Slovenská Technická Univerzita v Bratislave
Fakulta Informatiky a Informačných Technológií

Počítačové a komunikačné siete

Zadanie 2 - Komunikácia s využitím UDP protokolu

Marek Adamovič

Cvičenie: Piatok 10:00 – 11:40
Ing. Peter Kaňuch

2020/2021

Slovenská Technická Univerzita v Bratislave
Fakulta Informatiky a Informačných Technológií

Zadanie

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 10-60s pokiaľ používateľ neukončí spojenie. Odporúčame riešiť cez vlastne definované signalizačné správy.

Analýza

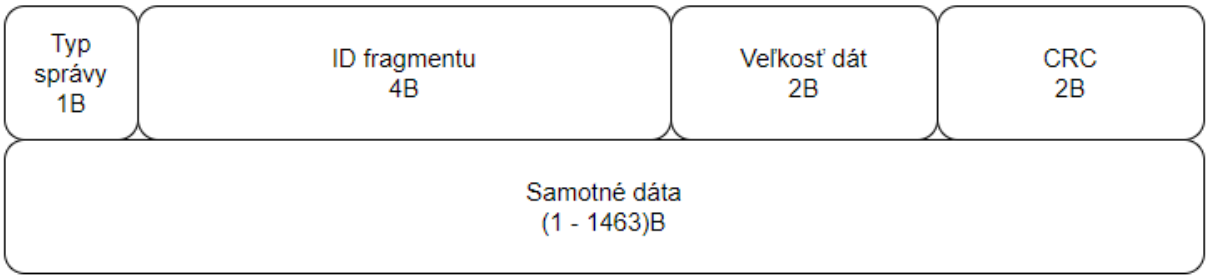
Mojou úlohou je spraviť vlastný protokol, podobný TCP, s viacerými typmi správ, nad protokolom UDP. Klasický TCP používa osobitné správy pre začatie komunikácie, udržiavanie a ukončovanie, všetky označené s pár bitmi. V mojom protokole budem uvažovať podobný systém. Veľkosť jedného fragmentu bude limitovaná linkovou vrstvou (vyplýva zo zadania), teda 1500B, od ktorých budem odčítavať veľkosť IP hlavičky a hlavičky môjho protokolu.

Návrh

Hlavička:

- 1B použijem na typ správy, jeho rôzne hodnoty budú predstavovať rôzne typy správ

- 4B použijem ako identifikátor fragmentu, resp. id fragmentu, podľa ktorých budem na strane prijímateľa dávať súbor dokopy (fragmentov môže byť veľa, preto potrebujeme až 4B)
- 2B použijem na veľkosť dát (stačí mi short, keďže veľkosť nemôže byť väčšia ako 1500B)
- 2B využijem pre CRC, teda kontrolný súčet, či sa dáta doručili korektne .. použijem CRC-16, ktorého výsledok sú práve 2B, čo nie je ani málo (veľa kolízií), ani veľa (pamäťovo náročnejšie)

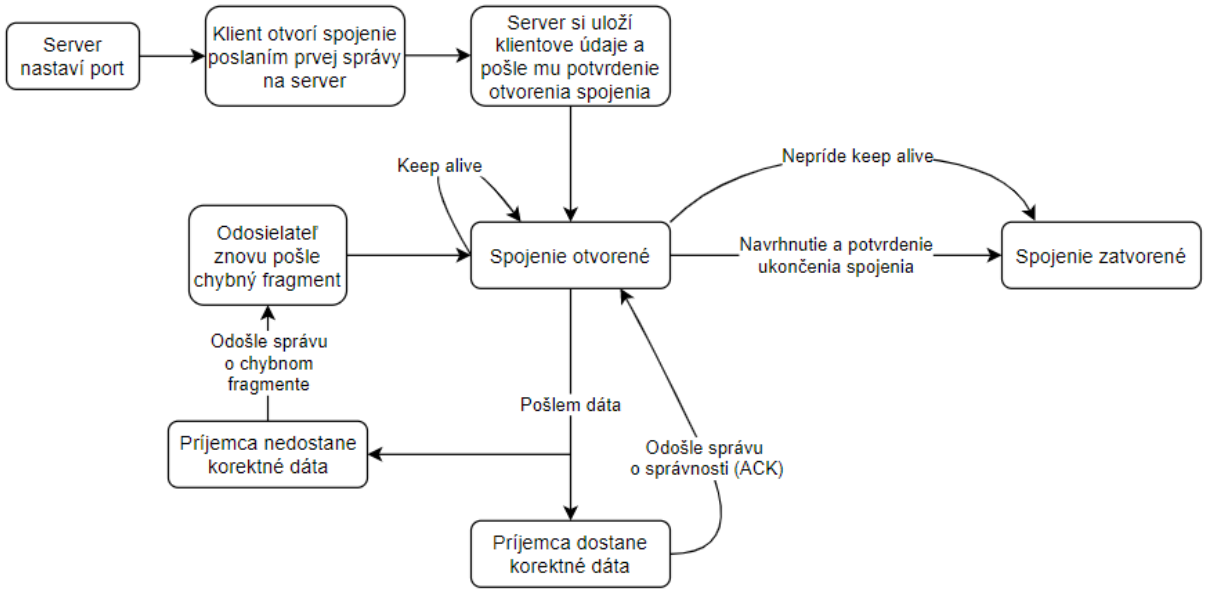


Minimálna veľkosť dát je 1B, čo s mojou hlavičkou činí 10B (avšak signalizačné správy neobsahujú dáta). Maximálna veľkosť samotných dát je 1463B, čo je 1500B - 20B (IP hlavička) – 8B (UDP hlavička) - 9B (hlavička môjho protokolu). Toto bude predstavovať interval výberu veľkosti fragmentu pre užívateľa.

Typy správ:

- pre začatie komunikácie (hodnota 1.B = 1)
- odosielanie dát (hodnota 1.B = 2)
- potvrdenie, že som dostal dáta korektne (hodnota 1.B = 3)
- potvrdenie, že som dostal dáta s chybou, error (hodnota 1.B = 4)
- odosielanie dát o súbore, názov, prípona (hodnota 1.B = 5)
- ~~správa udržiavajúca komunikáciu, keep alive (hodnota 1.B = 6)~~
- pre ukončenie komunikácie (hodnota 1.B = 7)
- pre uzavretie súboru (hodnota 1.B = 9)
- pre poslanie módu programu – budeš prijímateľ (hodnota 1.B = 10)
- pre poslanie módu programu – budeš odosielateľ (hodnota 1.B = 11)
- pôjde o normálne správy (hodnota 1.B = 12)
- pôjde o posielanie súboru (hodnota 1.B = 13)
- koniec konkrétneho posielania sprav (hodnota 1.B = 14)

Tok programu



Pre úspešné otvorenie spojenia bude treba 2 správy pre začatie komunikácie (jedna od odosielateľa, druhá od prijímateľa). Ten istý systém bude platiť pri ukončovaní (potreba jednej správy od každého). **Potvrdenie správnosti fragmentov sa bude posielať po každom jednom fragmente.** Pri nekorektných dátach sa pošle error správa, ktorá vyžiada znovu-poslanie daného fragmentu. ~~Keep alive bude chodiť približne každých 60 sekúnd (a následná odpoveď). Ak nepríde, spojenie sa preruší.~~ ~~**Keep alive sa nebude posielať počas posielanie súborov.**~~

Používateľské rozhranie

Zadávanie príkazov do konzoly, kde taktiež budeme môcť sledovať informácie o doručení fragmentov a podobne. Program využíva klient/server rozhranie.

Na začiatku užívateľ zapne program na oboch uzloch a zvolí, ktorý bude server a ktorý bude klient.

```
Vyber si stranu programu:
1  ->  Server
2  ->  Klient
1
```

Následne zadá IP adresu a port pre server a potom tieto isté údaje zadá aj na strane klienta (aby klient vedel, kde sa má pripojiť). Po zadaní na strane klienta sa spojenie úspešne vytvorí.

```
Nastav IP adresu serveru (napr. 192.168.0.248)
192.168.0.45
Nastav port serveru (napr. 5002)
5002
Prisiel pociatocny fragment, robim spojenie a posielam odpoved
```

Teraz sa užívateľ nachádza v hlavnom menu (kde sa vždy po poslaní správ/súboru vráti). Vyberie si, ktorá strana bude posielať dáta, ktorá prijímať, maximálnu veľkosť dátovej časti fragmentov a či chce posielať správy alebo súbor.

```
Kto bude odosielatel?
1  ->  ja
2  ->  druha strana
3  ->  koniec spojenia
1
Zadaj maximalnu velkost dat fragmentu z rozmedzia 1 - 1463
10
Chces poslat spravu alebo subor?
sprava  ->  1
subor  ->  2
1
Poslal som info, ze pojde o spravy
Dostal som potvrdenie
```

Ak si vyberie správy, vždy dostane informáciu o (ne)správnom doručení. Správou sa myslí string oddelený novým riadkom (teda enterom). Po poslaní správy sa dostáva späť do hlavného menu.

```
Ahoj
Poslal som 4B dat v sprave s ID 2
Prislo potvrdenie o spravnom prijati spravy s ID: 2
Chces poslat dalsiu spravu?
0  ->  nie
1  ->  ano
1
Ahoj cau
Poslal som 8B dat v sprave s ID 3
Prislo potvrdenie o spravnom prijati spravy s ID: 3
Chces poslat dalsiu spravu?
```

Keď si užívateľ vyberie súbor, bude musieť zadať absolútnu cestu k súboru, ktorý chce odoslať. Ak zadá nesprávnu cestu, program mu umožní ju zadať znovu (až do úspešného otvorenia súboru). Následne sa pošlú informácie o názve súboru

(taktiež s príponou), ktoré sa môžu posielať vo viacerých fragmentoch, ak zvolená veľkosť užívateľa bude príliš malá.

```
Zadaj absolutnu cestu k suboru, co chces poslat
C:\Users\Admin\Desktop\pks2\to_send\1.jpg
Subor 1.jpg sa podarilo otvorit
Posielam fragment s datami o subore s ID 2 o velkosti 5
Dostal som potvrdenie o spravnosti prijatych dat o subore vo fragmente s ID 2
```

Teraz musí užívateľ zadať na druhej strane cestu uloženia a následne začne prenos súboru.

```
Pojde o posielanie suborov
Nazov suboru je 1.jpg
Zadaj cestu priecinku a zahajim prenos
C:\Users\Marek\Desktop\pks2\recieved
```

Keď sa ukončí posielanie, obe strany nám vypíšu informácie o prijatí fragmentov a celkovú veľkosť dát. Pre pokoj v duši môžeme skontrolovať na strane odosielateľa veľkosť súboru, či sa nič nestratilo.

```
Dostal som ziadost o zavretie suboru, zatvaram a posielam potvrdenie
Subor prisiel v 31972 fragmentoch (aj chybne) obsahujucich dokopy 3197045 B dat
Absolutna cesta ulozenia je C:\Users\Marek\Desktop\pks2\recieved\1.jpg
```

Umiestnenie:	C:\Users\Admin\Desktop\pks2\to_send
Veľkosť:	3,04 MB (3 197 045 bajtov)

Dokončením posielania sa odosielateľ dostane naspäť do hlavného menu.

```
Subor bol uspesne poslany v 31972 fragmentoch (chybne sa pocitaju tiez)
Celkove data suboru cinia 3197045 B
Kto bude odosielatel?
1  ->  ja
2  ->  druha strana
3  ->  koniec spojenia
```

Pridané veci oproti návrhu

- nový typ správy 9, pre ukončenie a uzavretie prenosu jedného súboru

- nové typy správ 10 a 11, pre poslanie informácie o tom, kto bude odosielateľ a kto prijímateľ

- nové typy správ 12 a 13, pre poslanie informácie o tom, či budeme posielať správy alebo súbor

- nový typ správy 14, ktorý znamená koniec súčasného posielania (a návrat do menu)

- keep alive nebol implementovaný, spojenie sa dá ukončiť iba manuálne

- hlavička ostala rovnaká, systém potvrdzovania fragmentov tak isto

Splnená funkcionalita

- optimálna práca s dátami
- možnosť určenia IP adresy a portu
- možnosť voľby max. veľkosti fragmentu
- zobrazovanie veľkosti a počtu fragmentov
- vypísanie cesty k prenášanému súboru
- možnosť simulácie chyby
- signály (spätné správy) o ne/správnom doručení dát
- možnosť odoslať aspoň 2MB súbor (správnosť testovaná aj pre 200MB video)
- zachovanie názvu a prípony súboru
- implementácia v jazyku C za pomoci knižnice winsock2.h
- možnosť prepínať medzi prijímateľom a odosielateľom na uzloch
- CRC-16 pre kontrolu chýb

Záver

Snažil som sa postaviť riešenie na zrozumiteľnosti a ľahkej práci so samotným programom. Vďaka konzole si udržuje obstojnú rýchlosť (ak by sme chceli riešenie ešte zrýchliť, stačí nám zakomentovať menej dôležité výpisy). Nedostatkom riešenia je určite absencia funkcionality keep alive, ktorú som sa rozhodol neimplementovať pre nedostatok času a taktiež skúseností s threadmi. Rád by som vyzdvihol možnosť fragmentácie informácií o samotnom súbore (názov a prípona), keďže táto funkcionalita nebola úplne povinná. Ak by som refaktoroval kód, snažil by som sa ho ešte viacej skomprimovať a zjednodušiť Vďaka zadaniu som získal prehľad o fungovaní komunikácií cez vrstvový model a vyskúšal si, ako sa takéto veci programujú.