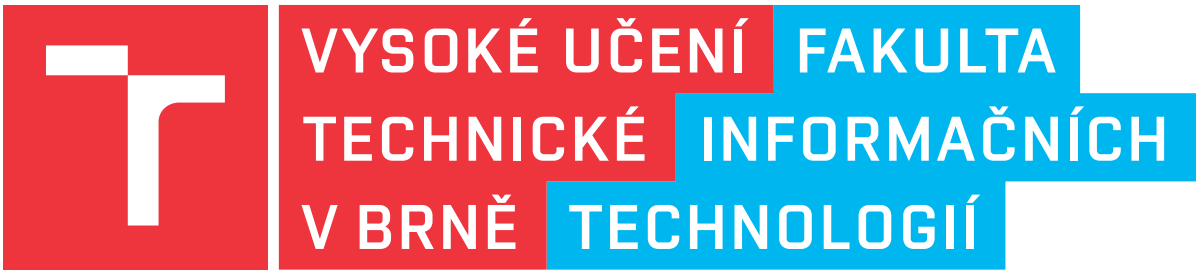


Síťové aplikace a správa sítí

Klient POP3 s podporou TLS

2017/2018

Autor: Petr Jůda (xjudap00)

26. října 2017

Obsah

1 Úvod

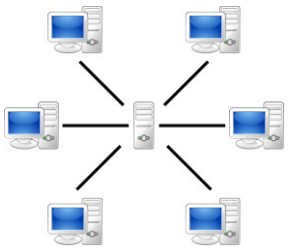
Tento dokument vznikl jako dokumentace projektu `Klient POP3 s podporou TLS` do předmětu `Síťové aplikace a správa síťí`. Cílem tohoto projektu bylo vytvořit konzolovou aplikaci, která je schopna pomocí protokolu POP3 komunikovat se serverem, stahovat a ukládat z něj e-mailové zprávy. Tento dokument se zabývá popisem návrhu, implementace a výsledného použití aplikace.

1.1 Úvod do problematiky

Pro pochopení projektu je potřeba nastudovat základní pojmy a znalosti z oblasti počítačových sítí. Nejdůležitější principy jsou vysvětleny v následujících podkapitolách.

1.1.1 Komunikace typu Klient-Server

Síťová architektura **Klient-Server** odděluje `klienta` (např. hostitelský počítač) a `server`, kteří spolu komunikují přes počítačovou síť. [6] Server odpovídá na požadavky zaslané klientem. Pravidla komunikace jsou řízena protokolem. Pro stahování e-mailové pošty se v dnešní době využívají především protokoly **IMAP** (Internet Message Access Protokol) nebo **POP3** (Post Office Protokol verze 3). V rámci projektu se dále budeme zabývat pouze protokolem POP3.



Obrázek 1: Komunikace typu Klient-Server [6]

1.1.2 Protokol POP3

Protokol POP3 je ve své základní verzi standardizován v `RFC 1939` [3]. Pro použití šifrovaného spojení TLS v projektu bylo nutné nastudovat i příkazy z `RFC 2595` [4]. Při komunikaci se serverem klient zasílá textový požadavek. Server zasílá textovou odpověď, která v případě úspěchu začíná znaky `+OK`, v případě chyby `-ERR`. Podle zvoleného typu dotazu je odeslaná odpověď jednořádková nebo víceřádková. Jednořádková zpráva je ukončena znaky `"\r\n"`. Víceřádková odpověď je pak ukončena znaky `"\r\n.\r\n"`.

Pro plné pochopení protokolu jsem využil aplikace `telnet` a `openssl` [1], které umožňují připojení k serveru a praktické vyzkoušení odesílání a přijímání zpráv.

Server:	**Klient:**
+OK Hello, this is Seznam POP3 server unknown.`\r\n`	
	USER <username>`\r\n`
+OK Enter your password please.`\r\n`	
	PASS <heslo>`\r\n`
+OK 6 703869`\r\n`	
	RETR 1`\r\n`
+OK Message follows (6377 bytes)`\r\n` <obsah e-mailu>`\r\n` .`\r\n`	
	QUIT`\r\n`
+OK Closing connection, see you later.`\r\n`	

Obrázek 2: Příklad komunikace se serverem pop3.seznam.cz

2 Návrh aplikace

Při návrhnu aplikace jsem se rozhodl využít objektově orientovaného přístupu programování, které jazyk C++ nabízí. Díky tomu je možné rozdělit jednotlivé části do oddělených tříd. Každá třída je zodpovědná za určitou funkcionalitu programu. Třídy jsou navrženy tak, aby metody byly znovupoužitelné. Vzhledem k většímu množství možných vstupních parametrů programu bylo potřeba navrhnout třídu `ProgramParams`, která se stará o zpracování, validaci a uchování parametrů zadaných uživatelem.

Třída `Pop3` je jádrem celé aplikace. Stará se o navázání a udržování komunikace s POP3 serverem, stahování a ukládání jednotlivých zpráv ve formátu `IMF` [5]. Pro potřeby stahování pouze nových zpráv jsem navrh třídu `Database`, která má za úkol ukládat a načítat informace o již stažených e-mailech. Celý chod programu je pak řízen z hlavní funkce `main()`.

3 Popis implementace

Program je implementován v jazyce C++11. Výsledná aplikace je vytvořena pro `OS Linux` a byla otestována na systémech Centos 7 (merlin.fit.vutbr.cz), Ubuntu 14.04 a MacOS 10.12.6.
Následující podkapitoly se zabývají vlastními aspekty implementace.

3.1 Zpracovní a validace parametrů

Jako prvním krok po spuštění program je potřeba zjistit, zda uživatel zadal správné parametry. Za tímto účelem je zkonstruován objekt třídy `ProgramParams` a volána metoda `ParseParams()`. Tato metoda zkontroluje, zda jsou zvolené přepínače korektní a vstupní hodnoty uloží do privátních atributů objektu. Pro identifikaci zadaných parametrů jsou v objektu nastaveny globální flagy. Následuje volání metody `ValidParams()`, která zkontroluje, zda zadané adresy souborů a adresářů existují. Kontroluje se existence výstupního adresáře a vstupního souboru s přihlašovacími údaji, případně parametry přepínačů -C a -c. Pomocí metody `LoadLoginPass()` je ze souboru načteno uživatelské jméhо a heslo.

Očekává se přesné dodržení následující struktury souboru <auth_file>. (Před klíčovými slovy username, password nesmí být žádné bílé znaky.)

```
username = "příhlašovací jméno"\n
password = "heslo"\n
EOF
```

3.2 Vytvoření spojení

O navázání komunikace se serverem se stará objekt třídy `Pop3`. Metoda `ConnectToServer()` na základě vstupního parametru rozhodne, zda má být navázáno IPv4 nebo IPv6 spojení.

Pokud byl zadán parametr -T je volána metoda `CreateSecConn()`. Ta pomocí TSL naváže šifrované spojení na výchozím portu 995 a ověří certifikát serveru. Při zadaní parametru -c se nejprve ověřuje certifikát zaslaný serverem vůči certifikátu zadaného uživatelem. Pokud byl zadán i parametr -C je certifikát ověřen vůči všem certifikátům v adresáři <cert_addr>. (Předpokládá se, že v adresáři byla zavolána systémová funkce `c_rehash`). Jinak se k ověření použije výchozí systémová cesta k certifikátům.

Při použití parametru -S je volána metoda `CreateSTLSConn()`. Ta nejprve naváže se serverm nešifrované spojení na výchozím portu 110. Po obdržení uvítací zprávy je odeslán příkaz `STLS`, který ověří zda server podporuje rozšíření STARTTLS. Pokud je odpověď pozitivní, je proveden SSL handshake a komunikace přepnuta do šifrovaného režimu. Ověření certifikátu se provádí stejným způsobem jako u parametru -T.

V ostatních případech je pomocí metody `CreateUnsecConn()` navázáno nešifrované spojení na portu 110 a zprávy jsou odesílány nezabezpečeně.

Metody využívají `BIO` objekt z knihovny `openssl`. Tento objekt má na připojení nastavený minutový timeout. Pokud je zadána špatná adresa a port serveru, je po vypršení časovače program ukončen.

Změnu hodnoty výchozího portu je požné nastavit parametrem -p.

3.3 Autentifikace a stahování e-mailů

Po úspěšném navázání spojení je zapotřebí se vůči serveru autentizovat. Metoda `Login()` zajistí přihlášení do emailové schránky uživatele. Poté je odeslán příkaz `STAT`, který zjistí počet a velikost všech zpráv uložených ve schránce. Pokud schránka obsahuje alespoň jednu zprávu, je zaslán dotaz `LIST`, který vrátí velikosti jednotlivých zpráv. Tyto velikosti jsou uloženy do atributu `sizesToDownload`. Při stahování zpráv je tento atribut využit k ověření toho, že ze serveru dorazil kompletní e-mail.

Následuje odeslání testovacího příkazu `UIDL 1`, který ověří zda server podporuje rozšíření UID. Pokud ano je při stahování jednotlivých zpráv ukládáno jejich `UID`. Pokud server tento příkaz nepodporuje jsou ukládány hodnoty `Message-Id`.

Bez zadání parametru -n je volána metoda `DownloadAll()`. Tato metoda stáhne všechny zprávy ze serveru a uloží je do jednotlivých souborů v <out_dir>. Každý soubor je pojemnován jako: `"uživatelské jméno"+ '_' + "IP adresa serveru"+ '_' + "unikátní číslo v out_dir"`. Pokud tedy bude program opakovaně spuštěn bez parametru -n se stejným výstupním adresářem, budou všechny zprávy uloženy znovu a nedojde k přemazání dřívěji stažených zpráv.

Zároveň se stahováním a ukládáním zpráv je vytvořen objekt třídy `Database`. Ten ve stejné adresářové úrovni, kde je program spouštěn vytvoří adresář `Database`. Pro každého uživatele a server zde uloží soubor pojmenovaný jako: `"username"+ '_' +"ip"`. Do tohoto souboru jsou ukládány jednotlivé UID nebo Message-Id již stažených zpráv.

Pokud byl zadán parametr -d dojde ke smazání všech stažených zpráv ze serveru. O tuto funkcionalitu se stará metoda `DeleteAll()`.

3.4 Stahování pouze nových zpráv

Pri zadání parametru -n jsou stahovány pouze dosud nestažené zprávy. Před samotným procesem stahování zpráv je otevřen soubor z adresáře `Database`. Z něj jsou načteny informace o uživatelem dříve již stažených zprávách. Potom jsou ze serveru stahovány jednotlivé zprávy a ověřovány vůči této databázi. Pokud se v databázi nenachází UID/Message-Id staženého e-mailu, je do databáze přidáno a zpráva uložena do <out_dir>.

Stahování nových zpráv je nezávislé na zvolené výstupní složce. Nové zprávy je tak možné stahovat do jiných adresářů, než kde se nacházejí původně stažené zprávy. Pokud by tato funkcionalita měla být vázána na výstupní adresář, stačilo by přesunout umístění adresáře `Database` do <out_dir>.

Omezení:
Soubor v adresáři `Database` je vázán na IP adresu serveru. Server tedy během svého provozování nesmí změnit svoji IP adresu. Pokud server nepodporuje volitelný příkaz `UIDL`, může nastat problém s rozpoznáváním nových zpráv u těch, které by neobsahovaly položku `Message-Id`. (Odeslané zprávy by však dle doporučení v RFC [5] tuto položku obsahovat měly.)

Pokud byl zadán parametr -d, pak dojde ke smazání pouze nově stažených zpráv.

3.5 Použité knihovny

Při tvorbě programu byly použity následující knihovny jazyka C++:

<iostream>	<openssl>
<string>	<sys/socket.h>
<sys/stat.h>	<sys/types.h>
<stdlib.h>	<arpa/inet.h>
<fstream>	<netinet/in.h>
<sstream>	<regex>
<cstring>	<stdio.h>
<limits.h>	<vector>

4 Základní informace o programu

Program komunikuje s POP3 serverem, ze kterého stahuje a ukládá e-mailové zprávy. Se serverem je možné navázat šifrovanou komunikaci pomocí `TSL` nebo komunikaci nešifrovanou, která pokud to server podporuje může být přepnuta do šifrovaného režimu pomocí `STARTTLS`. Každá stažená zpráva je pojmenována jako `"uživatelské jméno"+ '_' + "IP adresa serveru"+ '_' + "unikátní číslo"` a uložena do výstupního adresáře. Ve výstupním adresáři tak nedochází k přepisování již stažených e-mailů. Pokud program ukončil svoji činnost úspěšně, informuje uživatele o počtu stažených zpráv a vrátí návratovou hodnotou 0. Došlo-li k chybě, je program ukončen s nenulovou návratovou hodnotou.

4.1 Návratové kódy

Popis návratových hodnot specifikovaných v souboru `error.h`.

Označení v kódu:	**Hodnota**:	**Popis**:
OK	0	vše v pořádku
ERR_BAD_PARAMS	1	špatné vstupní parametry programu
ERR_FILE_DIR_NOT_EXISTS	2	zadaný soubor nebo adresář neexistuje
ERR_BAD_AUTH_FILE	3	špatná struktura <auth_file>
ERR_CONNECTION	4	problém s připojením k serveru
ERR_SERVER_FAIL	5	problém na straně serveru
ERR_LOGIN	6	špatné přihlašovací údaje
ERR_SAVE_FILE	7	problém s vytvářením výstupních souborů

4.2 Metriky kódu

Počet souborů:	8
Počet řádků kódu:	1440
Velikost všech zdrojových kódu:	49.3 kB
Velikost spustitelného souboru:	233 kB

4.3 Převzaté metody

Metody `isDir()`, `isFile()`, `convertRelativePath()`, `fileOrDirExist()` a `testDirFile()` ze třídy **Pop3** jsou mým autoplagiátem. Metody jsem převzal ze svého prvního projektu do předmětu IPK.

Projekt: `Klient-Server aplikace pro přenos souborů pomocí RESTful API`. Školní rok: 2016/17. Autor: Petr Jůda (xjudap00), FIT VUT Brno.

5 Návod k použití

Pokud je program spuštěn bez zadání jakéhokoliv parametru, je zobrazena nápověda a program ukončen. Jinak se očekává zadání minimálně 3 povinných parametrů. Cesta k výstupnímu adresáři (`-o <out_dir>`) pro ukládání e-mailových zpráv. Cestu k souboru s přihlašovacími údaji (`-a <auth_file>`). Doménové jméno nebo IPv4 / IPv6 adresu POP3 serveru (`<server>`). Každý parametr je možné zadat v libovolném pořadí, ale pouze jednou. Dále je možné přidávat další volitelné parametry dle následujícího popisu.:

./popcl **<server> -a <auth_file> -o <out_dir>**
[-p <port>] [-T | -S [-c <certfile>] [-C <certaddr>]] [-d] [-n] [-h | –help]

Volitelné parametry:

-p <port>	- číslo portu (výchozí 995 pro TLS a 110 pro nezabezpečené/STARTTLS připojení [2])
-T	- TLS připojení (nelze kompinovat s -S)
-S	- STARTTLS připojení (nelze kombinovat s -T)
-c <certfile>	- cesta k cetifikátu [soubor]. (lze použít pouze s -S nebo -T)
-C <certaddr>	- cesta k adresáři s certifikáty (lze použít pouze s -S nebo -T)
-d	- smazání zpráv ze serveru
-n	- stažení pouze nových (dosud nestažených zpráv)
-h | –help	- výpis nápovědy

5.1 Praktická ukázka použití:

Stažení všech emailů ze serveru pop3.seznam.cz a jejich smazání ze serveru.
```
./popcl -d -a credentials -o ./out pop3.seznam.cz
```

Stažení pouze nových zpráv ze serveru 74.125.206.109 pomocí šifrovaného spojení TSL.
```
./popcl -a ./dir/credentials 74.125.206.109 -o outdir -n -T
```

Stažení zpráv ze serveru 2a02:598:a::78:46 běžícího na portu 1234 pomocí STARTTLS s definicí vlastní cesty k certifikátům.
```
./popcl 2a02:598:a::78:46 -p 1234 -a credentials -o /home/user/Desktop -S -C ./certDir
```

Reference

[1] Foundation, O. S. *Openssl - OpenSSL command line tool* [online]. [cit. 2017-10-23]. Dostupné na: <https://www.openssl.org/docs/man1.0.2/apps/openssl.html>.

[2] IANA. *Service Name and Transport Protocol Port Number Registry* [online]. Poslední změna 6. 10. 2017 [cit. 2017-10-23]. Dostupné na: <https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml>.

[3] Myers, J. a Rose, M. *Post Office Protocol - Version 3* [Internet Requests for Comments]. [b.m.]: RFC Editor, May 1996 [cit. 2017-10-23]. 1-22 s. RFC, 1939. Dostupné na: <https://tools.ietf.org/html/rfc1939>.

[4] Newman, C. *Using TLS with IMAP, POP3 and ACAP* [Internet Requests for Comments]. [b.m.]: RFC Editor, June 1999 [cit. 2017-10-23]. 1-15 s. RFC, 2595. Dostupné na: <https://tools.ietf.org/html/rfc2595>.

[5] P. Resnick, E. *Internet Message Format* [Internet Requests for Comments]. [b.m.]: RFC Editor, October 2008 [cit. 2017-10-23]. 1-57 s. RFC, 5322. Dostupné na: <https://tools.ietf.org/html/rfc5322>.

[6] Wikipedia. *Klient-Server* [online]. Poslední změna 4. 10. 2017 [cit. 2017-10-23]. Dostupné na: <https://cs.wikipedia.org/wiki/Klient-server>.