Klient POP3 s podporou TLS

Sieťové aplikácie a správa sietí

Marek Németh

Obsah

1. Úvod

1.1. Zadanie projektu:

Napište program popcl, který bude umožňovat čtení elektronické pošty skrze protokol POP3 (RFC 1939 s rozšířeními pop3s a POP3 STARTTLS - RFC 2595). 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.

1.2. Spustenie aplikácie

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

Pořadí parametrů je libovolné. Popis parametrů:

- Povinně je uveden název <*server*> (IP adresa, nebo doménové jméno) požadovaného zdroje.
- Volitelný parametr -*p* specifikuje číslo portu <*port*> na serveru. Zvolte vhodnou výchozí hodnotu v závislosti na specifikaci parametru -T a číslech portů registrovaných organizací IANA.
- Parametr -*T* zapíná šifrování celé komunikace (pop3s), pokud není parametr uveden použije se nešifrovaná varianta protokolu.
- Parametr -S naváže nešifrované spojení se serverem a pomocí příkazu STLS (RFC 2595) přejde na šifrovanou variantu protokolu.
- Volitelný parametr -*c* definuje soubor <*certfile*> s certifikáty, který se použije pro ověření platnosti certifikátu SSL/TLS předloženého serverem (použití jen s parametrem -*T*, nebo -*S*).
- Volitelný parametr -*C* určuje adresář <*certaddr*>, ve kterém se mají vyhledávat certifikáty, které se použijí pro ověření platnosti certifikátu SSL/TLS předloženého serverem. (Použití jen s parametrem -*T*, nebo -*S*.)
- Pokud není uveden parametr -*c* ani -*C*, pak použijte úložiště certifikátů získané funkcí *SSL_CTX_set_default_verify_paths()*.
- Při použití parametru -*d* se zašle serveru příkaz pro smazání zpráv.
- Při použití parametru -*n* se bude pracovat (číst) pouze s novými zprávami. Zamyslete se nad možným zjištěním novosti zpráv. Zvolený přístup a případné nedostatky popište v dokumentaci.
- Povinný parametr -*a* <*auth_file*> vynucuje autentizaci (příkaz USER), obsah konfiguračního souboru <auth_file> je zobrazený níže.
- Povinný parametr -*o* <*out_dir*> specifikuje výstupní adresář <*out_dir*>, do kterého má program stažené zprávy uložit.

2. Systém

2.1. Návrh

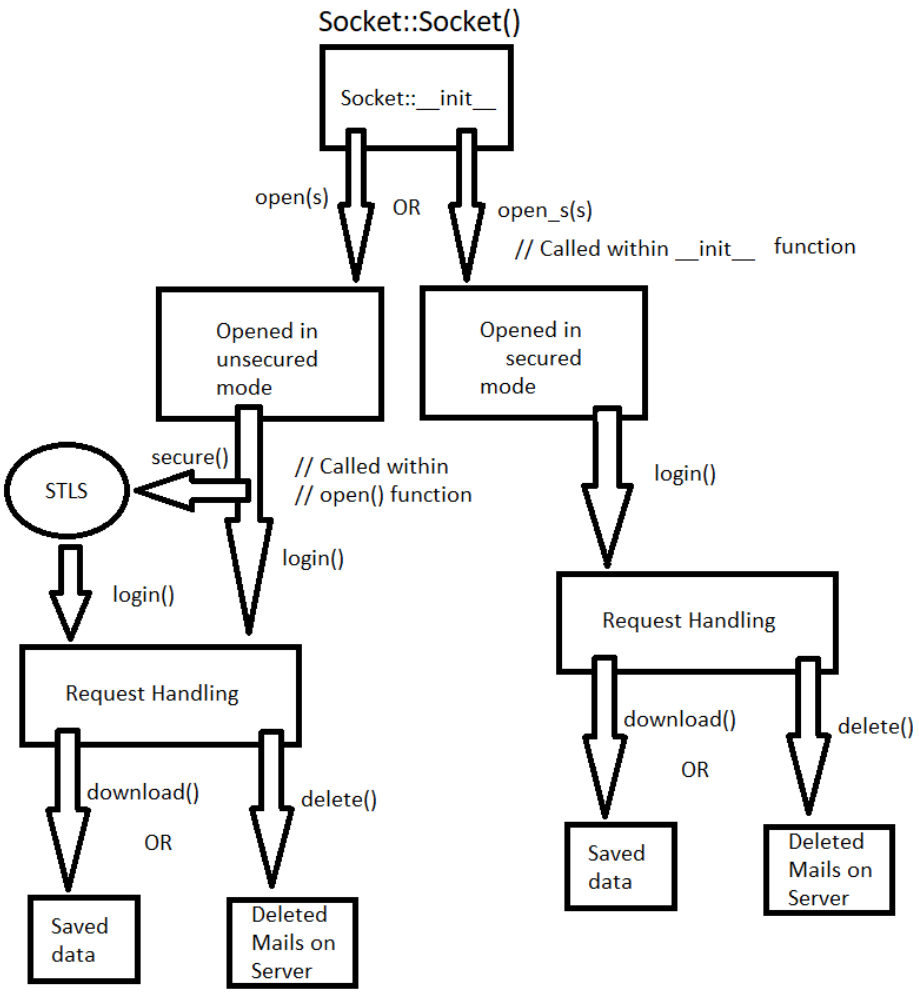
Program je implementovaný v jazyku C++ vzhľadom na jednoduchšiu prácu s reťazcami.

Princíp systému je jednoduchý a spočíva v jednotlivých krokoch:

1. Spracovanie argumentov
2. Vytvorenie inštancie objektu Socket
3. Vykonanie požadovaného úkonu na základe argumentov

Celý system využíva triedu Socket. Vykonanie úkonov nastáva bezprostredne po vytvorení inštancie objektu.

Grafické znázornenie automatu:



2.2 Implementácia

Trieda Socket obsahuje nasledujúce funkcie ktoré sa vykonajú v poradí zobrazenom na automate v sekcii 2.1.

Pri vytvorení inštancie objektu Socket sa uložia základné parametre pre prácu so serverom.

Na základe poskytnutých parametrov sa potom otvára zabezpečená komunikácia(parameter T) alebo nezabezpečená komunikácia ktorá sa po otvorení spojenia môže(parameter S) zmeniť na zabezpečenú pomocou requestu STLS na server

Komunikácia sa otvára pri inicializácii objektu pomocou funkcií open alebo open_s.

V tele týchto funkcií sa vykonávajú nasledujúce operácie:

1. Vytvorí sa spojenie so serverom prostredníctvom objektu BIO [1]
2. Pomocou funkcii receive a send_m sa komunikuje so serverom nasledovným spôsobom:
   - send_m zasiela požiadavky na server napr. STLS, USER, PASS, RETR, STAT...[2]
   - receive príjma odpoveď od serveru a kontroluje jej obsah (+OK,-ERR)
3. Vykoná sa operácia na stiahnutie alebo zmazanie správ ktorá sa zavolá v tele funkcie open/open_s:
   - funkcia delete_m():
     Získa od serveru počet správ pomocu požiadavky STAT a počet si uloží. Následne preiteruje každú správu a pomocou požiadavky DELE ju vymaže zo serveru.
   - funkcia download():
     Získa od serveru počet správ pomocou požiadavky STAT a počet si uloží. Následne preiteruje každú správu pomocou požiadavky RETR, získa z nej Message-ID pomocou funkcie get_msg_id(). Ak sa vo výstupnom adresári takáto správa už nachádza, pokračuje ďalej. Ak nie, túto správu si vyžiada znova a jej obsah zapíše do súboru s názvom jej Message-ID.
4. Ukončí sa komunikácia so serverom pomocou požiadavky QUIT na zachovanie zmien

V triede Socket sa nachádza niekoľko menších pomocných funkcií:

1. login()

   Pošle serveru požiadavky na autentizáciu užívateľa:

   USER username

   PASS password

2. secure()

   Pošle požiadavku STLS, a zabezpečí spojenie pomocou TLS

3. receive_num()
   Funkcia takmer rovnaká ako receive, s rozdielom že táto funkcia vracia číslo
   Používa sa pri zistení počtu správ po požiadavke STAT
4. Get_msg_id()
   Pošle požiadavku RETR a z tela správy získa pomocou regex_search(knižnica regex)
   Message-ID ktoré vráti
5. get_mail()
   Podobná funkcii get_msg_id, táto funkcia telo správy uloží do súboru s názvom jej ID.
6. resolve_auth()
   Funkcia na získanie autentizačných údajov zo súboru ktorý je poskytnutý pri spustení programu

3. Referencie

[1] https://developer.ibm.com/tutorials/l-openssl/

[2] https://www.suburbancomputer.com/tips_email.htm