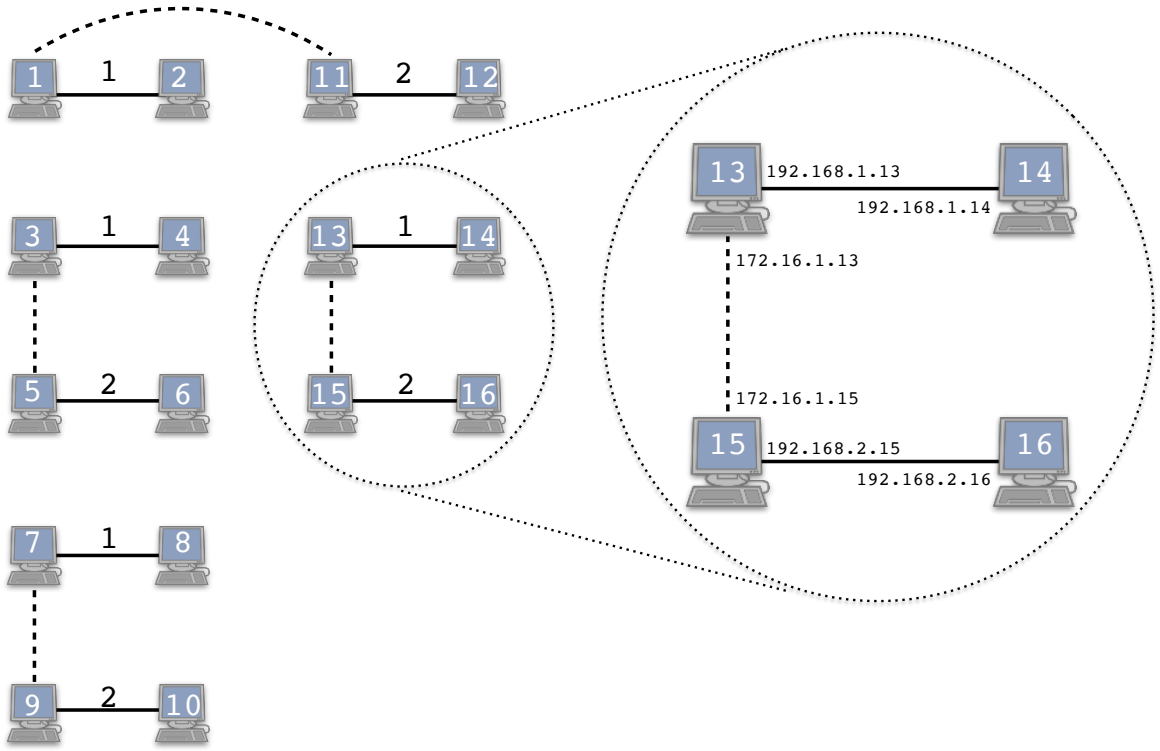
Warsztaty z Sieci komputerowych
Lista 2

Topologia sieci na te zajęcia została przedstawiona poniżej; każda czwórka komputerów jest osobną strukturą niepołączoną z niczym innym, jak na rysunku poniżej. Linią ciągłą oznaczono połączenia między interfejsami `enp1s0`, zaś przerywaną — między interfejsami `enp3s0`. Celem dzisiejszych zajęć jest konfiguracja interfejsów sieciowych i routingu pomiędzy sieciami (sieci są dwupunktowymi połączeniami zaznaczonymi na rysunku).

Komputery, które mają połączone karty `enp3s0` (komputery o nieparzystych numerach) będziemy nazywać komputerami *typu L*, zaś pozostałe komputerami *typu R*.



Zadanie 1. Usuń istniejącą konfigurację sieciową interfejsu `enp3s0` poleceniem

```
#> ifdown enp3s0
```

Wszystkim komputerom uaktywnij kartę sieciową `enp1s0` i przypisz jej odpowiedni adres IP poleceniami

```
#> ip link set up dev enp1s0
#> ip addr add 192.168.y.x/24 dev enp1s0
```

gdzie $x \in \{1, \ldots, 16\}$ jest numerem komputera, zaś $y \in \{1, 2\}$ jest numerem krawędzi (patrz rysunek wyżej). Przykładowo karta `enp1s0` komputera nr 10 powinna otrzymać adres `192.168.2.10`.

Następnie interfejsom `enp3s0` komputerów typu L przypisz adres `172.16.1.`*x*`/16`, gdzie *x* jest numerem komputera. W tym celu uruchom polecenia

```
#> ip link set up dev enp3s0
#> ip addr add 172.16.1.x/16 dev enp3s0
```

Wyświetl aktualnie skonfigurowane interfejsy i tablicę routingu poleceniami

```
$> ip addr
$> ip route
```

a także wykorzystując starsze narzędzia `ifconfig` i `route`:

```
#> ifconfig
#> route
```

W przypadku błędu wszystkie adresy IP przypisane do danego interfejsu można usunąć poleceniem `ip addr flush dev` *`interfejs`*. Zauważ, że polecenie `ip route` wyświetla przy trasach do sieci opis `proto kernel`. Oznacza to, że trasa do danej sieci została dodana automatycznie przez jądro systemu podczas dodawania adresu do interfejsu.

Za pomocą programu `ping` sprawdź, czy połączone bezpośrednio ze sobą komputery „widzą się" wzajemnie.

Zadanie 2. Zauważ, że masz skonfigurowany interfejs `lo`. Pingnij adres pętli lokalnej `127.0.0.1`. Zauważ, że komunikaty dochodzą pomimo tego, że odpowiedni wpis nie jest wyświetlany poleceniem `ip route`. Te dodatkowe wpisy w tablicy można wyświetlić poleceniem

```
$> ip route list table local
```

Przeanalizuj poszczególne wiersze. Zwróć uwagę na adresy rozgłoszeniowe i różnice w polach `scope`.

Włącz Wiresharka i rozpocznij nasłuchiwanie na wszystkich interfejsach. Zaobserwuj, co jest wypisywane w konsoli oraz jakie pakiety są wysyłane i odbierane jeśli pingasz:

1. adres `127.0.0.1`;
2. swój własny adres IP przypisany do interfejsu `enp1s0`;
3. adres IP sąsiedniego komputera podłączonego do interfejsu `enp1s0`;
4. adres rozgłoszeniowy sieci podłączonej do interfejsu `enp1s0` (poleceniem `ping -b 192.168.1.255` lub `ping -b 192.168.2.255`);
5. nieistniejący adres IP należący do sieci podłączonej do interfejsu `enp1s0`;
6. adres z sieci, do której nie jesteś bezpośrednio podłączony, np. `10.10.10.10`.

Porównaj otrzymane komunikaty, przesyłane pakiety i czasy reakcji.

Zadanie 3. Z komputera typu R sprawdź osiągalność karty `enp3s0` jego sąsiada typu L (połączonego z nim za pomocą karty `enp1s0`):

```
$> ping 172.16.1.x
```

gdzie x jest numerem komputera sąsiada. Przykładowo na komputerze nr 6 należy wydać polecenie `ping 172.16.1.5`. Adres ten jest nieosiągalny, gdyż nadawca nie wie, jak dostać się do sieci `172.16.0.0/16`. Spróbujmy to naprawić dodając na komputerze typu R trasę domyślną przechodzącą przez osiągalną bezpośrednio kartę sąsiada:

```
#> ip route add default via 192.168.y.x
```

gdzie $x \in \{1, \ldots, 16\}$ jest numerem sąsiedniego komputera typu L, zaś $y \in \{1, 2\}$ jest numerem incydentnej krawędzi (patrz rysunek). Napis `default` jest skrótem notacyjnym na `0.0.0.0/0`. Przykładowo na komputerze nr 16 należy wydać polecenie `ip route add default via 192.168.2.15`. Jeśli pomylisz się wpisując polecenie `ip route`, dodaną pomyłkowo trasę możesz skasować zamieniając parametr `add` na `del`.

Wyświetl bieżącą tablicę routingu a następnie spróbuj wykonać poprzednie polecenie `ping` (powinno zakończyć się sukcesem).

Czy inne adresy z sieci `172.16.0.0/16` są osiągalne? Aby to sprawdzić, pingnij drugi adres IP z sieci `172.16.0.0/16`, tj. należący do drugiego komputera typu L. Przykładowo na komputerze nr 12 należy wydać polecenie `ping 172.16.1.1`. Zapamiętaj to polecenie; będziemy je określać mianem „pingnij najdalszy komputer typu L".

Co jest przyczyną niepowodzenia? Jaki komunikat otrzymujesz? Na komputerach typu L sprawdź Wiresharkiem, że odpowiedni komunikat ICMP jest otrzymywany i przekazywany do komputera docelowego. Dlaczego więc nie jest odsyłana odpowiednia odpowiedź?

Zadanie 4. Na komputerze typu L dodaj trasę prowadzącą do tej sieci, która nie jest do niego bezpośrednio połączona:

```
#> ip route add 192.168.y.0/24 via 172.16.1.x
```

gdzie x jest numerem sąsiedniego komputera L, zaś $y \in \{1, 2\}$ jest numerem krawędzi odpowiadającej tej sieci. Przykładowo na komputerze nr 7 należy wydać polecenie `ip route add 192.168.2.0/24 via 172.16.1.9`. Na komputerze typu R pingnij najdalszy komputer typu L. Dlaczego ostatnie polecenie `ip route` pomogło w otrzymywaniu odpowiedzi na `ping`?

Przedstawiony na rysunku obraz sieci nie jest kompletny. W rzeczywistości komputery typu L są podłączone za pośrednictwem routera `172.16.255.252` do Internetu. Na komputerach typu L skonfiguruj trasę domyślną do Internetu poleceniem

```
#> ip route add default via 172.16.255.252
```

Ze wszystkich komputerów pingnij jakiś znany Ci istniejący adres IP (np. `8.8.8.8`). Obejrzyj Wiresharkiem wszystkie przesyłane komunikaty. Dlaczego ping z komputerów typu L udaje się, a z komputerów typu R nie? W którym momencie zawodzi komunikacja: na trasie do `8.8.8.8`, czy na trasie powrotnej? Kogo należałoby powiadomić o sieciach `192.168.y.x`? Jak można inaczej rozwiązać ten problem?

Zadanie 5. Na wszystkich komputerach zdekonfiguruj warstwę sieciową i wyłącz interfejsy `enp1s0` i `enp3s0` poleceniami

```
#> ip addr flush enp1s0
#> ip link set down dev enp1s0
```

```
#> ip addr flush enp3s0
#> ip link set down dev enp3s0
```

Na wszystkich komputerach uzyskaj „systemową” konfigurację interfejsu `enp3s0` poleceniem

```
#> ifup enp3s0
```

Polecenia `ifup` i `ifdown` są wysokopoziomowymi programami Debiana konfigurującymi i dekonfigurującymi interfejsy sieciowe na podstawie informacji zawartych w pliku `/etc/network/interfaces` (obejrzyj ten plik).

Obejrzyj przypisany w ten sposób adres IP i tablice routingu poleceniami `ip addr` i `ip route`. Teraz wszystkie komputery są połączone interfejsem `enp3s0` z siecią `172.16.0.0/16` i za pośrednictwem routera `172.16.255.252` z resztą Internetu. (Fizyczne połączenie zawsze istniało, ale na komputerach typu R interfejs `enp3s0` był nieaktywny).

Wykonaj polecenie `traceroute` do jakiegoś znanego Ci adresu IP (np. `8.8.8.8`) lub nazwy domeny (np. `wikipedia.com`). Zaobserwuj przesyłane pakiety Wiresharkiem. Wypróbuj też wariant programu `traceroute` wykorzystujący pakiety *ICMP echo request*:

```
#> traceroute -I 8.8.8.8
```

W razie potrzeby w Wiresharku odfiltruj wszystkie pakiety poza tymi, które są skierowane do Twojego adresu IP, lub z niego wychodzą wpisując w Wiresharku filtr `ip.addr == Twój_adres_IP`. (Filtr `ip.addr == a.b.c.d` jest równoważny filtrowi `ip.src == a.b.c.d || ip.dst == a.b.c.d`).

Uruchom wirtualną maszynę (program VirtualBox, maszyna Virbian0. Do wirtualnej maszyny możesz zalogować się na konto `student` (hasło `student.8`) i na konto `root` (hasło `root.8`). Pobierz do maszyny wirtualnej program `icmp_receive.c` z wykładu. Skompiluj go i uruchom. Wykonaj w maszynie wirtualnej polecenie `ping 8.8.8.8`. Zaobserwuj i porównaj pakiety odbierane w Wiresharku i programie `icmp_receive`. Powtórz eksperyment wykorzystując polecenia `traceroute 8.8.8.8` i `traceroute -I 8.8.8.8`.

Lista i materiały znajdują się pod adresem http://www.ii.uni.wroc.pl/~mbi/dyd/.

Marcin Bieńkowski