**SLOVENKSÁ TECHNICKÁ UNIVERZITA V BRATISLAVE  
Fakulta informatiky a informačných technológií**

**Ilkovičova 3, 842 19 Bratislava 4**

**PKS – 2. Zadanie: (Návrh) Komunikácia s využitím UDP protokolu**

Martin Beňa  
FIIT STU  
Cvičenie: Utorok 16:00  
27.11.2022

# Teória k UPD a linkovej vrstve

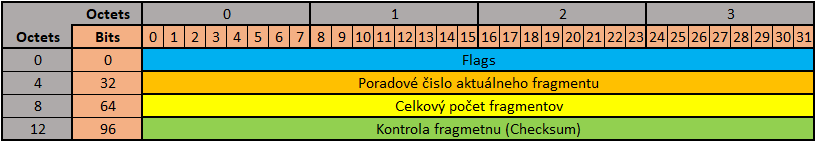
UDP je protokol, ktorý pracuje nad IP protokol a zabezpečuje rýchlejší prenos dát ako TCP ale nekontroluje či pakety prišli v neporušené, v správnom poradí alebo niektoré pakety neprišli viackrát. UDP sa používa pri online hrách, video hovoroch, telefonátoch alebo DNS.  
Maximálna veľkosť rámcu posielaného na linkovej vrstve je 1518 bajtov. Avšak netreba zabudnúť, že z tejto hodnoty treba odpočítať veľkosť IP(20B), ethernet(18B) a UDP(8B) hlavičiek. 1518 – 20 – 18 – 8 = 1472. Po odrátaní sme zistili, že nám ostalo 1472 bajtov od ktorých musíme ešte odrátať veľkosť našej hlavičky.

# Vlastná hlavička

Posielané dáta budem obaľovať do vlastnej hlavičky, ktorá bude zostavená z 4 častí: flags, číslo aktuálneho fragmentu, celkový počet fragmentov a kontrolná CheckSum. Všetky tieto hodnoty budú typu uint32 (cele číslo bez znamienka na 4 bajty) a preto bude mať moja hlavička veľkosť 16 bajtov a maximálna veľkosť posielaného rámca teda bude bude 1472 – 16 = 1456 bajtov.

Zvažujem ešte možnosť, že flags budem ukladať ako uint8 a tým ušetrím 3 bajty.

Na obrázku č.1 nižšie je vizuálna reprezentácia toho ako bude moje hlavička vyzerať.

  
obr.č.1

## Flags

Táto časť hlavičky bude obsahovať informáciu o type správy. V tabuľke č1 sú zobrazené všetky aktuálne možnosti pre typy správ.

|  |  |  |
| --- | --- | --- |
| **Flags** | **Typ správy** |  |
| 0 | Žiadosť o zahájenie komunikácie | 2-way-handshake |
| 1 | Potvrdenie o prijatí zahájenie komunikácie |
| 2 | Textová správa |  |
| 3 | Názov súboru |  |
| 4 | Obsah súboru |  |
| 5 | Potvrdenie o úspešnom prijatí |  |
| 6 | Potvrdenie o neúspešnom prijatí |  |
| 7 | Keep alive |  |
| 8 | Žiadosť o výmenu rolí (klient/server) |  |
| 9 | Ukončenie komunikácie |  |

Tabuľka č.1

## Aktuálny a celkový počet fragmentov

Tieto 2 časti budú v sebe uchovávať informácie o aktuálne poslanom rámci a celkovom počte očakávaných rámcov. Tieto informácie sú dôležite aby prijímateľ vedel usporiadať prijate rámce do správneho poradia, nakoľko ich poradenie nie je garantované a taktiež bude vedieť kedy prijal všetky rámce.

## CheckSum

Posledná časť bude obsahovať výsledok z kontroly posielaného rámca. Po prijatí si prijímateľ vypočíta vlastnú hodnotu a skontroluje si ju s doručenou. Pokiaľ by sa líšili, tak došlo k chybe pri posielaní a požiada o odosielateľa o opätovné zaslanie.

# CheckSum

Pre kontrolu som sa rozhodol použiť CRC metódu. Táto metóda funguje na princípe exkluzívneho OR-u a binárneho posunu v doprava. Najprv sa nastaví polynóm, ktorým sa bude vstup XOR-ovať (porovnávať). Pri crc32 je to polynóm x32 + x26 + x23 + x22 + x16 + x12 + x11 + x10 + x8 + x7 + x5 + x4 + x2 + x + 1. Avšak ešte pred XOR-ovaním sa k vstupu pridá 32 núl. Potom sa výsledok podelí 2 (binárny posun vpravo). Tento proces sa opakuje až dokým nebudeme mať 32 bitové číslo.

# ARQ

ARQ metóda nám dohliada na to aby boli všetky rámce doručené. Pokiaľ by nejaký rámce chýbal alebo by nebol doručený v čas, tak si príjemca vyžiada o jeho opätovne zaslanie.

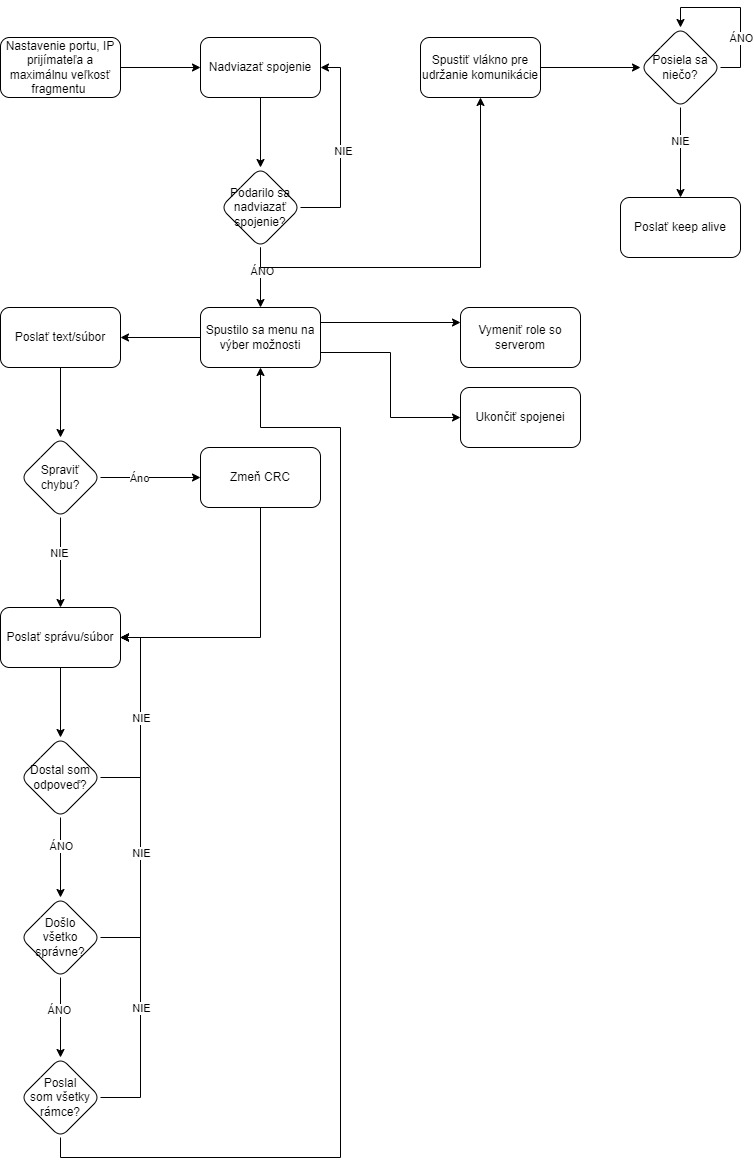
V mojom riešení využijem metódu Stop & Wait n, kde odosielateľ pošle n rámcov a čaká na správu o ich prijatí. Pokiaľ by dostal správu o chybovom prijatí, tak pošle opäť n rámcov. S touto metódou budem posielať niektoré rámce zbytočne dvakrát ale nebudem musieť čakať na odpoveď na každí jeden rámce.

# Keep alive

Na udržanie spojenia budem používať metódu keep alive, kde odosielateľ bude v časových intervaloch posielať správy bez dát s *flagon* 7 (Keep alive). Táto správa sa bude vykonávať v samostatnom vlákne a časovač intervalu odoslanie bude plynúť iba, ak bude používateľ neaktívny.

# Diagram

## Klient (Odosielateľ)



## Prijímateľ (Server)

