Komunikácia s využitím UDP

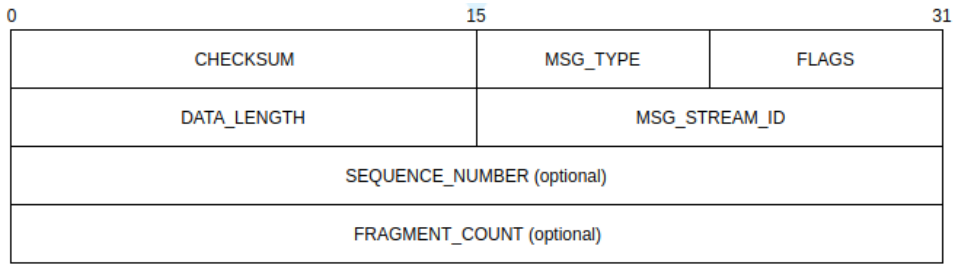
Patrik Siget

Opis zadania

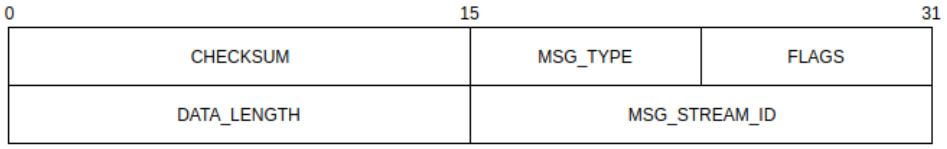
Ide o implementáciu komunikátora s využitím UDP datagramov. Keďže UDP nepodporuje aktívnu komunikáciu, ošetrovanie chyb a usporadúvanie parketov je treba nad UDP protokolom vytvoriť pseudo-protokol, ktorý tieto problémy rieši. Na začiatku je treba si definovať hlavičku tohto protokolu, všetky možne správy, tak aby oba komunikátory mohli podľa nich očakávať nasledujúce správy. Ak sa stane niečo neočakávané, napr. Príde sprav viac/menej, mal by na to patrične komunikátor zareagovať a to zaslaním požiadavky o znovu odoslanie konkrétnej správy. Inými slovami zabezpečiť plynulú a bezchybnú komunikáciu.

Hlavička

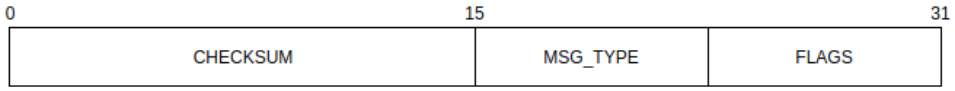
Hlavička je definovaná nasledovne:



Avšak pri rôznych typoch správ (definované nižšie) sa niektoré polia v hlavičke nenachádzajú. Z pravidla vždy sa vynechajú posledné polia ak nie sú potrebné, tj. pri posielaní správy o veľkosti menšej ako “max veľkosť fragmentu” nie je potrebné pole “SEQUENCE_NUMBER” ani “FRAGMENT_COUNT”, a teda sa vynechajú. Hlavička teda vyzerá takto:



Z dôvodu optimalizácie som definoval aj hlavičku pre správy, ktoré nepotrebujú ani polia “DATA_LENGTH” a “MSG_STREAM_ID” a teda sa vynechajú. Takúto hlavičku využíva správa typu "KEEP_ALIVE", za ktorou nenasledujú žiadne dáta.



Polia v hlavičke:

CHECKSUM – kontrolný súčet
MSG_TYPE – typ správy
- 0 – No data
- 1 – Binary data (file transfer)
- 2 – Raw ascii text

FLAGS - S A F I M R X E – 8bit (1Byte)
- S – Start connection
- A – Acknowledgement
- F – Finish
- I – Message id field is set
- M – Incomming new message stream.
- R – Resend
- E – End of message stream

DATA_LENGTH - dĺžka pola dát daného datagramu
MSG_STREAM_ID - unikátne id toku správ, každý fragment jednej správy má rovnaké ID
SEQUENCE_NUMBER – poradové číslo fragmentu
FRAGMENT_COUNT - počet fragmentov daného toku

Fragmentácia

Správy sa fragmentujú defaultne, ak je daná správa/súbor väčší ako 512Byteov. Posielajúci má samozrejme možnosť zvoliť maximálnu veľkosť jednej správy v rozsahu od 1B do 512B. Dôvodom je skutočnosť, že podľa zadania, jediná vrstva, kde má fragmentácia nastať je aplikačná, teda náš vlastný protokol. 512B max je preto, že v sieti je väčšina zariadení nastavená tak, aby fragmentovala automaticky IP packety väčšie ako 576B. 576-16(max veľkosť mojej hlavičky) - 8B (veľkosť UDP hlavičky) = 550B. Po prezretí mnoho materiálov je jasné odporúčanie 512B pre to, aby v žiadnom prípade nenastala fragmentácia v nižšej vrstve.

Implementácia:

Sender nastaví maximálnu veľkosť správy a podľa nej správu rozseká na fragmenty. Pošle prvú správu s MSG_TYPE = 0, nastavenými bitmi v poli FLAGS na “I M”, nastaveným unikátnym MSG_STREAM_ID (poradové číslo toku/správy od nadviazania komunikácie). Táto prvá správa neobsahuje žiadne dáta a preto pole DATA_LENGTH je rovné 0.

Sender ďalej čaká na rovnakú správu, ale navyše s nastaveným bitom A (acknowledgement), aby vedel, či je Reciever schopný prijať správu a potom začne posielať postupne fragmentované dáta.

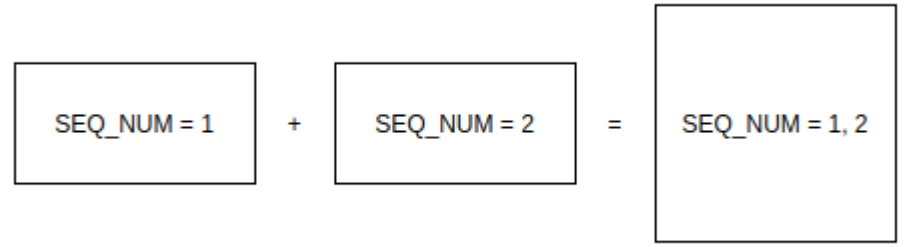
Nakoniec Sender pošle prázdnu správu (rovnaké datľa ako pri prvej) ale s nastaveným FLAGOM na IMF. Čaká na ACK alebo Resend nejakého fragmentu. Ak príde ACK na poslednú správu, Sender pošle ešte jeden ACK a tým je prenos ukončený. (pozri sekciu PRIKLADY)

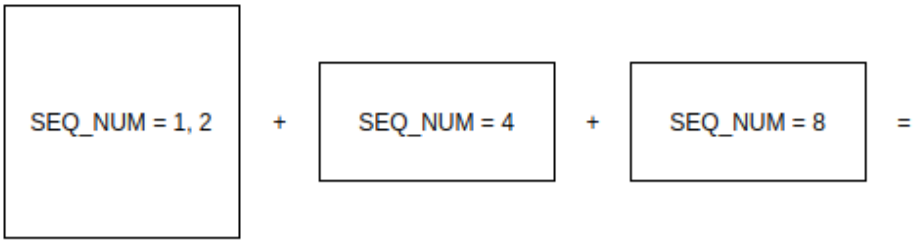
Kontrola chýb

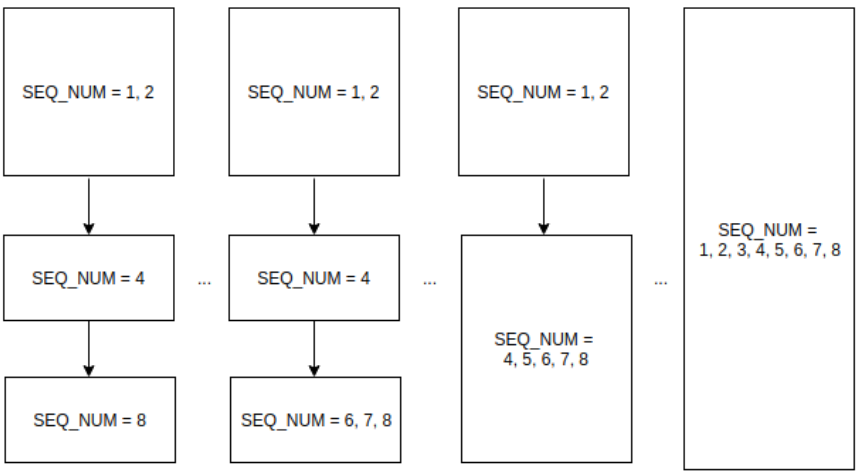
Tá je zabezpečená kontrolným súčtom generovaným funkciou crc16 s pevne danou maskou, odporúčanou organizáciou CCITT a to x**16 + x**12 + x**5 + 1. Pokiaľ reciever príjme správu, ktorej vypočítaný checksum nezodpovedá tomu v správe, túto správu uloží do zásobníka nesprávne prijatých správ a na po prijatí poslednej, ukončovacej správy si všetky chybné vypýta znova. Až pokým sa nepodarí zložiť celú správu. Ďalšia chyba, aká môže nastať je taká, že správy chodia neskoro, alebo dokonca vôbec neprídu. V takom prípade sa čaká 15 sekúnd na správu a nasledne sa spojenie preruší.

Usporadúvanie poradia fragmentov

Prichádzajúce fragmenty sa ukladajú do linked listu, postupne ako prichádzajú, a následne sa spájajú. Spájanie je vykonané pri ukladaní každého ďalšieho fragmentu, a to tak, že ak je ďalší fragment v správnom poradí, spojí sa s predošlým. Ak však nebude, vytvorí sa nový záznam a zaradí sa podľa poradia do listu.







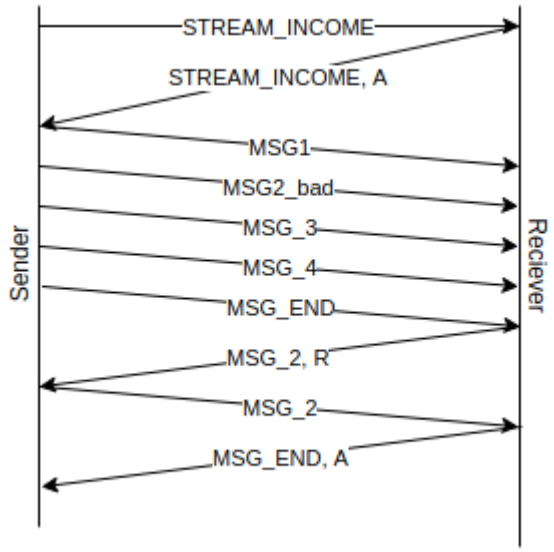
Štruktúra listu záznamu je definovaná:

```
struct fragment{
        int merged_fragments[2]
        unsigned char* data;
        Struct fragment * next;
        Struct fragment * prev;
}
```

Jeden záznam bude teda obsahovať pole dvoch int-ov. To bude poradie spojených správ od – do, vrátane. Dáta a pointre na predošlý záznam a nasledujúci.

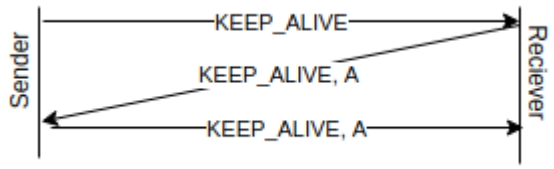
Znovu vyžiadanie chybných správ

Bude riešené až po prijatí poslednej správy. Reciever si ukladá všetky zahodené správy do linked listu a postupne si pýta každú správu s nastaveným bitom R s príslušnou hlavičkou. Až keď má reciever všetky správy, odošle Ack o prijatí poslednej(ukončovacej správy).



Udržiavanie spojenia

Bude implementované počítadlo sekúnd od poslednej správy od partnera a každých 15 sekúnd sa pošle KEEP_ALIVE správa. Priebeh je nasledovný:



Príklady

Súbor, 10kb, max veľkosť nedefinovaná: