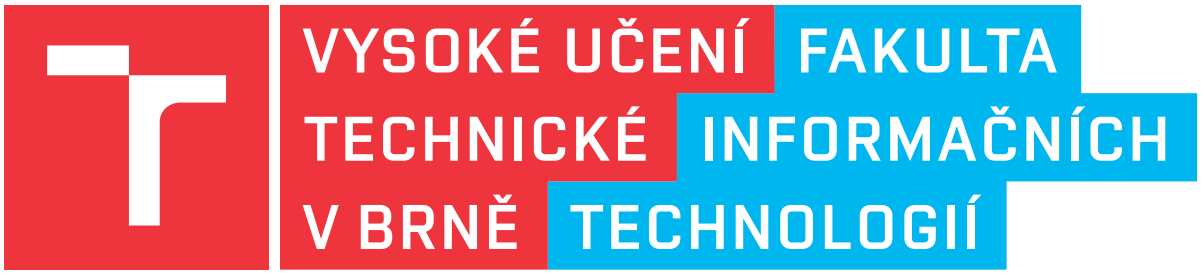

Síťové aplikace a správa sítí (ISA)

Klient POP3 s podporou TLS

v Olomouci
20. listopadu 2017

Václav Martinka
xmarti76

Obsah

1 Zadání

Napište program popcl, který bude umožňovat čtení elektronické pošty skrze protokol POP3 (RFC 1939 [1] s rozšířeními POP3S a POP3 STARTTLS - RFC 2595 [2]). Program může podporovat pouze autentizaci příkazy USER/PASS, příkaz APOP nemusíte podporovat.
Program po spuštění stáhne zprávy uložené na serveru a uloží je do zadaného adresáře (každou zprávu zvlášť). Na standardní výstup vypíše počet stažených zpráv. Pomocí dodatečných parametrů je možné funkcionalitu měnit.

2 Protokol POP3

2.1 Popis protokolu

Post Office Protokol neboli POP je internetový protokol používaný pro stahování e-mailových zpráv ze vzdáleného serveru na klienta. Jedná se o aplikační protokol pracující přes TCP/IP připojení. V současnosti je používána zejména třetí verze (POP3), která byla standardizována v roce 1996 v RFC 1939. POP3 má rezervovaný port 110.

Dále existuje rozšíření POP3S které používá pro zabezpečení TSL nebo SSL. POP3S má rezervovaný port 995. Mimo to existuje rozšířený příkaz STLS, který přepne klasické POP3 připojení do šifrovaného režimu.[3] [4]

Komunikaci zahajuje klient připojením na server. Server zašle uvítací zprávu a klient může zasílat jednotlivé příkazy. Ty jsou vždy ve tvaru 4 velkých písmen, následuje parametr příkazu, zakončuje se CRLF. Odpověď serveru začíná buď `+OK` v případě úspěchu nebo `-ERR` v případě chyby. Následuje samotná odpověď serveru zakončená buď CRLF u jednořádkových odpovědí nebo .CRLF dlouhých odpovědí. Pokud odpověď obsahuje tečku na novém řádku (`CRLF.`) je escapována na `CRLF..`

Protokol byl navržen pro snadné stažení a následné smazání zpráv ze serveru. To ovšem dnes většina uživatelů nevyžaduje a chtějí mít zprávy uložení jak v PC (mobilu) tak na serveru, proto byl vyvinut protokol IMAP.
Seznam základních příkazů: [1]

- `USER username` – Zaslání uživatelského jména, zpravidla první příkaz
- `PASS password` – Zaslání uživatelského hesla, následuje USER
- `STAT` – Žádá o počet zpráv a jejich celkovou velikost
- `LIST` – Odpovědí je seznam zpráv a jejich velikostí
- `RETR index` – Odpovědí je konkrétní
- `DELE index` – Smažu zprávu
- `QUIT` – Ukončí spojení a smaže zprávy označené příkazem `DELE`

2.2 Příklad jednoduché komunikace

S: *Server naslouchá na TCP portu*

C: *Otevření spojení*

S: `+OK POP3 server ready <1896.697170952@dbc.mtview.ca.us>`

C: `USER mrose`

S: `+OK User accepted`

C: `PASS mrosepass`

S: `+OK Pass accepted`

S: `+OK mrose's maildrop has 2 messages (320 octets)`

C: `STAT`

S: `+OK 2 320`

C: `LIST`

S:
```
+OK 2 messages (320 octets)
1 120
2 200
.
```

C: `RETR 1`

S:
```
+OK 120 octets
<POP3 server posílá 1. zprávu>
.
```

C: `DELE 1`

S: `+OK message 1 deleted`

C: `RETR 2`

S:
```
+OK 200 octets
<POP3 server posílá 2. zprávu>
.
```

C: `DELE 2`

S: `+OK message 2 deleted`

C: `QUIT`

C: `DELE 3`

S: `-ERR message 3 does not exist`

C: `QUIT`

S: `+OK dewey POP3 server signing off (maildrop empty)`

C: *Uzavření spojení*

S: *Server čeká na další spojení*

3 Spuštění aplikace

3.1 Použití

`popcl [-h|-help] <server> [-p <port>] [-T|-S [-c <certfile>] [-C <certaddr>]] [-d] [-n] -a <auth_file> -o <out_dir> [-b] [-v] [-q]`
Pořadí parametrů je libovolné.

3.2 Popis parametrů

- `-h, -help` Zobrazí nápovědu a ukončí program. Ostatní parametry jsou ignorovány. **volitelný**
- `<server>` Doménové jméno nebo IP adresa požadovaného zdroje. **povinný**
- `-p` Specifikuje číslo portu `<port>` na serveru. Není-li zadán, vybere se výchozí port pro danou službu [5]. **volitelný**
- `-T` Zapíná šifrování celé komunikace (pop3s), pokud není parametr uveden použije se nešifrovaná varianta protokolu. Nelze kombinovat s parametrem `-S`. **volitelný**
- `-S` Naváže nešifrované spojení se serverem a pomocí příkazu STLS (RFC 2595 [2]) přejde na šifrovanou variantu protokolu. Nelze kombinovat s parametrem `-T`. **volitelný**
- `-c` Definuje soubor `<certfile>` s certifikáty, který se použije pro ověření platnosti certifikátu SSL/TLS předloženého serverem. Lze použít jen s parametrem `-T`, nebo `-S`. **volitelný**
- `-C` Určuje adresář `<certaddr>`, ve kterém se mají vyhledávat certifikáty, které se použijí pro ověření platnosti certifikátu SSL/TLS předloženého serverem. Lze použít jen s parametrem `-T`, nebo `-S`. **volitelný**
- `-d` Po stažení zpráv dojde k jejich vymazání na serveru **volitelný**
- `-n` Stáhnou se pouze nové zprávy. V kombinaci s `-d` se mažou pouze nové zprávy. **volitelný**
- `-a` Určuje soubor `<auth_file>` pro autentizaci. **povinný**
- `-o` Určuje adresář `<out_dir>`, do kterého se budou stažené zprávy ukládat. **povinný**
- `-b` Zobrazí jednoduchý progress bar informující o průběhu stahování. **volitelný**
- `-v` Zapne zobrazování debugovacích výpisů, nevhodné kombinovat s `-b`. **volitelný**
- `-q` Nezobrazí zcela nic (mimo chybových výpisů na stderr), nevhodné kombinovat s `-b` a `-v`. **volitelný**

Pokud není uveden parametr `-c` ani `-C`, pak se použije úložiště certifikátů získané funkcí `SSL_CTX_set_default_verify_paths()`.

4 Implementace

4.1 Základní návrh

Na základní kostru programu byly použity programy z předmětu IPK. Program je napsán v jazyce *C++*. Je rozdělen do dvou tříd a funkcí. Nejdůležitější je třída `Client`, která obsahuje veškeré metody pro komunikaci se serverem. Druhou důležitou třídou je `Options`, která shlukuje nastavení získané z příkazové řádky. Tyto dvě třídy doplňuje hlavičkový soubor `functions.h` obsahující zejména funkci na parsování argumentů a několik dalších podpůrných funkcí.

Struktura programu je pak následující:

Vytvořit objekt `options` třídy *Options*

Funkcí `read_args` načíst argumenty do objektu `options`

Vytvořit objekt `client` třídy *Client*

Volat metody objektu `client`

`connect` Připojit se k serveru
`log_in` Poslat přihlašovací údaje
`list` Získat seznam zpráv na serveru
`statistics` Získat počet zpráv na serveru
`read_message` Získat konkrétní zprávu
`delete_message` Smazat konkrétní zprávu

Ukončit připojení metodou `disconnect_server`

Metody `read_message` and `delete_message` jsou volány ve smyčce.

4.2 Stahování pouze nových zpráv

Stahování pouze nových zpráv je řešeno pomocí pomocného souboru `last.ini` v kterém je uloženo `Message-ID` naposled stažené zprávy. Zprávy jsou stahovány ve smyčce od nejnovější po nejstarší. Mají-li se stahovat jen nejnovější, tak se u každé zprávy porovná její ID s tím ze souboru. V případě schody se stahování ukončí.

Potenciální problémy

Zpráva neobsahuje `Message-ID`: Jelikož tato položka v hlavičce je pouze doporučená, může se stát, že dorazí zpráva bez ID. V tom případě se porovnání neprovádí.

Soubor `last.ini` neexistuje: Pokud soubor neexistuje, zobrazí se varování a stáhnou se všechny zprávy.

Zpráva byla smazána na serveru: Protože se provádí porovnání jen na jednu konkrétní zprávu, tak v případě, že bude ze serveru smazána, dojde ke stažení všech zpráv. Řešením by bylo ukládat ID (nebo něco jiného, např hash) všech stažených zpráv, ale to by se v případě většího množství zpráv negativně podepsalo na výkonu a protože POP3 protokol na to ani nebyl navržen, rozhodl jsem se to neřešit.

4.3 Mazání zpráv

Zpráva je smazána až po jejím úspěšném stažení. Jedná se o jednoduchou ochranu zpráv. Z toho vychází vlastnost, že v případě kombinace stahování jen nových zpráv a mazání zpráv dojde ke smazání jen nových (právě stažených) zpráv.

4.4 Ukládání zpráv

Zprávy se ukládají do složky definované uživatelem. Pokud složka neexistuje, program na to upozorní a ukončí se. Jméno souboru obsahuje datum a čas odeslání a odesílatele. Původně místo odesílatele měl být titulek zprávy, ale je zde problém s UTF kódováním. Pokud hlavička emailu neobsahuje datum, čas nebo odesílatele, je použit náhradní.

Zprávy jsou ukládány s příponou `.imf`. V případě, že ve složce existuje již zpráva se stejným jménem, je ta nová doplněna o `_i`, kde `i` představuje pořadí nové zprávy.

4.5 Autentizační soubor

Parser autentizačního souboru je poměrně benevolentní. Je odolný vůči mezerám navíc a prázdným řádkům. Struktura souboru je:

`username =` *Uživatelské jméno*
`password =` *Heslo*

Do budoucna by bylo možné soubor rozšířit například o definici výstupní složky.

4.6 Rozšíření

Oproti zadání jsou přidány tři přepínače. Přepínač `-v` zapne debugovací výpisy. Přepínač `-b` zapne vykreslovaní jednoduchého progress baru informujícím o průběhu stahování. Přepínač `-q` vypne všechny výstupy na `stdout`.

4.7 Cizí zdrojové kódy

Většina kódu je přebrána z mých projektů do *IPK*. Nový je zejména `main.cpp`. Pro zabezpečenou komunikaci byli použity kousky kódů ze stránek *IBM* [6] a *stackoverflow* [7].
Komunikace byla založena na *RFC dokumentech* [1] a popisu ze stránek *wikipedie*. [3] [4]
IPv6 konektivita byla zprovozněna podle návodu ze stránek *electronicsFAQ*. [8]

Reference

[1] RFC. *Post Office Protocol - Version 3.* [b.m.]: RFC, 1996. Dostupné na: <https://tools.ietf.org/html/rfc1939>.

[2] RFC. *Using TLS with IMAP, POP3 and ACAP.* [b.m.]: RFC, 1999. Dostupné na: <https://tools.ietf.org/html/rfc2595>.

[3] *Post Office Protocol* [Wkipedia CZ]. Dostupné na: <https://cs.wikipedia.org/wiki/Post_Office_Protocc

[4] *Post Office Protocol* [Wkipedia EN]. Dostupné na: <https://en.wikipedia.org/wiki/Post_Office_Protocc

[5] *Service Name and Transport Protocol Port Number Registry.* November 16, 2017. Reference RFC6335. Dostupné na: <https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xht

[6] Ballard, K. *Secure programming with the OpenSSL API.* June 28, 2012. Dostupné na: <https://www.ibm.com/developerworks/library/l-openssl/>.

[7] *Turn a simple socket into an SSL socket* [stackoverflow]. 2016. Dostupné na: <https://stackoverflow.com/questions/7698488/turn-a-simple-socket-into-an-ssl-socket>.

[8] Chugh, A. *Simple TCP client server sockets application using IPv6 and IPv6.* December 28, 2012. Dostupné na: <http://www.electronicsfaq.com/2012/12/simple-tcp-client-server-sockets.html>.