Programowanie Sieciowe - Z 2

Jan Lewandowski
Szymon Ryszka
Krzysztof Sokół

23 Listopada 2024

1 Treść zadania

Napisz zestaw dwóch programów – klienta i serwera komunikujących się poprzez TCP. Transmitowany strumień danych powinien być stosunkowo duży, nie mniej niż 100 kB.

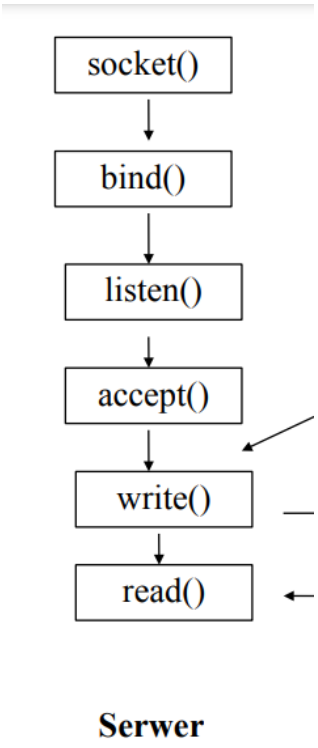
Klient powinien wysyłać do serwera strumień danych w pętli (tzn. danych powinno być „dużo", minimum rzędu kilkuset KB). Serwer powinien odbierać dane, ale między odczytami realizować sztuczne opóźnienie (np. przy pomocy funkcji sleep()). W ten sposób symulujemy zjawisko odbiorcy, który „nie nadąża" za szybkim nadawcą. Stos TCP będzie spowalniał nadawcę, aby uniknąć tracenia danych. Należy zidentyfikować objawy tego zjawiska po stronie klienta (dodając pomiar i logowanie czasu) i krótko przedstawić swoje wnioski poparte uzyskanymi statystykami czasowymi. Wskazane jest też przeprowadzenie eksperymentu z różnymi rozmiarami bufora nadawczego po stronie klienta (np. 100 B, 1 KB, 10 KB)

To zadanie należy wykonać, korzystając z kodu klienta i serwera napisanych w języku C.

2 Rozwiązanie problemu

2.1 Serwer

Serwer korzysta z funkcji z API C widocznych na rysunku 1.



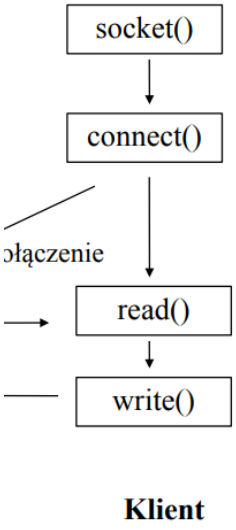
Rysunek 1: Schemat budowy serwera

Zgodnie ze schematem wywołujemy kolejno:

- `socket()` - tworzymy gniazdo sieciowe,
- `bind()` - przypisujemy lokalny adres do gniazda,
- `listen()` - zamieniamy socket niepołączony w socket nasłuchujący - długość kolejki oczekujących ustaliliśmy na 3,
- `accept()` - przyjmujemy wchodzące połączenie sieciowe,
- `read()` - odbieramy i odczytujemy ile bajtów przesłał nam klient.

2.2 Klient

Klient korzysta z funkcji z API C widocznych na rysunku 2.



Rysunek 2: Schemat budowy klienta

Klient działa następująco:

- jako parametr do programu podajemy rozmiar bufora nadawczego,
- bufor zapełniany jest literą A,
- `socket()` - tworzymy gniazdo sieciowe,
- `connect()` - nawiązujemy połączenie z serwerem,
- `send()` - wysyłanie zgodnie z poleceniem w pętli - 20 razy podejmujemy próbę wysyłki danych,
- `difftime()` - mierzymy czasy.

3 Opis konfiguracji testowej

Do testów wykorzystaliśmy rozmiar bufora odbiorczego 10240 bajtów. Zmienialiśmy rozmiar bufora nadawczego dla tych wartości: 100B, 40kB, 60kB, 100kB, 200kB. Adres ip serwera: 172.21.32.3, adres ip klienta: 172.21.32.2, port: 8080.

4 Testowanie

```
ksokol@bigubu:~$ cat result_client.txt
Połączono z serwerem.
Wysłano dane w 0.000000 s
Wysłano dane w 0.000000 s
Wysłano dane w 0.000000 s
Wysłano dane w 0.000000 s
Wysłano dane w 0.000000 s
Wysłano dane w 0.000000 s
Wysłano dane w 0.000000 s
Wysłano dane w 15.000000 s
Wysłano dane w 0.000000 s
Wysłano dane w 20.000000 s
Wysłano dane w 0.000000 s
Wysłano dane w 24.000000 s
Wysłano dane w 0.000000 s
Wysłano dane w 22.000000 s
Wysłano dane w 0.000000 s
Wysłano dane w 0.000000 s
Wysłano dane w 21.000000 s
Wysłano dane w 0.000000 s
Wysłano dane w 23.000000 s
Wysłano dane w 0.000000 s
ksokol@bigubu:~$
```

Rysunek 3: Wysyłanie danych o rozmiarze 100kB po stronie klienta

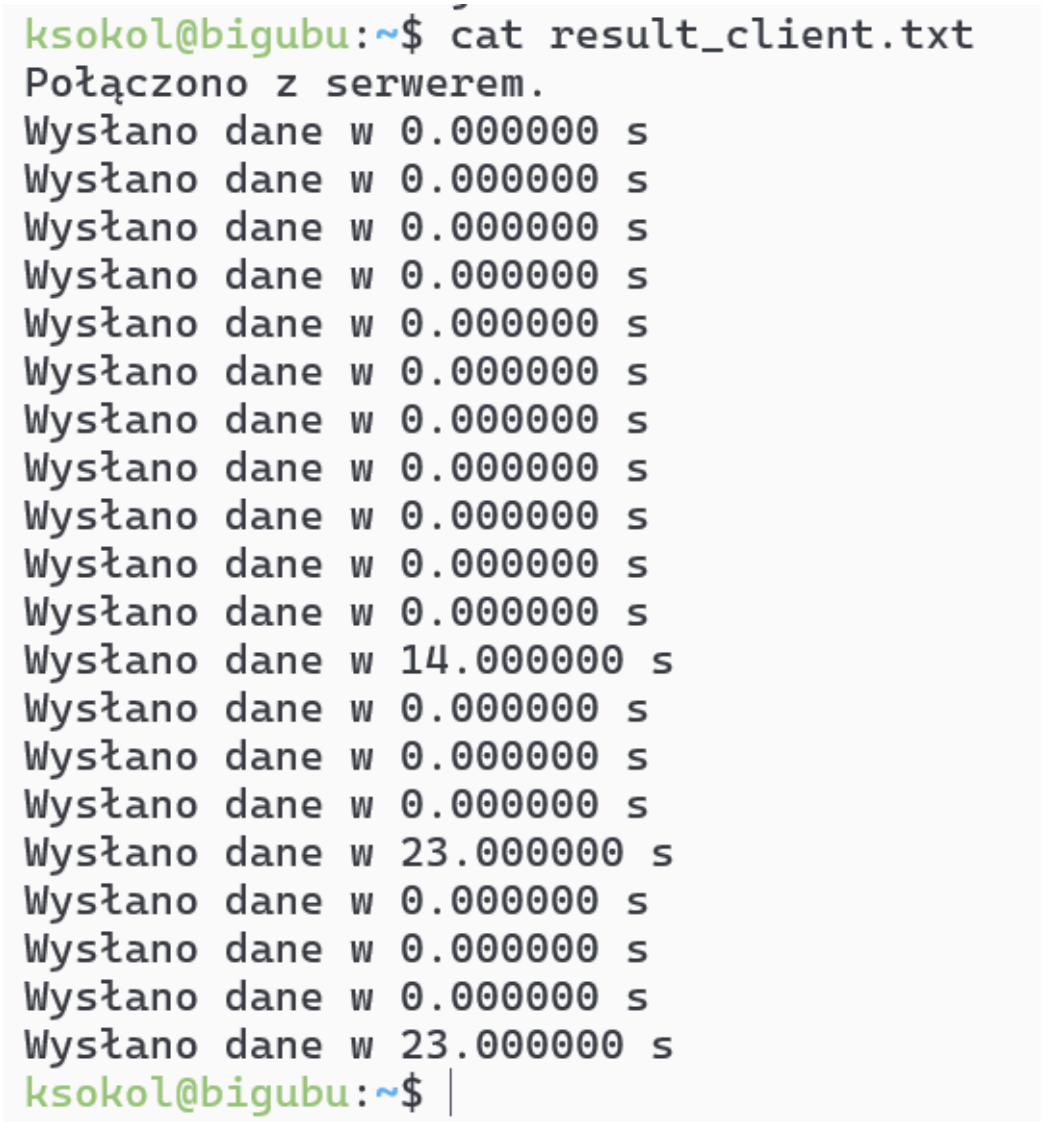
```
ksokol@bigubu:~$ cat result_server.txt
Serwer oczekuje na połączenie...
Połączenie nawiązane.
Odebrano 7240 bajtów.
Odebrano 10240 bajtów.
Odebrano 10240 bajtów.
Odebrano 10240 bajtów.
Odebrano 10240 bajtów.
Odebrano 10240 bajtów.
Odebrano 10240 bajtów.
Odebrano 10240 bajtów.
Odebrano 10240 bajtów.
Odebrano 10240 bajtów.
Odebrano 10240 bajtów.
Odebrano 10240 bajtów.
Odebrano 10240 bajtów.
Odebrano 10240 bajtów.
Odebrano 10240 bajtów.
Odebrano 10240 bajtów.
Odebrano 10240 bajtów.
Odebrano 10240 bajtów.
Odebrano 10240 bajtów.
Odebrano 10240 bajtów.
Odebrano 10240 bajtów.
Odebrano 10240 bajtów.
Odebrano 10240 bajtów.
Odebrano 10240 bajtów.
Odebrano 10240 bajtów.
Odebrano 10240 bajtów.
Odebrano 10240 bajtów.
Odebrano 10240 bajtów.
Odebrano 10240 bajtów.
Odebrano 10240 bajtów.
Odebrano 10240 bajtów.
Odebrano 10240 bajtów.
```

Rysunek 4: Odbieranie danych o rozmiarze 100kB po stronie serwera

Jak widać na powyższych zrzutach ekranu, gdy wysyłane dane są dość spore, to serwer zaczyna po jakimś czasie spowalniać klienta w ich wysyłaniu. Sam odbiera dane bardzo systematycznie, natomiast czasy wysyłania danych po stronie klienta od ósmego pakietu zaczynają się różnić, co jest własnie tego efektem.

```
ksokol@bigubu:~$ cat result_client.txt
Połączono z serwerem.
Wysłano dane w 0.000000 s
Wysłano dane w 0.000000 s
Wysłano dane w 0.000000 s
Wysłano dane w 0.000000 s
Wysłano dane w 0.000000 s
Wysłano dane w 0.000000 s
Wysłano dane w 0.000000 s
Wysłano dane w 0.000000 s
Wysłano dane w 0.000000 s
Wysłano dane w 0.000000 s
Wysłano dane w 0.000000 s
Wysłano dane w 0.000000 s
Wysłano dane w 0.000000 s
Wysłano dane w 0.000000 s
Wysłano dane w 14.000000 s
Wysłano dane w 0.000000 s
Wysłano dane w 0.000000 s
Wysłano dane w 0.000000 s
Wysłano dane w 0.000000 s
Wysłano dane w 0.000000 s
ksokol@bigubu:~$
```

Rysunek 5: Wysyłanie danych o rozmiarze 40kB po stronie klienta



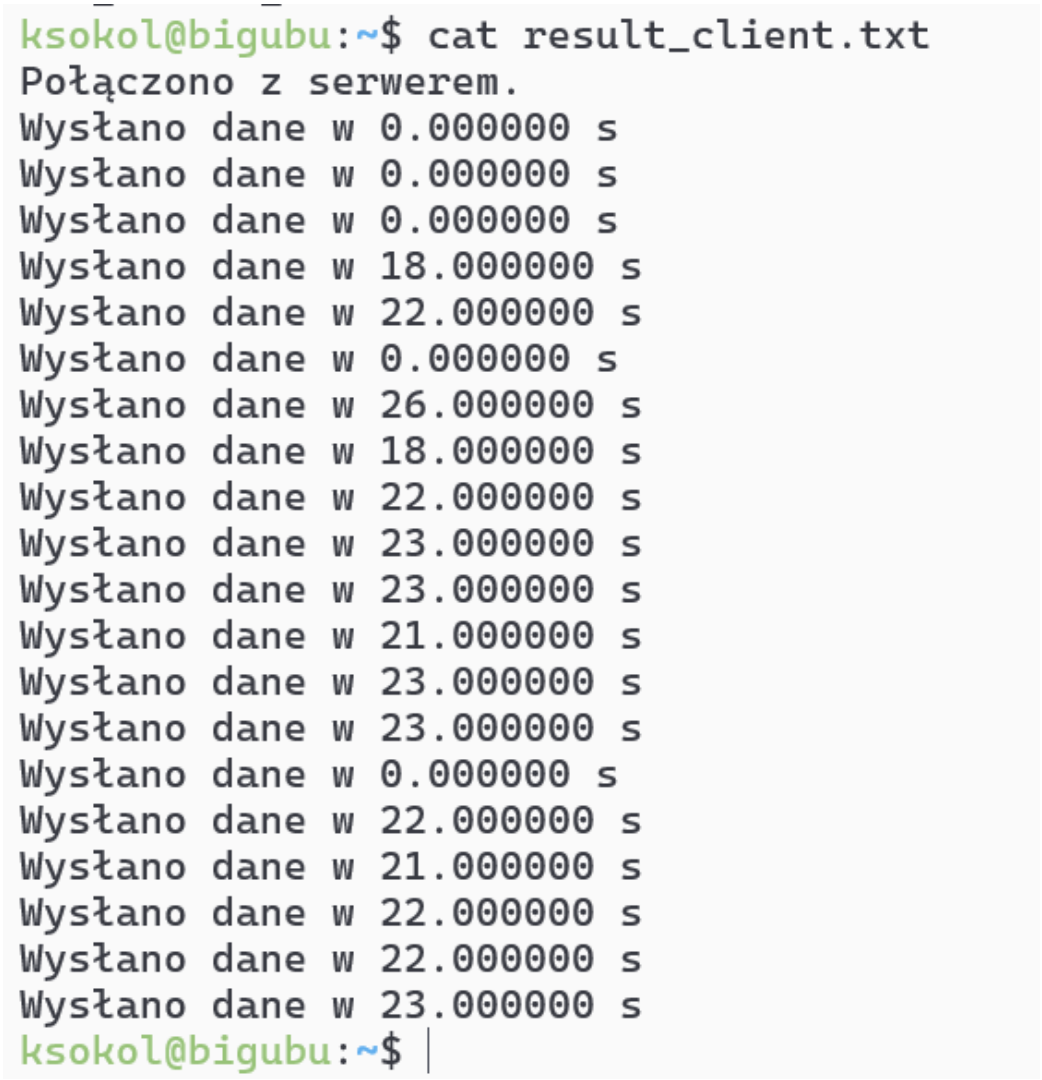
Rysunek 6: Wysyłanie danych o rozmiarze 40kB po stronie klienta

Dwa powyższe zrzuty ekranu są dokumentacją przy próbie wyszukania wielkości, która zaczyna opóźniać wysyłanie danych po stronie klienta. Opóźnianie klienta przez serwer zaczyna się mniej więcej gdy rozmiar danych jest 3-krotnie większy od rozmiaru buforu odbiorczego, a następny skok w długości oczekiwania jest dla 6-krotnie większych danych.

```
ksokol@bigubu:~$ cat result_server.txt
Serwer oczekuje na połączenie...
Połączenie nawiązane.
Odebrano 100 bajtów.
Odebrano 1900 bajtów.
ksokol@bigubu:~$ cat result_client.txt
Połączono z serwerem.
Wysłano dane w 0.000000 s
Wysłano dane w 0.000000 s
Wysłano dane w 0.000000 s
Wysłano dane w 0.000000 s
Wysłano dane w 0.000000 s
Wysłano dane w 0.000000 s
Wysłano dane w 0.000000 s
Wysłano dane w 0.000000 s
Wysłano dane w 0.000000 s
Wysłano dane w 0.000000 s
Wysłano dane w 0.000000 s
Wysłano dane w 0.000000 s
Wysłano dane w 0.000000 s
Wysłano dane w 0.000000 s
Wysłano dane w 0.000000 s
Wysłano dane w 0.000000 s
Wysłano dane w 0.000000 s
Wysłano dane w 0.000000 s
Wysłano dane w 0.000000 s
Wysłano dane w 0.000000 s
ksokol@bigubu:~$
```

Rysunek 7: Wysyłanie danych o rozmiarze 100B po stronie klienta

Bardzo mały rozmiar wysyłanych danych nie ma żadnego wpływu na czas wysyłania danych.



Rysunek 8: Wysyłanie danych o rozmiarze 200kB po stronie klienta

Natomiast im większy rozmiar danych, tym większe opóźnienia występuja w nadawaniu po stronie klienta.

5 Wnioski końcowe

Testowanie programów opartych o działanie protokołu TCP pokazuje, że jest to niezawodny protokół za sprawą m.in. sztucznego opóźnienia. Gdy serwer nie nadąża z odbieraniem pakietów, aby ich nie tracić umyślnie spowalnia wysyłanie pakietów przez klienta. Opóźnienia wystepują dopiero przy kilkukrotnie większym rozmiarze bufora nadawczego względem bufora odbiorczego. Częstsze opóźnienia pojawiają się dopiero przy 10-krotnie większym buforze nadawczym.