Martin Hric
ID: 111696

SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE

FAKULTA INFORMATIKY A INFORMAČNÝCH TECHNOLÓGIÍ

UDP KOMUNIKÁTOR

NÁVRH ZADANIA

Vypracoval: Martin Hric

Cvičiaci: Ing. Matej Janeba

Ak. rok: 2021/2022

Martin Hric
ID: 111696

OBSAH

Martin Hric
ID: 111696

Implementácia

Program bude implementovaný v jazyku *Python 3*, s použitím knižníc *socket, queue, math, os, libscrc, threading.*

Spojenie sa nadviaže medzi mojim PC, ktorý bude vystupovať ako server a laptop ako client. Aj PC aj laptop disponujú operačným sytémom Windows.

Používateľské rozhranie:

Na začiatku stále si vypýtam od užívateľa potrebné údaje, pokiaľ sú chybné, znova vypýtam od začiatku údaje.

Program sa spustí aj keď nevyplní správnu IP a port, bohužiaľ ale komunikácia neprebehne.

Server

Pri spustení programu na PC ako server, užívateľ vyberie port, na ktorom bude server figurovať. Následne čaká na inicializačnú správu. Ak neprichádza, server stále počúva a čaká.

Ak príjde inicializačný fragment, odošle klientovi naspäť správu potvrdzujúcu spojenie.

Server bude kontrolovať prijaté fragment po desiatkach, čiže ak začnú prichádzať dáta, až po 10 fragmentoch ich všetky skontroluje pomocou CRC (Cyclic redundancy check) a skontroluje ak dorazí posledný fragment (ak fragmentov je menej ako 10 alebo počet fragmentov mod 10 != 0)

Keď narazí na nejaký chybný fragment, informuje o tom klienta vypísaním, v ktorej desiatke sa nachádza a taktiež aj jeho poradie, inak pošle správu o úspešne prenesených dátach.

Po obdržaní všetkých fragmentov, ktoré boli úspešné, tieto fragementy budú spojené a uložené ako súbor.

Čiže ARQ je Stop and Wait , po desiatkách , aby to časovo nebolo príliš obtiažne.

Martin Hric
ID: 111696

Klient

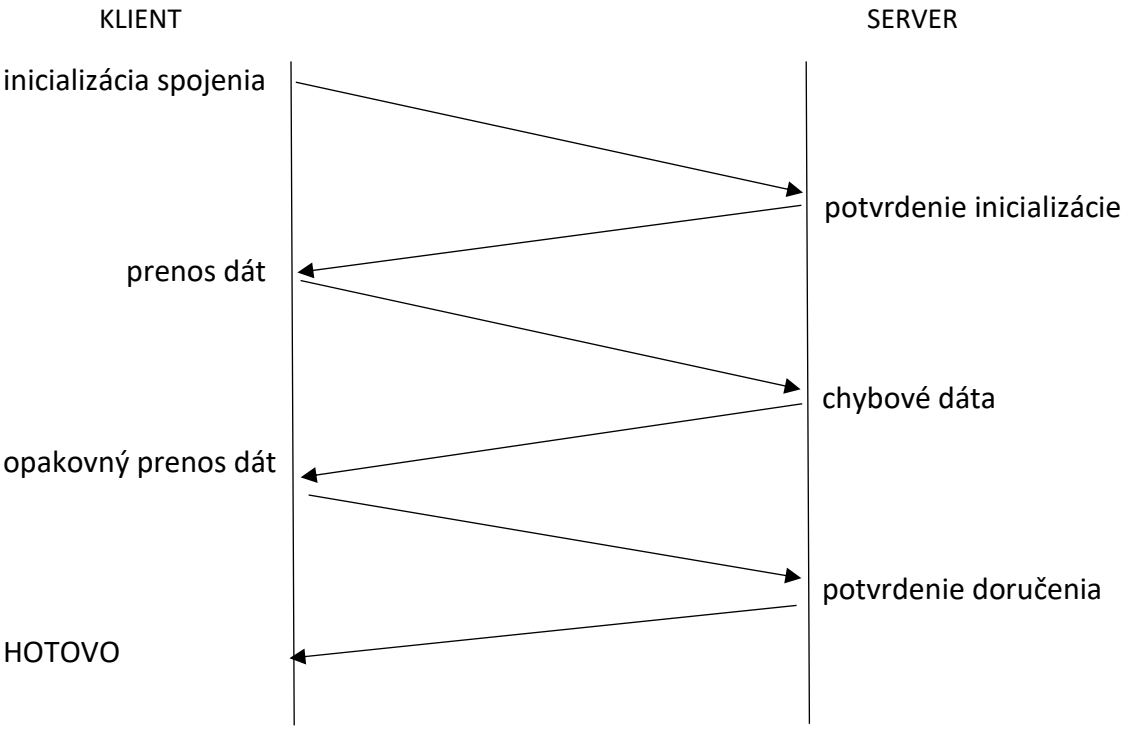
Klient musí ako input vpísať viac dát a to konkrétne IP adresu servera, port , veľkosť fragmentu, druh správy – či je to jednoduchý message alebo súbor, ak je to message tak ďalej požiada o napísanie správy , ak je to súbor tak požiada o cestu k súboru. Taktiež sa opýta, č isa majú dáta simulovane naschvál poškodiť.

Následne pošle inicializačnú správu serveru, po ktorej príde správa zo servera, ktorá nám hovorí o úspešnej inicializácii. Ak nepríde, bude užívateľ vyzvaný na opätovné zadanie IP adresy a portu.

Súbor rozdelí na fragment podľa špecifikovanej veľkosti, ak klient zadal väčšiu veľkosť ako je potrebná, tak sa zmení na takú veľkosť aká je veľkosť dát.

Ak po odoslaní správy alebo súboru klient po vyzvaní od servera neodošle inicializačný fragment serveru, v prípade ak chce poslať ďalšiu správu (v prípade ak nechce tak sa klient vypne) tak sa posiela tzv. “keep_alive” správa, na zachovanie spojenia.

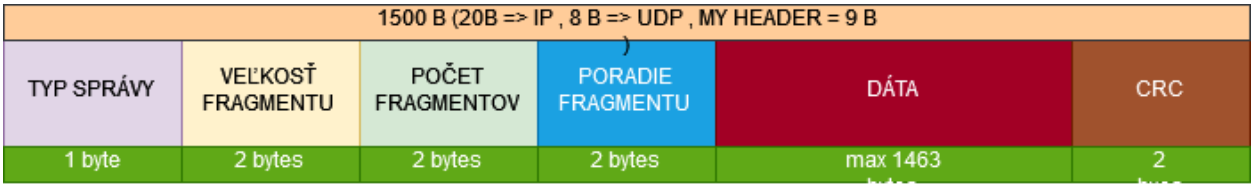
Po odoslaní desiatich fragmentov klient čaká na správu od servera o správnosti, ak server ohlási chybové dáta, tak ich klient zaradí do queue a ak prídu na rad tak sa znova pošlu.



Martin Hric
ID: 111696

Hlavička

Takto vyzerá celý fragment aj s dátami.



Typy správ

DEC	BIN	Význam
1	001	Inicializácia spojenia
2	010	Odosielajúci sa fragment
3	011	Chybné dáta
4	100	Keep_alive
5	101	Správne dáta

CRC

Na kontrolu správnosti fragementu, ktorý sa prijal sa použije CRC (cyclic redundancy check).

Pri odosielané dáta sú delené polynómom n- tého stupňa, pričom na koniec sa pripisuje remainder.

Pri prijímaní sa dáta znovu vydelia, ak bude zvyšok rovnaký, znamená to, že nenastala žiadna chyba.

Najčastejšie sa používa tento polynóm => $\overline{x^{16} + x^{12} + x^5 + 1}$

CRC je omnoho presnejší ako checksum, kvôli tomu som sa ho rozhodol použiť z knižnice *libscrc.*

(Veľmi skrátená verzia, presnejšie si to ešte len budem študovať ako to funguje.)

Martin Hric
ID: 111696

DIAGRAM