Raport z zadania Protokół Przesyłania Ciągów Bajtów

Jacek Kajdan

1 Kod testera w pythonie

```
import time
import sys
import subprocess


def runc(protocol, address, port, input_file):
    count = 0
    totaltime = 0
    for _ in range(20):
        try:
            myinput = open(input_file)
            start = time.time()
            subprocess.check_call(["./ppcbc",
               protocol, address,port],stdin=myinput)

            totaltime += time.time() - start
            count+=1
            myinput.close()
            time.sleep(0.1)
        except subprocess.CalledProcessError:
            myinput.close()
            if (protocol == "udpr"):
                time.sleep(6)
            else:
                time.sleep(2)
    return count, totaltime



address = sys.argv[1]
port = sys.argv[2]
```

```
mode = sys.argv[3]


res = open("wyniki.txt", "a")
prots = ['tcp'] if mode == 'tcp' else ['udp','udpr']
for prot in prots:
    res.write(f"{prot}:\n")
    for file in ["B100.txt", "kb10.txt", "kb100.txt", "mb1.txt"]:
        succ, total_time = runc(prot, address, port, file)
        res.write(f"{file[:-4]}, {total_time}s, {succ}/20\n")
    res.write("\n\n")
res.close()
```

2 Sposób przeprowadzenia testów

Testy składały się z 4 plików o wielkości kolejno: 100 bajtów, 10 kilobajtów, 100 kilobajtów, 1000 kilobajtów, 1 megabajt. Po uruchomienu serwera, włączałem testerkę komendą "python testerka.py ip-address ip-port tcp/udp". Czas podawany niżej to sumaryczny czas 20 uruchomień klientów.

3 Wyniki testów

3.1 Brak opóźnień

tcp:
B100, 0.9104874134063721s, 20/20
kb10, 0.9099271297454834s, 20/20
kb100, 0.9008166790008545s, 20/20
mb1, 1.1751430034637451s, 20/20
udp:
B100, 0.8734889030456543s, 20/20
kb10, 0.8967444896697998s, 20/20
kb100, 0.917102575302124s, 20/20
mb1, 1.1469388008117676s, 20/20
udpr:
B100, 0.9641001224517822s, 20/20
kb10, 0.8624241352081299s, 20/20
kb100, 0.9641544818878174s, 20/20
mb1, 1.3719873428344727s, 20/20

3.2 Brak opóźnień - loopback

tcp:
B100, 0.031734466552734375s, 20/20
kb10, 0.028905630111694336s, 20/20
kb100, 0.04711771011352539s, 20/20
mb1, 0.5815749168395996s, 20/20
udp:
B100, 0.03318333625793457s, 20/20
kb10, 0.03404402732849121s, 20/20
kb100, 0.04480123519897461s, 20/20
mb1, 0.03971123695373535s, 5/20
udpr:
B100, 0.031102418899536133s, 20/20
kb10, 0.03382682800292969s, 20/20
kb100, 0.04779481887817383s, 20/20
mb1, 0.1660153865814209s, 20/20

3.3 Opóźnienie 30ms

tcp:
B100, 3.650810956954956s, 20/20
kb10, 3.6506102085113525s, 20/20
kb100, 4.909391403198242s, 20/20
mb1, 9.917988777160645s, 20/20
udp:
B100, 2.446805953979492s, 20/20
kb10, 2.447436571121216s, 20/20
kb100, 2.4587454795837402s, 20/20
mb1, 0.12878012657165527s, 1/20
udpr:
B100, 2.4458582401275635s, 20/20
kb10, 2.447554588317871s, 20/20
kb100, 3.6669561862945557s, 20/20
mb1, 20.703190803527832s, 20/20

3.4 Packet loss - 25%

tcp:
B100, 11.375205278396606s, 16/20
kb10, 12.343240022659302s, 14/20
kb100, 15.961175203323364s, 13/20
mb1, 17.55463695526123s, 8/20
udp:
B100, 5.1740875244140625s, 4/20
kb10, 0.008618831634521484s, 5/20
kb100, 0.015752315521240234s, 7/20
mb1, 0s, 0/20
udpr:
B100, 35.323508501052856s, 13/20
kb10, 35.70095658302307s, 11/20
kb100, 101.87967252731323s, 11/20
mb1, 0s, 0/20

4 Wnioski

Po testach bez opóźnień widać, że zdecydowana większość pakietów dociera do celu bez problemu. Wyjątkiem jest plik mb1 wysyłany przez udp loopbackiem. Wydaje mi się, że wynika to z tego, że wiadomości przychodzą do serwera za szybko i czytając z bufora, serwer traci część pakietów. W teście z opóźnieniem 30ms nie odnotowałem większych różnic co do programów zakończonych sukcesem, natomiast oczywiście całość działa wolniej. Ostatni test polegający na gubieniu 25% pakietów działa zdecydowanie najwolniej oraz co spodziewane traci na tyle dużo pakietów, że spora część uruchomień klientów kończy się błędami - we wszystkich protokołach i plikach.