SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE

Fakulta informatiky a informačných technológií  
Ilkovičova 2, 842 16 Bratislava 4

**Zadanie č. 2 – Komunikácia s využitím UDP protokolu**

POČÍTAČOVÉ A KOMUNIKAČNÉ SIETE

Róbert Junas

FIIT STU

Cvičenie: Štvrtok 14:00

4.10.2021

# Zadanie

**Zadanie úlohy**

Navrhnite a implementujte program s použitím vlastného protokolu nad protokolom UDP (User Datagram Protocol) transportnej vrstvy sieťového modelu TCP/IP. Program umožní komunikáciu dvoch účastníkov v lokálnej sieti Ethernet, teda prenos textových správ a ľubovoľného súboru medzi počítačmi (uzlami).

Program bude pozostávať z dvoch častí – vysielacej a prijímacej. Vysielací uzol pošle súbor inému uzlu v sieti. Predpokladá sa, že v sieti dochádza k stratám dát. Ak je posielaný súbor väčší, ako používateľom definovaná max. veľkosť fragmentu, vysielajúca strana rozloží súbor na menšie časti - fragmenty, ktoré pošle samostatne. Maximálnu veľkosť fragmentu musí mať používateľ možnosť nastaviť takú, aby neboli znova fragmentované na linkovej vrstve.

Ak je súbor poslaný ako postupnosť fragmentov, cieľový uzol vypíše správu o prijatí fragmentu s jeho poradím a či bol prenesený bez chýb. Po prijatí celého súboru na cieľovom uzle tento zobrazí správu o jeho prijatí a absolútnu cestu, kam bol prijatý súbor uložený. Program musí obsahovať kontrolu chýb pri komunikácii a znovuvyžiadanie chybných fragmentov, vrátane pozitívneho aj negatívneho potvrdenia. Po prenesení prvého súboru pri nečinnosti komunikátor automaticky odošle paket pre udržanie spojenia každých 5-20s pokiaľ používateľ neukončí spojenie. Odporúčame riešiť cez vlastne definované signalizačné správy.

**Program musí mať nasledovné vlastnosti (minimálne):**

1. Program musí byť implementovaný v jazykoch C/C++ alebo Python s využitím knižníc na prácu s UDP socket, skompilovateľný a spustiteľný v učebniach. Odporúčame použiť python modul socket, C/C++ knižnice sys/socket.h pre linux/BSD a winsock2.h pre Windows. Iné knižnice a funkcie na prácu so socketmi musia byť schválené cvičiacim. V programe môžu byť použité aj knižnice na prácu s IP adresami a portami:

*arpa/inet.h  
netinet/in.h*

1. Program musí pracovať s dátami optimálne (napr. neukladať IP adresy do 4x int).
2. Pri posielaní súboru musí používateľovi umožniť určiť cieľovú IP a port.
3. Používateľ musí mať možnosť zvoliť si max. veľkosť fragmentu.
4. Obe komunikujúce strany musia byť schopné zobrazovať:
   1. názov a absolútnu cestu k súboru na danom uzle,
   2. veľkosť a počet fragmentov.
5. Možnosť simulovať chybu prenosu odoslaním minimálne 1 chybného fragmentu pri prenose súboru (do dátovej časti fragmentu je cielene vnesená chyba, to znamená, že prijímajúca strana deteguje chybu pri prenose).
6. Prijímajúca strana musí byť schopná oznámiť odosielateľovi správne aj nesprávne doručenie fragmentov. Pri nesprávnom doručení fragmentu vyžiada znovu poslať poškodené dáta.
7. Možnosť odoslať 2MB súbor a v tom prípade ich uložiť na prijímacej strane ako rovnaký súbor, pričom používateľ zadáva iba cestu k adresáru kde má byť uložený.

**Odovzdáva sa:**

1. Návrh riešenia
2. Predvedenie riešenia v súlade s prezentovaným návrhom

Program musí byť organizovaný tak, aby oba komunikujúce uzly mohli prepínať medzi funkciou vysielača a prijímača bez reštartu programu - program nemusí (ale môže) byť vysielač a prijímač súčasne. Pri predvedení riešenia je podmienkou hodnotenia schopnosť doimplementovať jednoduchú funkcionalitu na cvičení.

**Hodnotenie**

Celé riešenie - max. 20 bodov (min. 7), z toho:

- max. 5 bodov za návrh riešenia;

- max. 1 bod za doplnenú funkčnosť (doimplementáciu) priamo na cvičení v požadovanom termíne podľa harmonogramu cvičení; V prípade, ak študent nesplní úlohu zadanú priamo na cvičeniach, nehodnotí sa výsledné riešenie;

- max. 14 bodov za výsledné riešenie.

Návrh a zdrojový kód implementácie študent odovzdáva v elektronickom tvare do AISu v určených termínoch.

# Návrh

## Hlavička

0 7 15 31

|  |  |  |
| --- | --- | --- |
| Typ | Flags | size |
| seqNum | | ackNum |
|  | | |

Rámec môže mať niekoľko typov, ktoré určujú ako sa k nim bude program správať:

* Control – 0x00 – na kontrolné rámce, ako ukončenie, udržanie spojenia alebo na zistenie či je cieľ dosiahnuteľný,
* Text – 0x01 – na posielanie textovej správy,
* File – 0x02 – posielanie súboru.

Bajt, v ktorom je pole flag, určuje o akú správu sa jedná:

* SYN – 0x01
* FIN – 0x02
* ACK – 0x04
* RESEND – 0x08
* FRAGMENTED – 0x10

## Checksum

Kontrola chýb, resp. checksum pridaný na koniec posielaného rámca, bude vykonávaná pomocou CRC16 s polynómom 0x1021 (4129), resp. 0x11021. Táto funkcionalita bude zložená z dvoch funkcií jednej na overenie korektnosti prijatých dát a druhej pre vytvorenie kontrolnej hodnoty.

## ARQ

## Udržiavanie spojenia

## Fragmentácia