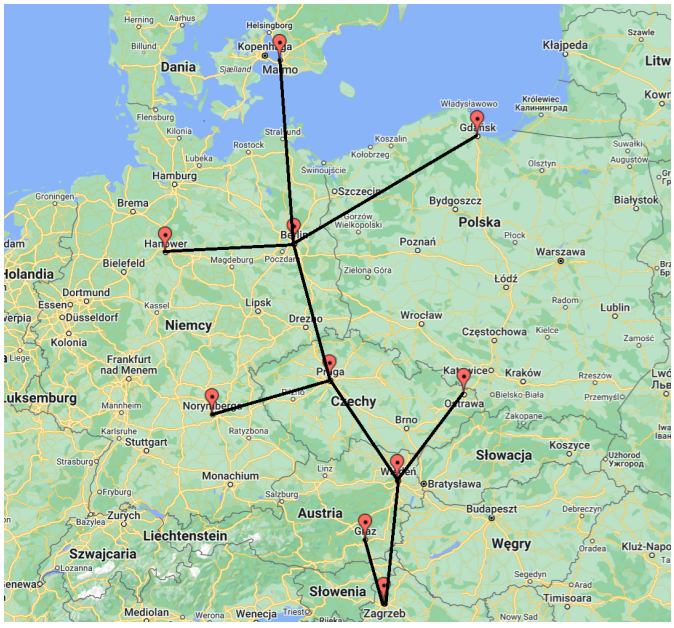
SCHT Laboratorium 1 - Płaszczyzna danych sieci

Tymoteusz Malec, Kacper Średnicki

Politechnika Warszawska, Cyberbezpieczeństwo 23Z

28 października, 2023

Spis treści



Rys. 1. Mapa sieci - wybrane miasta i połączenia między nimi

1. Wstęp

Laboratorium nr 1 z przedmiotu Sieci i Chmury Teleinformatyczne dotyczyło płaszczyzny danych sieci. Przed przystąpieniem do wykonania zadań zainstalowaliśmy maszynę wirtualną z **emulatorem sieci Mininet**, zapoznaliśmy się z jego dokumentacją oraz przetestowaliśmy jego podstawowe funkcjonalności.

2. Część 1

W pierwszej części ćwiczenia zaprojektowaliśmy topologię naszej sieci oraz ustaliliśmy jej charakterystyki. Następnie rozpoczęliśmy badanie parametrów przesyłu danych oraz ich związku z charakterystykami łączy.

2.1. Zadanie 1

W ramach zadania pierwszego wybraliśmy 10 europejskich miast oraz połączenia między nimi tak, by przygotowana sieć miała strukturę drzewiastą. Mapę sieci przedstawiono na rysunku 1. Wybrane miasta to: Malmo (h1), Berlin (h2), Gdańsk (h3), Hanower (h4), Praga (h5), Norymberga (h6), Wiedeń (h7), Ostrawa (h8), Zagrzeb (h9), Graz (h10).

W języku Python przygotowaliśmy słownik miast z ich współrzędnymi geograficznymi oraz listę połączeń między nimi. W każdym mieście umieszczony został jeden switch, a do każdego switcha dołączony jeden host. W celu zdefiniowania realistycznych opóźnień dla każdego z łączy użyliśmy wzoru $\frac{s\sqrt{2}}{v}$, gdzie s jest odległością pomiędzy poszczególnymi miastami wyrażoną w kilometrach, a v szybkością propagacji sygnału świetlnego w światłowodzie wynoszącą $200\frac{km}{ms}$. Odległości uzyskaliśmy ze wzoru Haversine, korzystającego ze współrzędnych geograficznych miast. Dzięki tym obliczeniom, wykonanym przez przygotowany w Pythonie skrypt, uzyskaliśmy unikatowe i realistyczne wartości opóźnień na każdym z łączy, wyrażone w milisekundach. Do testów wykonanych w ramach pierwszej części, wartość przepustowości wszystkich łączy ustaliliśmy na $\mathbf{1000}\frac{\mathbf{Mb}}{\mathbf{s}}$.

Wykorzystując polecenie `ifconfig` uzyskaliśmy adres potrzebny do przesłania na maszynę wirtualną skryptu z topologią sieci. Skrypt przesłaliśmy przy użyciu narzędzia `pscp` poleceniem: `pscp main.py mininet@192.168.56.102:/home/mininet`. Komendą `pscp` mogliśmy również przesyłać pliki z maszyny wirtualnej na nasz komputer. Na maszynie wirtualnej uruchomiliśmy naszą sieć poleceniem: `sudo mn -custom main.py -topo mytopo -link tc`. Po uruchomieniu, nasz program drukuje obliczone wartości opóźnień na poszczególnych łączach, aby ułatwić nam analizę wyników testów.

```
Malmo-Berlin: 2.39ms
Hanower-Berlin: 1.79ms
Berlin-Gdansk: 2.85ms
Berlin-Praga: 1.98ms
Praga-Norymberga: 1.78ms
Praga-Wieden: 1.81ms
Wieden-Ostrawa: 1.59ms
Wieden-Zagrzeb: 1.90ms
Zagrzeb-Graz: 1.02ms
```

2.2. Zadanie 2

Przed rozpoczęciem badań przygotowanej sieci, sprawdziliśmy poprawność jej definicji i konfiguracji. Wykonaliśmy polecenie `pingall`, które udowodniło, że wszystkie hosty skutecznie wymieniają się między sobą pakietami. Następnie podjęliśmy się weryfikacji drogi jaką pakiety podążają w naszej sieci w przypadku relacji niesąsiadujących ze sobą miast. Poleceniem `h10 ping h4` spróbowaliśmy połączyć się z Grazu (h10) do Hanoweru (h4).

```
15 packets transmitted, 15 received, 0% packet loss, time 14016ms
rtt min/avg/max/mdev = 18.108/19.967/28.552/2.845 ms
```

Przed rozpoczęciem analizy wyników pierwszego testu należy wziąć pod uwagę, że czas RTT jest czasem pokonywania przez pakiety drogi między hostami "tam i z powrotem" (od hosta źródłowego do docelowego i znów do źródłowego). Na potrzeby analizy czasu RTT ręcznie odliczaliśmy czas transmisji pierwszych 2 próbek, ponieważ widocznie zawyżały one średni czas. Teoretyczna wartość opóźnienia w

jedną stronę, obliczona jako suma opóźnień na poszczególnych łączach, w przypadku połączenia Graz - Hanower wynosi $8,5ms$, więc przewidywany czas RTT to przynajmniej $17ms$ (do sumy opóźnień na łączach dochodzą również inne, niewielkie opóźnienia związane z przemieszczaniem się pakietów). Średni czas RTT w wyniku przeprowadzonego testu wyniósł $18,9ms$. Wynik ten jest o ok. 11% wyższy niż przewidywany. Tę samą procedurę przeprowadziliśmy dla kilku innych par miast, w celu zauważenia analogii. W przypadku testowanych relacji zbadaliśmy również średnią zmienność opóźnienia. Wyniki obliczeń i testów czasu RTT oraz jitteru zapisaliśmy w tabeli 1. Podczas testów nie odnotowano strat pakietów.

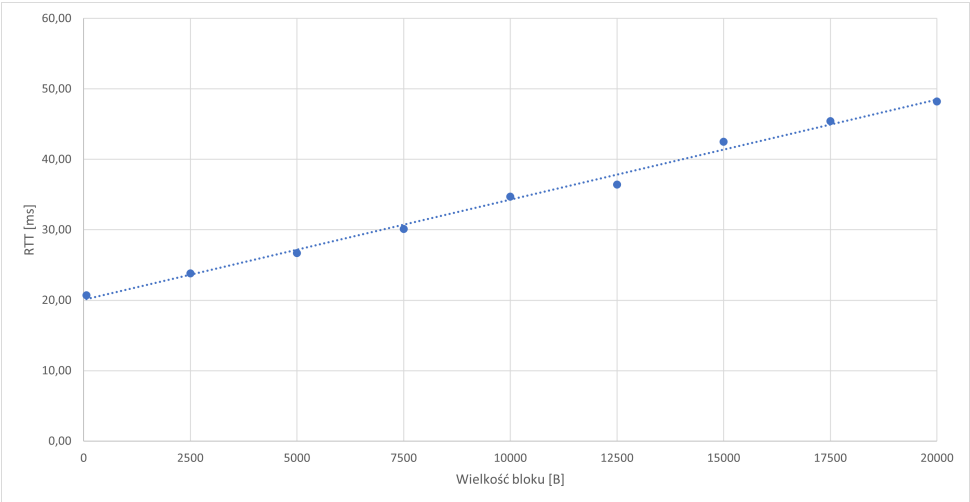
Relacja	Teoretyczne RTT [ms]	Zbadane RTT [ms]	Różnica	Jitter [ms]
Graz - Hanower	17,0	18,9	11%	0,5
Hanower - Malmo	8,3	9,7	17%	0,6
Gdańsk - Norymberga	13,2	15,4	17%	0,9
Malmo - Ostrawa	15,5	17,6	14%	0,7
Berlin - Zagrzeb	11,4	13,1	15%	0,6

Tab. 1. Wyniki obliczeń oraz testów czasu RTT oraz średniego jitteru dla poszczególnych relacji

Proporcjonalnie wyższe, w stosunku do teoretycznych, empirycznie wyznaczone wartości czasu RTT możemy zaobserwować w przypadku każdej relacji między dwoma miastami. Pozwala nam to stwierdzić, że pakiety zgodnie z oczekiwaniami podążają jedyną najkrótszą dla nich drogą łączy między miastami.

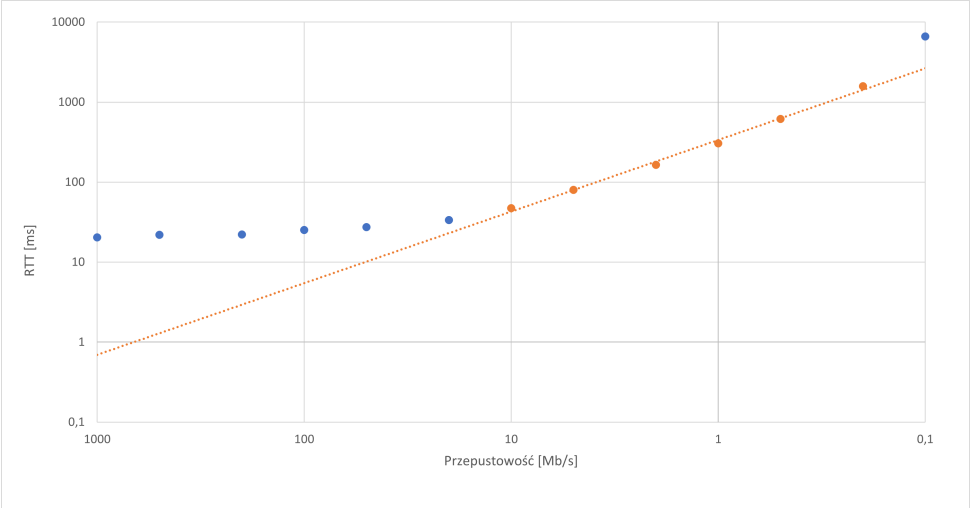
Dla dodatkowego potwierdzenia naszej tezy zmanipulowaliśmy siecią w taki sposób, aby na łączach Hanower - Berlin, Berlin - Praga, Praga - Norymberga opóźnienie wynosiło $1ms$, a na wszystkich pozostałych łączach $1000ms$. Pingując Norynbergę z Hanoweru średni czas RTT wyniósł $7,3ms$, co jest w przybliżeniu równe teoretycznemu czasowi RTT obliczonemu na podstawie jedynej najkrótszej drogi. Pakiety zatem na pewno podążały tą właśnie drogą, w przeciwnym wypadku otrzymany empirycznie czas RTT byłby zdecydowanie większy ze względu na znacznie zwiększone opóźnienia na pozostałych łączach.

Następnie postanowiliśmy sprawdzić jak na **czas RTT** wpłynie **wielkość bloku danych**. W tym celu, przeprowadziliśmy testy dla relacji Malmo - Graz. Przed ich rozpoczęciem, zdecydowaliśmy się zmienić przepustowość łączy na $10\frac{Mb}{s}$, ponieważ zauważyliśmy, że przy takiej wartości przepustowości zależność czasu RTT od wielkości przesyłanych danych jest łatwiejsza do zaobserwowania. Manipulowaliśmy parametrem `-s` polecenia `ping`, wyrażającym wielkość przesyłanego bloku danych w bajtach. Pierwszy test wykonaliśmy przy domyślnej wartości parametru, pozostałe natomiast - zwiększając go o $2500B$. Obserwowaliśmy również średni jitter. Oscylował on w granicach $0,5ms - 1,0ms$ niezależnie od wielkości danych. Podczas żadnego z testów nie zanotowano strat pakietów. Na podstawie testów sporządziliśmy wykres (2) zależności czasu RTT od wielkości bloku danych, ustawionej jako parametr `-s` polecenia `ping`. Na wykres naniesiono prostą najlepszego dopasowania metodą najmniejszych kwadratów. Na podstawie wykresu możemy stwierdzić, że zależność ta jest w przybliżeniu liniowa.



Rys. 2. Zależność czasu RTT od wielkości bloku dla relacji Malmo - Graz

Postanowiliśmy zbadać także **zależność czasu RTT od przepustowości łączy**. Wykonywaliśmy polecenie `ping` z blokiem danych o wielkości 20kB z Gdańska do Ostrawy, za każdym razem zmniejszając przepustowość łączy. W przypadku pierwszych kilku testów (dla przepustowości z zakresu od $1000\frac{Mb}{s}$ do $20\frac{Mb}{s}$) czas RTT był w przybliżeniu stały. Dla przepustowości niższych od $20\frac{Mb}{s}$ czas RTT zaczął wzrastać. Na podstawie zebranych danych, sporządziliśmy wykres (3) zależności czasu RTT od przepustowości łączy, ze skalą logarytmiczną.



Rys. 3. Zależność czasu RTT od przepustowości dla relacji Gdańsk - Ostrawa

Jak wynika z wykresu, czas RTT jest w przybliżeniu odwrotnie proporcjonalny do przepustowości (dla przepustowości z zakresu od $10\frac{Mb}{s}$ do $0,2\frac{Mb}{s}$). Oznacza to, że dla tych przepustowości, przy dwukrotnym jej zmniejszeniu, czas RTT wzrasta dwukrotnie. Dowodem tego stwierdzenia jest prosta najlepszego dopasowania, naniesiona na wykres dla punktów pomiarowych ze wspomnianego zakresu przepustowości. Przy przepustowości $0,1\frac{Mb}{s}$ odnotowano już widocznie większy wzrost czasu RTT i stratę pakietów na poziomie 13%. Przy przepustowości jeszcze dwukrotnie mniejszej większość pakietów została już stracona. Jitter był w przybliżeniu stały i wahał się w granicach $0,5ms - 1,1ms$. Drastycznie wzrósł dopiero przy przepustowości $0,1\frac{Mb}{s}$, aż do $759ms$.

Powyższy przykład świetnie ilustruje zjawisko przeciążania sieci prowadzące do zwiększania czasu RTT oraz przy pewnej krytycznej przepustowości - strat pakietów. Przy małej względem przesyłanego rozmiaru danych przepustowości sieć traci swoją wydajność.

2.3. Zadanie 3

W ramach tego zadania uruchomiliśmy serwer na hoście h1. Na początku sprawdziliśmy adres IP hosta poleceniem `h1 ifconfig`, aby możliwe było późniejsze połączenie się z nim (adres to 10.0.0.1). Serwer uruchomiliśmy poleceniem `h1 python3 -m http.server 80 &`. Nasłuchiwał on na porcie 80. Następnie spróbowaliśmy połączyć się z nim z hosta h2. Użyliśmy polecenia `h2 wget http://10.0.0.1:80`. Host h2 otrzymał plik *index.html* zawierający wszystkie pliki dostępne na tym hoście (czyli te na maszynie wirtualnej Mininet). Gdy utworzymy plik na hoście, możemy uzyskać go dodając do polecenia jego nazwę, np. `h10 wget http://10.0.0.1:80/example.txt`.

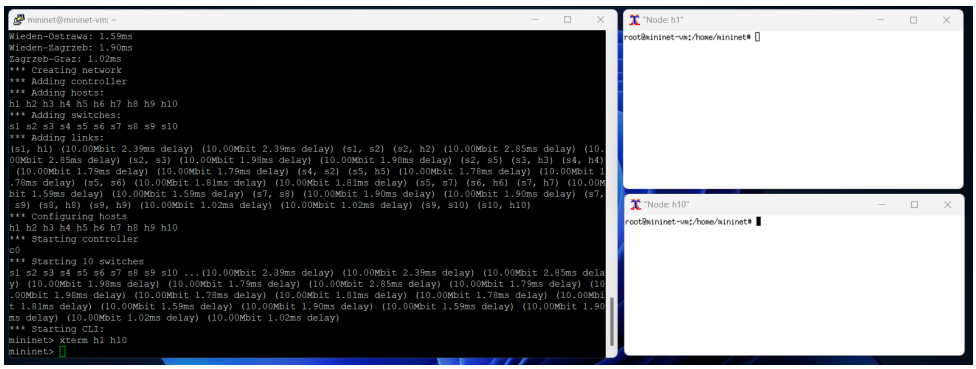
```
--2023-10-20 12:29:36--  http://10.0.0.1/example.txt
Connecting to 10.0.0.1:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 17 [text/plain]
Saving to: 'example.txt.1'

example.txt.1        100%[===================>]      17  --.-KB/s    in 0s

2023-10-20 12:29:36 (1.71 MB/s) - 'example.txt.1' saved [17/17]
```

Skonfigurowaliśmy również dostęp do maszyny wirtualnej za pomocą terminala graficznego. Wszystkie testy przeprowadzaliśmy z terminala Windows przy użyciu polecenia `ssh`. Udało nam się także połączyć

z maszyną wirtualną wykorzystując narzędzie Putty, dzięki czemu mogliśmy uruchamiać oddzielne terminale dla każdego hosta przy użyciu polecenia `xterm`.



Rys. 4. Uruchomione graficzne terminale dla dwóch hostów

2.4. Wnioski

Przeprowadzone w ramach części pierwszej testy sieci pozwoliły nam ustalić związki wartości parametrów przesyłu danych z charakterystykami łączy. Mimo stosowania jedynie polecenia `ping` i manipulacji parametrami, udało nam się potwierdzić jakimi drogami przesyłane są pakiety w przypadku poszczególnych relacji. Ogólnym wnioskiem ze wszystkich wykonanych testów jest fakt, iż parametry przesyłu danych mają duży wpływ na poprawność i jakość działania sieci. Ważne jest, aby projektować i zarządzać siecią w taki sposób, aby zapewnić odpowiednią przepustowość łącz, dostosowaną do rozmiarów przesyłanych w niej danych. Pozwoli to unikać nadmiernego obciążania sieci.

3. Część 2

W drugiej części ćwiczenia laboratoryjnego, w bardziej rozbudowany sposób badaliśmy zależność jakości przesyłu danych od charakterystyk ruchu i sieci. Poza emulatorem sieci Mininet wykorzystaliśmy w tym celu **generator ruchu Iperf**, który pozwolił nam na generowanie strumieni danych **TCP i UDP**. Do realizacji poszczególnych testów przygotowaliśmy skrypty z poleceniami i odpowiednio ustawianymi parametrami. Przy projektowaniu testów wzięliśmy pod uwagę charakterystykę obu wspomnianych protokołów. Jako, iż obserwacja jakości obsługi ruchu w naszej sieci, przy przepustowości łączy na poziomie domyślnej przepustowości z pierwszej części ćwiczenia byłaby trudna, na czas wykonywania testów w ramach części drugiej, postanowiliśmy ustawić przepustowość łączy w naszej sieci na $\mathbf{10}\frac{\mathbf{Mb}}{\mathbf{s}}$.

3.1. Zadanie 4

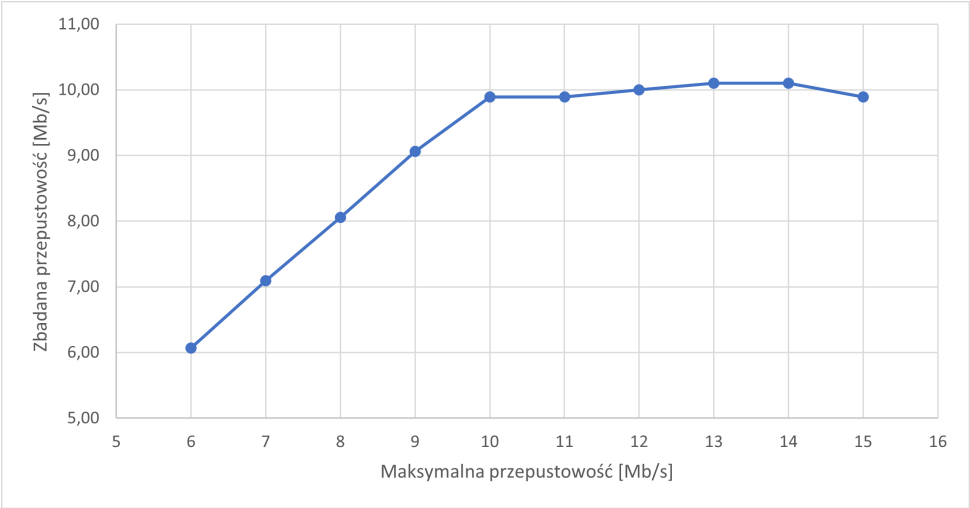
W ramach zadania czwartego, analizowaliśmy wpływ parametrów **pojedynczego strumienia danych** i charakterystyki łączy na jakość przesyłu danych. Poszczególne testy wykonaliśmy na parze reprezentatywnych relacji: Malmo - Graz (h1 - h10) i Hanower - Norymberga (h4 - h6). W celu zachowania zwięzłości sprawozdania, umieściliśmy w nim szczegółowe wyniki testów przeprowadzonych jedynie dla relacji Malmo - Graz.

3.1.1. TCP

W pierwszej kolejności przeprowadziliśmy testy z wykorzystaniem pojedynczych strumieni TCP. Na hostach nasłuchujących uruchamialiśmy serwery poleceniem: `iperf -e -i 1 -s`, a na hostach nadających - klientów poleceniem: `iperf -e -i 1 -c <adress> -N -S 0x08 -n <volume> -b <max_bit_rate> -l <buffer_size> -w <window>`. Badając zależność pomiędzy konkretnym parametrem, a jakością przesyłu, nie podawaliśmy w poleceniu pozostałych parametrów, aby pozostały one na domyślnych wartościach. Podczas każdego testu, pomiędzy hostami przesyłaliśmy dane o wielkości 10MB.

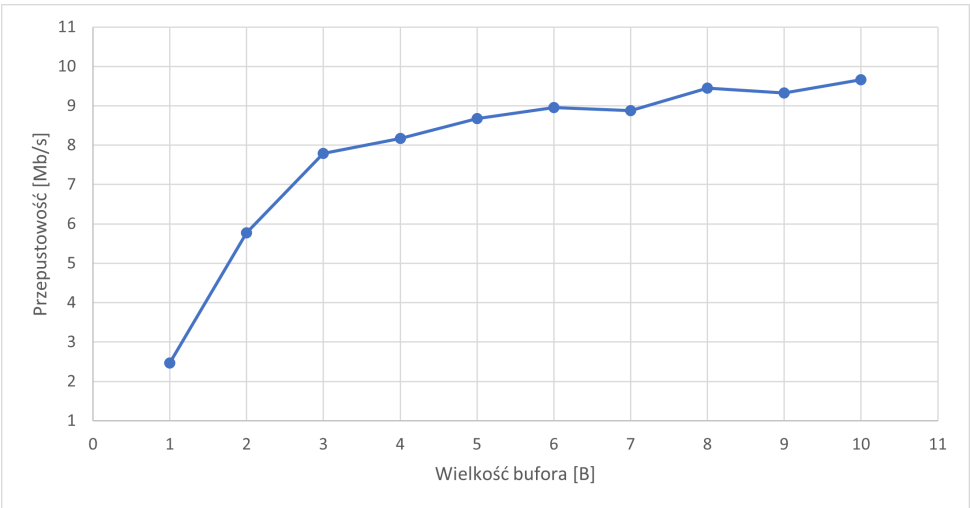
Parametrem, którym manipulowaliśmy jako pierwszym był **parametr -b**, odpowiedzialny za **maksymalną przepustowość**. Zgodnie z oczekiwaniami, ograniczanie przepustowości do wartości mniejszej od bw ustawionego na łączach sprawiało, że zbadana przepustowość nie przekraczała tej zdefiniowanej parametrem `-b` i była jej w przybliżeniu równa. Natomiast przy parametrze ustawionym na wartość

większą niż przepustowość łączy, uzyskiwana prędkość przesyłu nie przekraczała wartości przepustowości na łączach. Nasze testy pokazały, że w takiej sytuacji badana prędkość przesyłu stabilizuje się na poziomie ustawionej przepustowości łączy, mimo zwiększania parametru `-b`. Opisana zależność została przedstawiona na wykresie 5.



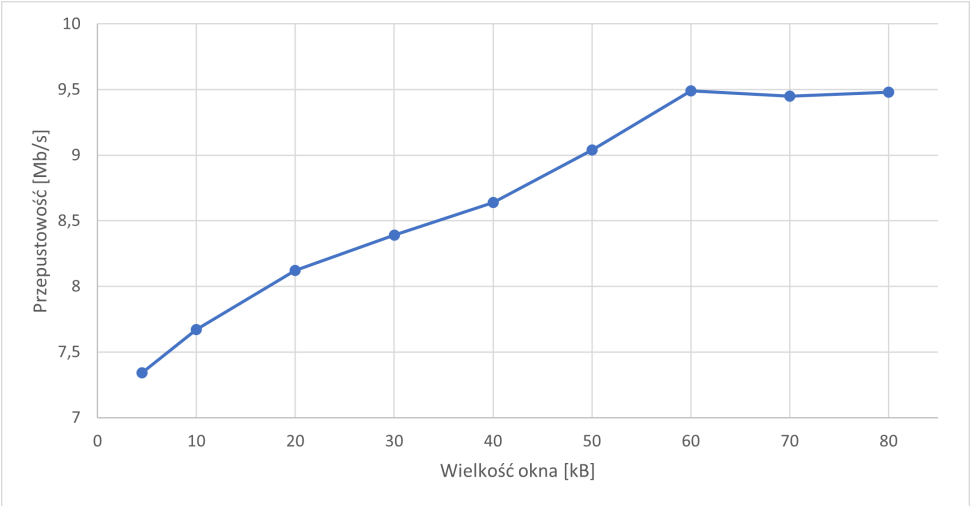
Rys. 5. Zależność zbadanej przepustowości od wartości ręcznie ograniczonej przepustowości dla relacji Malmo - Graz

Drugim parametrem, którym manipulowaliśmy był **parametr -l**, odpowiedzialny za **wielkość bufora**. Przeprowadzone testy pokazały, że bardzo małe wielkości bufora znacznie ograniczają przepustowość. Zwiększając jego wielkość co $1B$ zauważaliśmy dynamiczny wzrost uzyskiwanej przepustowości. Dla wielkości bufora większych niż $8B$, badana przepustowość ustabilizowała się na poziomie przepustowości ustalonej na łączach, co doskonale widać na wykresie 6.



Rys. 6. Zależność przepustowości od wielkości bufora dla relacji Malmo - Graz

Kolejnym badanym parametrem był **parametr -w**, odpowiedzialny za **wielkość okna**. Pokazaliśmy doświadczalnie, że rozmiar okna mniejszy niż optymalny, skutkuje ograniczeniem przepustowości. Stopniowo zwiększając wielkość okna, uzyskiwana przepustowość wzrastała, aż do momentu ustabilizowania się na poziomie bliskim maksymalnemu. W tym przypadku, przy domyślnych wartościach pozostałych parametrów, optymalna wielkość okna wynosiła ok. $60kB$. Dla wielkości okna większych niż optymalne zaobserwowano wzrost czasu RTT. Wyniki z obserwacji zostały zebrane i przedstawione na wykresie 7.



Rys. 7. Zależność przepustowości od wielkości okna dla relacji Malmo - Graz

Takie same testy przeprowadziliśmy dla relacji Hanower - Norymberga. Obserwacje dla tej relacji okazały się analogiczne. Badane zależności zachowywały się w bardzo zbliżony sposób (zgodny z powyższymi opisami oraz wykresami) i potwierdziły nasze przypuszczenia. Wnioski z tej części zadania czwartego zostały podsumowane w sekcji 3.1.3.

3.1.2. UDP

W ramach zadania czwartego, przeprowadziliśmy także testy z wykorzystaniem pojedynczych strumieni UDP. Na hostach nasłuchujących uruchomialiśmy serwery poleceniem: `iperf -e -i 1 -s -u`, a na hostach nadających - klientów poleceniem: `iperf -e -i 1 -c <adress> -u -S 0x10 -t <duration> -b <block_rate>pps -l <block_size> -w <buffer>`.

Na początku zaobserwowaliśmy, że na prędkość przesyłu danych bezpośrednio wpływają **parametr -b (liczba bloków wysyłanych na sekundę)** i **parametr -l (wielkość bloku)**. Jest ona równa w przybliżeniu iloczynowi obu tych parametrów (dochodzą również wielkości nagłówków przesyłanych bloków). Dla następujących kombinacji tych parametrów: $1024pps$ i $256B$, $512pps$ i $512B$, $256pps$ i $1024B$ wyniki były w przybliżeniu równe. W związku z tym, podjęliśmy się manipulacji tylko jednym z tych parametrów. Testy wykonywaliśmy dla $1024pps$ i zmienialiśmy wielkość bloku. Wartości parametru `-l` dobierano tak, aby przetestować sytuacje, w których prędkość przesyłu będzie mniejsza, równa i większa od przepustowości łączy. Z logów hosta wysyłającego odczytywaliśmy prędkość przesyłu danych, a z logów hosta odbierającego: parametr jitter, straty pakietów i średnie opóźnienie (latency). Wyniki testów przedstawiliśmy w tabeli 2.

Wielkość bloku [B]	Prędkość przesyłu danych [Mb/s]	Jitter [ms]	Strata pakietów [%]	Średnie opóźnienie [ms]
1024	8,39	0,442	0	12,19
1280	10,5	0,378	1,5	656,37
1536	12,6	0,476	46	789,12

Tab. 2. Zbadane z wykorzystaniem pojedynczego strumienia UDP parametry jakości obsługi ruchu dla relacji Malmo - Graz

Możemy zaobserwować, że gdy prędkość przesyłu danych jest mniejsza od przepustowości sieci, to jakość transmisji jest bardzo dobra (brak strat pakietów, niewielkie opóźnienie). W sytuacji, gdy jest w przybliżeniu równa tej przepustowości, znacznie wzrasta opóźnienie i pojawiają się małe straty pakietów. Natomiast, gdy znacznie przekracza ona przepustowość, opóźnienie praktycznie już nie wzrasta, ale dochodzi do strat znacznej części pakietów.

Następnie sprawdzaliśmy wpływ wielkości bufora (parametr `-w`) na jakość transmisji i niezależnie od podanej wartości, nie zauważyliśmy żadnych znaczących różnic w logach.

Podobnie jak w przypadku testów z protokołem TCP, takie same testy przeprowadziliśmy dla relacji Hanower - Norymberga. Obserwacje i zależności uzyskane w wyniku badania tej relacji również okazały

się analogiczne do powyższych. Wnioski z tej części zadania czwartego także zostały podsumowane w sekcji 3.1.3.

3.1.3. Wnioski

Poniżej zostały podsumowane wnioski, wyciągnięte na podstawie badań przeprowadzonych za pomocą generowanego pojedynczego strumienia danych:

TCP

— ograniczenie przepustowości wartością mniejszą niż przepustowość łączy skutecznie ogranicza przepustowość transmisji, w przeciwnym wypadku górną granicą jest przepustowość całej sieci i sesja wykorzystuje ją w pełni
— zbyt mała wielkość bufora skutkuje znacznym zmniejszeniem przepustowości
— zbyt małe okno (względem optymalnego cwnd) również powoduje ograniczenie przepustowości, okna większe od optymalnego nie obniżają przepustowości lecz zwiększą czas RTT

UDP

— prędkość przesyłu danych zależy od iloczynu liczby bloków wysyłanych na sekundę i wielkości bloku
— UDP nie ma mechanizmów kontroli przepustowości i wysyła dane z ustaloną prędkością, gdy przepustowość sieci jest mniejsza od tej prędkości to zwiększa się opóźnienie i tracone są pakiety
— nie zaobserwowano wpływu wielkości bufora na jakość przesyłu danych

3.2. Zadanie 5

W ramach zadania piątego, analizowaliśmy wpływ parametrów **kilku strumieni danych** (przesyłanych w ramach kilku realizowanych jednocześnie sesji transportowych) oraz charakterystyki łączy na jakość przesyłu danych. Większość testów wykonaliśmy na parze relacji: Hanower - Norymberga (h4 - h6) i Gdańsk - Wiedeń (h3 - h7), aby możliwa była obserwacja sytuacji, w której jedno z łączy musi być współdzielone w osobnych sesjach transportowych. Skorzystaliśmy z tych samych poleceń co w zadaniu czwartym, jednakże nasłuchujące serwery i klientów wysyłających dane uruchamialiśmy na więcej niż jednym hoście. Przepustowość łączy dla większości testów została ustawiona na $\mathbf{10}\frac{\mathbf{Mb}}{\mathbf{s}}$ (przy opisach testów ze zmienioną przepustowością łączy pojawi się informacja o takim fakcie).

3.2.1. TCP

W tym zadaniu, podczas każdego testu dla protokołu TCP pomiędzy hostami przesyłaliśmy dane o wielkości 30MB. Umożliwiło to uzyskanie przejrzystych wyników.

W pierwszej fazie eksperymentów jednocześnie przesyłaliśmy po jednym strumieniu danych, w obydwu wspomnianych we wstępie relacjach. Badaliśmy rywalizację strumieni o zasoby na wspólnym łączu. W tym celu, określiliśmy trzy przypadki eksperymentalne - ograniczenie przepustowości obu transmisji tak, aby suma tych przepustowości była kolejno: mniejsza, równa i większa od przepustowości ustawionej na łączach. Ograniczeń tych dokonaliśmy z wykorzystaniem parametru `-b`. Wyniki zostały przedstawione w tabeli 3. Na podstawie wyników można domniemywać, że protokół w sposób sprawiedliwy rozdzielił przepustowość pomiędzy strumienie obu sesji.

Maks. przepustowość[Mb/s]	Zbadana przepustowość [Mb/s]	Czas przesyłu [ms]	Retries
3 / 3	3,01 / 2,74	83,73 / 91,94	475 / 498
5 / 5	4,91 / 4,75	51,24 / 52,99	379 / 331
7 / 7	6,19 / 4,98	40,64 / 50,56	291 / 338

Tab. 3. Zbadane z wykorzystaniem dwóch strumieni TCP parametry jakości obsługi ruchu dla relacji Hanower - Norymberga i Gdańsk - Wiedeń (<wynik dla Hanower - Norymberga> / <wynik dla Gdańsk - Wiedeń>)

Podobne testy powtórzyliśmy dodając trzecią sesję transportową - dla relacji Malmo - Praga (h1 - h5). W tym przypadku prędkość przesyłu danych również została przydzielona sprawiedliwie, w ramach przepustowości łącza.

W drugiej części eksperymentów badaliśmy wpływ charakterystyki łączy na jakość przesyłu danych, dla dwóch sesji transportowych. W kodzie definiującym topologię sieci zmniejszyliśmy do $3\frac{Mb}{s}$ przepustowość łącza Hanower - Berlin, a w drugim przypadku Praga - Norymberga. Dzięki temu, w pierwszym przypadku jeden ze strumieni przed dotarciem do współdzielonego z drugim strumieniem łącza Berlin - Praga, podążał łączem o znacznie ograniczonej przepustowości. W drugim natomiast, taka sytuacja następowała na łączu za tym współdzielonym. Wyniki z tego eksperymentu zostały zawarte w tabeli 4.

Przep. ograniczona na łączu:	Zbadana przepustowość [Mb/s]	Czas przesyłu [ms]	Retries
przed wspólnym łączem	4,19 / 9,04	60,05 / 27,82	104 / 240
za wspólnym łączem	4,02 / 8,29	62,58 / 30,34	241 / 183

Tab. 4. Zbadane z wykorzystaniem dwóch strumieni TCP, przy zmniejszeniu przepustowości na jednym łączu, parametry jakości obsługi ruchu dla relacji Hanower - Norymberga i Gdańsk - Wiedeń (<wynik dla Hanower - Norymberga> / <wynik dla Gdańsk - Wiedeń>)

Możemy zauważyć, że nie ma znaczenia czy dla jednej sesji przepustowość zostanie ograniczona na łączu przed tym współdzielonym przez obie sesje, czy za tym łączem. Jest to cecha charakterystyczna dla działania protokołu TCP. Niezależnie, gdzie na drodze przesyłu danych to łącze się znajduje, to w ramach tej sesji prędkość przesyłu będzie ograniczona przepustowością tego łącza.

3.2.2. UDP

Testy zaczęliśmy od dwóch sesji transportowych dla relacji wspomnianych we wstępie, o równych parametrach. Parametr `-b` ustawiliśmy na 1024*pps* i manipulowaliśmy tylko wielkością bloku. Na początku przeprowadziliśmy testy przy równych wielkościach bloku, aby sesje nadawały z równą prędkością. Analizowaliśmy wtedy jakość przesyłu w sytuacjach, gdy suma prędkości jest mniejsza, równa i większa od przepustowości sieci. Następnie zbadaliśmy przypadki przy różnych wielkościach bloku, a co za tym idzie - różnych prędkościach przesyłu. W takich sytuacjach również analizowaliśmy jakość przesyłu danych, gdy wspomniana suma prędkości jest mniejsza i większa niż przepustowość sieci. Uzyskane wyniki zebrano w tabeli 5.

Wielkości bloku [B]	Prędkość przesyłu danych [Mb/s]	Suma [Mb/s]	jitter [ms]	Strata [%]	Opóźnienie [ms]
512 / 512	4,19 / 4,19	8,38	0,496 / 1,864	0 / 0	7,9 / 9,0
640 / 640	5,24 / 5,24	10,48	0,318 / 0,681	8,1 / 6,8	465 / 468
768 / 768	6,29 / 6,29	12,58	0,370 / 1,609	24 / 19	600 / 605
512 / 256	4,19 / 2,10	6,29	0,421 / 0,878	0 / 0	7,8 / 8,8
1024 / 512	8,39 / 4,19	12,58	0,421 / 1,060	22 / 20	599 / 602

Tab. 5. Zbadane z wykorzystaniem dwóch strumieni UDP parametry jakości obsługi ruchu dla relacji Hanower - Norymberga i Gdańsk - Wiedeń (<wynik dla Hanower - Norymberga> / <wynik dla Gdańsk - Wiedeń>)

Możemy zaobserwować, że dla przypadków gdy suma prędkości była większa niż przepustowość sieci, obie sesje traciły w przybliżeniu taki sam procent pakietów i ich opóźnienia były równe, nawet jeżeli prędkość przesyłu danych była różna. Dzieje się tak, ponieważ sesji o większej wielkości bloku, naturalnie przypada większe prawdopodobieństwo utraty pakietów (jest ich przesyłanych więcej). Straconych zatem jest w niej więcej pakietów, ale udział procentowy pakietów straconych w stosunku do wszystkich przesyłanych w niej pakietów jest zbliżony.

Następnie przeprowadziliśmy testy dla tych samych sesji, przy równych wielkościach bloku i liczbie pakietów na sekundę, a co za tym idzie prędkości przesyłu. W jednym teście, identycznie jak w przypadku protokołu TCP w sekcji 3.2.1, ograniczyliśmy dla jednej z tych sesji przepustowość na połączeniu przed współdzielonym przez obie sesje łączem, a w drugim na połączeniu za współdzielonym łączem. Wyniki widoczne są w tabeli 6.

Przep. ograniczona na łączu:	Prędkość przesyłu danych [Mb/s]	jitter [ms]	Strata [%]	Opóźnienie [ms]
przed wspólnym łączem	6,29 / 6,29	0,454 / 0,395	63 / 0	2837 / 8,5
za wspólnym łączem	6,29 / 6,29	0,731 / 0,412	62 / 18	3326 / 605

Tab. 6. Zbadane z wykorzystaniem dwóch strumieni UDP, przy zmniejszeniu przepustowości na jednym łączu, parametry jakości obsługi ruchu dla relacji Hanower - Norymberga i Gdańsk - Wiedeń (<wynik dla Hanower - Norymberga> / <wynik dla Gdańsk - Wiedeń>)

Możemy zaobserwować, że w przypadku dwóch sesji UDP (w przeciwieństwie do dwóch sesji TCP) ma znaczenie, czy ograniczymy przepustowość łącza przed współdzielonym łączem czy za nim. Gdy ograniczone łącze było przed, jedna z tych sesji straciła wystarczająco pakietów, aby oba strumienie zmieściły się na wspólnym łączu i nie wpłynęło to negatywnie na jakość transmisji drugiej sesji. Natomiast, gdy

łącze ograniczone było za łączem współdzielonym, obie sesje konkurowały o zasoby na współdzielonym łączu i obie straciły część pakietów. Następnie ta z łączem o ograniczonej przepustowości straciła na nim resztę swoich pakietów.

3.2.3. TCP i UDP

Ostatnim eksperymentem w ramach laboratorium było badanie jakości przesyłu danych przy jednej sesji TCP i jednej sesji UDP, dla wspomnianych we wstępie relacji. We wszystkich testach w ramach tego eksperymentu sesja TCP była skonfigurowana tak samo. Przy domyślnych parametrach przesyłaliśmy dane o wielkości $30MB$. W sesji UDP zmienialiśmy wielkość bloku tak, by prędkość przesyłu była odpowiednio mniejsza, równa i większa od przepustowości łączy w naszej sieci. W każdym z testów sesja UDP trwała $30s$. Wyniki testów przedstawiliśmy w tabeli 7.

Wielkość bloku UDP [B]	Prędkość przesyłu danych UDP [Mb/s]	Strata UDP [%]	Przesłane dane w ciągu pierwszych 30s TCP [MB]	Czas przesyłu danych TCP [s]
768	6,29	0,8	15,02	43,58
1024	8,39	1,3	7,56	47,43
1280	10,5	5,8	3,15	49,66
1536	12,6	45,6	2,21	50,45

Tab. 7. Zbadane z wykorzystaniem jednego strumienia TCP (w relacji Gdańsk - Wiedeń) i jednego UDP (w relacji Hanower - Norynberga) parametry jakości obsługi ruchu dla relacji

Zaobserwowaliśmy, że sesja TCP miała trudności z przesyłem danych równolegle z sesją UDP. W przypadku, gdy sesja UDP wysyłała dane z prędkością wyższą niż pozwalała na to przepustowość sieci i sama traciła przy tym pakiety, sesja TCP prawie w ogóle nie mogła wysłać żadnych pakietów i w ciągu całych $30s$ wysłała ich zaledwie $2 - 3$ MB. Jak tylko sesja UDP się zakończyła, sesja TCP była w stanie wykorzystać wszystkie zasoby i wysłać resztę danych bardzo szybko.

Zaobserwowane zjawisko jest zgodne z teoretyczną wiedzą. Sesje TCP próbują dzielić się przepustowością z innymi sesjami, kosztem spowolnienia własnej transmisji. W protokole UDP bloki wysyłane są z daną, ustaloną prędkością i sesje te nie zwracają uwagi na jakiekolwiek inne. Uruchomienie obu sesji powoduje, że sesja TCP ma trudności z przesyłem danych równolegle z sesją UDP.

3.2.4. Wnioski

Wykonując testy w ramach zadania piątego obserwowaliśmy jak od kombinacji sesji, ich typu i parametrów zależy jakość przesyłu danych. Analizowaliśmy jak sesje dzielą się między sobą dostępnymi zasobami w przypadku współdzielenia łączy. Sesje TCP starają się samemu wykorzystać jak najwięcej zasobów sieci, ale gdy tylko dochodzi do strat pakietów, to spowalniają prędkość przesyłu. Powoduje to, że wiele sesji TCP dzieli się dostępnymi zasobami sprawiedliwie. Sesje UDP nie mają mechanizmów kontroli prędkości przesyłu i w przypadku współdzielenia łączy przez wiele takich sesji, wzajemnie sobie przeszkadzają i wszystkie tracą pakiety. Skutkuje to obniżeniem jakości przesyłu danych. Wreszcie w przypadku, gdy łącze jest współdzielone przez sesje różnego typu, sesje UDP potrafią uniemożliwić sesjom TCP jakościowe przesyłanie danych.

4. Podsumowanie

Podsumowując, ćwiczenie laboratoryjne uznaliśmy za bardzo ciekawe, wymagające od nas wykorzystania naszej kreatywności i wyobraźni. Co prawda w ramach ćwiczenia pracowaliśmy wyłącznie z emulatorem sieci i dobierane przez nas parametry nie były zgodne z realiami panującymi w rzeczywistych sieciach. Zbadane zależności pozwoliły jednak wyciągnąć wnioski na temat działania prawdziwych sieci oraz protokołów i w ten sposób pogłębić naszą wiedzę w tym zakresie.

5. Bibliografia

[1] Zee Source. *Zee Maps - Create and publish interactive maps.* URL: https://www.zeemaps.com/ (visited on 10/28/2023).