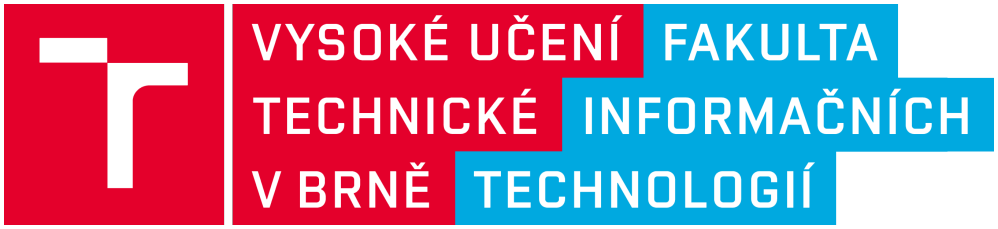

Dokumentácia k projektu z predmetu ISA
ISA - Síťové aplikace a správa sítí
2021/2022

Programovanie sieťovej služby: Klient POP3 s podporou TLS

Šimon Feňko (xfenko01)
xfenko01@stud.fit.vutbr.cz

12. novembra 2021

Obsah

1 Úvod

Tento dokument je dokumentácia k projektu Programování síťové služby: Klient POP3 s podporou TLS do predmetu ISA - Síťové aplikace a správa sítí na Fakulte Informačných technológií, VUT v Brně. Úlohou tohto projektu bolo vytvoriť konzolovú aplikáciu, ktorá je schopná pomocou protokolu POP3 komunikovať so serverom. V tom je zahrnuté aj sťahovanie a ukladanie do súborov e-mailové správy. Pre lepšie pochopenie projektu, je nutné mať základné vedomosti z oblasti počítačových sietí.

1.1 Komunikácia typu Klient-Server

Sieťová architektúra Klient-Server oddeluje klienta, ktorý môže byť napríklad hostiteľský počítač, a server. Oni spolu navzájom komunikujú za pomoci počítačovej siete. Server odpovedá na požiadavky, ktoré mu odošle klient. Pravidlá komunikácie sú riadené protokolom.

1.2 POP3 Protokol

Protokol POP3 je aplikačný protokol, ktorý sa využíva k prístupu k e-mailovej schránke, ktorá je na vzdialenom servery a pre prijímanie elektronickej pošty. POP je skratka z anglického výrazu Post Office Protocol a číslo 3 značí tretiu verziu tohoto protokolu. Pri komunikácii so serverom klient zasiela textovú žiadosť a server mu zašle textovú odpoveď:

- v prípade úspechu +OK
- v prípade neúspechu -ERR

Podľa zvoleného typu dotazu je odoslaná odpoveď:

- jednoriadková - ukončená znakmi "\r\n"
- viacriadková - ukončená znakmi "\r\n.\r\n"

2 Návrh aplikácie

V tejto časti sa zameriam na popísanie spôsobu riešenia, ktoré som si zvolil.

2.1 Spracovanie argumentov

Ako prvý krok po spustení programu, je potrebné zistiť, či užívateľ zadal valídne parametre. Za pomoci podmienok sa kontrolujú a zároveň prebieha "parsingärgumentov. Prebieha cyklus, v ktorom sa jednotlivé parametre kontrolujú a na základe toho aký parameter užívateľ zadá, sa danej pomocnej boolean premennej, ktorá je viazaná na daný parameter nastaví hodnota true. To zabráni opätovnému zadávaniu toho istého parametru, pretože stále pri obdržaní parametru sa kontroluje, či je pomocná boolean premenná nastavená na hodnotu false a tým pádom vieme, že jej výskyt je prvýkrát. Následne prebieha druhá časť kontroly, a to kontrola, či boli zadané všetky povinné parametre. V prípade chybného parametru sa program ukončí s návratovou hodnotou 1.

2.2 Pripojenie k serveru[3]

Ak užívateľ zadal ako parameter `-T`, je volaná funkcia `connect_to_sec_server()`.Pomocou nej sa vykoná pripojenie, ktoré je šifrované na východzom porte 995 a overí certifikát serveru.

V prípade, že uťivateľ zadá parameter `-S`, tak sa zavolá funkcia `connect_to_stls_server()`. Tá najprv naviaže so serverom nešifroév anéspojenie na východzom porte 110. Následne je zaslaný príkaz STLS, ktorý overí, či server podporuje rozšírenie STARTLS. V prípade, že príde kladná odpoveď, je

komunikácia prepnutá do šifrovanej verzie. Overovanie certifikátov je rovnaké ako pri zadaní parametru `-T`.

V ostatných prípadoch, kedy nieje zadaný parameter `-T` ani `-S`, sa zavolá funkcia `connect_to_server()`. Vďaka nej je neviazané nešifrované spojenie na porte 110 a správy sú odosielané nezabezpečene.

2.3 Autorizácia

Pri úspešnom pripojení sa otvorí autorizačný súbor `<auth_file>`, z ktorého sa načítaju údaje k prihláseniu. Tento súbor obsahuje meno a heslo k prihláseniu do určitej schránky na servery. Štruktúra je pevne daná následovne:
username = meno
password = heslo
Pričom za **"meno"** a **"heslo"** dosadíme konkrétne zvolené hodnoty.

Na skontrolovanie správnych údajov na prihlásenie, je volaná funkcia `load_credentials()`. Pomocou nej načítame údaje zo súboru `auth_file`. Funguje tak, že sa v nej zavolá funkcia funkcia `split()`[1], ktorá na základe nejakého oddeľovača string na časti. V našom prípade rozdeľujeme pomocou znaku -ä následne vieme pristúpiť k danému prihlasovaciemu menu a heslu, ktoré následne vieme odkontrolovať, či sú zadané v `auth_file`. Údaje, ktoré sme zistili sú nasledne parametrami pri volaní funkcie `login()`. Tá zaistí prihlásenie do emailovej schránky uživateľa pomocou funkcie `USER`[2] a `PASS`[2].

2.4 Komunikácia so serverom

Klient posiela požiadavky serveru, ktorý na ne odpovedá:

- `+OK`<*odpoveď serveru*> v prípade, že všetko prebehlo ako má
- `-ERR`<*odpoveď serveru*> v prípade, že došlo k nejakej chybe.

2.5 Ukladanie správ

Pri sťahovaní a ukladaní neprečítaných správ z e-mailovej schránky uživateľa postupuje program tak, že skontroluje či je pomocná premenná isnew nastavená na hodnotu true, čo znamená, že bol zadaný parameter -n, čo znamená sťahovanie iba nových správ, ktoré ešte neboli stiahnuté. Vtedy sa zavolá funkcia `download_new_messages()`. V prípade, že je pomocná premenná `isnew` nastavená na hodnotu false, voláme funkciu `download_all_messages()`, pretože vieme, že nebol zadaný parameter `-n` a budeme sťahovať všetky správy z e-mailovej schránky uživateľa.

`Funkcia download_all_messages()` funguje tak, že skontroluje, či je už existujúca zložka `out_dir`. V prípade, že existuje, tak sa zavolá funkcia `STAT`[2] a následne funkcia `RETR`[2], pomocou ktorých sa do nej stiahnu všetky neprečítané správy z emailovej schránky. V prípade, že zložka neexistuje, tak ju program vytvorí a následne vykoná rovnaký postup na uloženie správ.

`Funkcia download_new_messages()` funguje tak, že skontroluje, či existuje zadaná zložka `out_dir`. V prípade, že existuje, tak načíta súbory v nej (uložené správy, z e-mailovej schránky). Následne pomocou funkcie `split()` rozdelí názov každej správy na základe znaku "_ä znaku ".", pretože názov uloženej správy je vo formáte **mail_[1-9]*.txt**. Týmto krokom sa dostane k číslam správ. Následne si ich zoradí a číslo s maximálnou hodnotou uloží. Zavolá funkcie `STAT`, ktorá nám vráti počet neprečítaných správ v emailovej schránke. Potom sa zavolá funkcia `RETR`, ktorá stiahne správy aj so svojim ID. Pomocou tohto údaju si vyráta počet nových a neprečítaných správ, ktoré ešte neboli stiahnuté. Tie sa následne uložia k už stiahnutým správam a program vypíše počet novo stiahnutých správ. V prípade, že zložka `out_dir` neexistuje, tak vieme, že ešte neboli žiadne maily stiahnuté a tým pádom stiahne a uloží všetky neprečítané správy zo schránky, ktoré nájde.

2.6 Mazanie správ

Pri mazaní správ program kontroluje, či bol zadaný parameter `-n`. V prípade, že bol,tak sa zavolá funkcia `delete_new_messages`, ktorá má za úlohu zmazať len neprečítané správy, ktoré neboli ešte stiahnuté a teda sú nové. Vtedy počet nových správ zisťujeme rovnako ako pri sťahovaní a ukladaní. Následne sa zavolá funkcia `DELE`[2], ktorá ich zmaže z e-mailovej schránky.

V prípade, že parameter `-n` nebol zadaný, tak sa volá funkcia `delete_all_messages`, ktorá v sebe volá funkciu `DELE`, ktorá následne odstráni všetky neprečítané správy z e-mailovej schránky.

3 Implementácia

Program je implementovaný v jazyku `C++`. Výsledná aplikácia je vytvorená pre `OS Linux` a bola otestovaná na systémoch `Centos 7`(merlin.fit.vutbr.cz).

4 Spustenie programu

Aby sa vytvoril spustiteľný súbor popcl, je nutné súbory preložiť pomocou príkazu make. Preložený program je možné spustiť s následovnými parametrami, ktoré môžu byť zadávané v ľubovolnom poradí:

- `./popcl<server>[-p <port>][-T/-S][-c <certfile>][-C <certaddr>]][-d][-n]-a <auth_file>-o <out_dir>`

alebo v prípade, ak chceme vypísať nápovedu, tak:

- `./popcl --help`

Popis parametrov:

- Povinné parametre:
  - `<server>` - názov požadovaného zdroju (IP adresa alebo doménové meno.
  - `-a <auth_file>` vynucuje autentizáciu (príkaz `USER`), obsah konfiguračného súboru `<auth_file>`.
  - `-o <out_dir>` špecifikuje výstupný adresár `<out_dir>`, do ktorého má program stiahnuté správy uložiť.
- Nepovinné parametre:
  - `-p` špecifikuje číslo portu `<port>` na servery.
  - `-T` zapína šifrovanie celej komunikácie (POP3s), keď nieje parameter uvedení, použije sa nešifrovaná varianta protokolu.
  - `-S` naviaže nešifrované spojenie so serverom a pomocou príkazu STLS (RFC 2595) prejde na šifrovanú variantu protokolu.
  - `-c` definuje súbor `<certfile>` s certifikátmi, ktorý sa použije pre overovanie platnosti certifikátu SSL/TLS predloženého serverom (použitie len s parametrom `-T` alebo `-S`).
  - `-C` určuje adresár `<certaddr>`, v ktorom sa majú vyhľadávať certifikáty, ktoré sa použijú pre overovanie platnosti certifikátu SSL/TLS predloženého serverom. (Použitie len s parametrom `-T` alebo `-S`).

- V prípade, že nieje uvedený ani parameter `-c` ani `-C`, potom sa použije uložisko certifikátov získané funkciou `SSL_CTX_set_default_verify_paths()`.
- `-d` zašle serveru príkaz pre zmazanie správ.
- `-n` zariadi, že sa bude pracovať (čítať) len s novými správami.

4.1 Praktická ukážka použitia:

Stiahnutie všetkých správ z e-mailovej schránky:
```
./popcl -a auth.txt -o mails pop3.seznam.cz
```

Stiahnutie všetkých správ z e-mailovej schránky a následné ich zmazanie:
```
./popcl -a auth.txt -o mails pop3.seznam.cz -d
```

Stiahnutie len nových správ z e-mailovej schránky:
```
./popcl -a auth.txt -o mails pop3.seznam.cz -n
```

Vymazanie všetkých nových správ z e-mailovej schránky:
```
./popcl -a auth.txt -o mails pop3.seznam.cz -n -d
```

Zobrazenie nápovedy:
```
./popcl --help
```

5 Základné informácie k programu

5.1 Použité knižnice

Pri tvorbe programu boli použité následujúce knižnice jazyka `c++`:

```
<vector>
<iostream>
<dirent.h>
<string>
<algorithm>
<stdio.h>
<string.h>
<regex>
‘‘openssl/bio.h‘‘
‘‘openssl/ssl.h‘‘
‘‘openssl/err.h‘‘
```

5.2 Použité súbory

```
main.cpp
pop3.cpp
pop3.h
ParseParams.cpp
ParseParams.h
```

5.3 Návratové hodnoty

V prípade, že program nájde chybovú situáciu, vypíše návratové hodnoty v týchto situáciach následovne:

Návratová hodnota	**Popis situácie**
0	Všetko prebehlo správne
1	Nesprávne vstupné parametre
2	Nesprávna štruktúra `<auth_file>`
3	Problém pri spojení so serverom
4	Nesprávne prihlasovacie údaje
5	Chyba na strane serveru
6	Nenájdený súbor alebo zložka

6 Zdroje

Referencie

[1] *Function Split()*. [ONLINE]. URL: https://stackoverflow.com/questions/14265581/parse-split-a-string-in-c-using-string-delimiter-standard-c.

[2] J. Myers. *Post Office Protocol - Version 3*. [ONLINE]. URL: https://datatracker.ietf.org/doc/html/rfc1939.

[3] Kenneth Ballard. *Secure programming with the OpenSSL API*. [ONLINE]. 2004. URL: https://developer.ibm.com/tutorials/l-openssl/.