VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

Klient POP3 s podporou TLS

ISA - Síťové aplikace a správa sítí

28. října 2021

Tomáš Tomala (xtomal02)

Obsah

1 Uvedení do problematiky

Cílem tohoto projektu je implementovat klienta pro POP3 s podporou TLS. POP3 je internetový protokol pro přístup do e-mailové schránky a operace s ní, nejčastěji však pro příjem pošty. Pro přístup do e-mailové schránky potřebujeme dvě základní komponenty, a to klienta a server. Server slouží jako e-mailová schránka a reaguje na požadavky stažení od klienta.[1] Specifickou vlastností POP3 protokolu je stahování zpráv. Narozdíl od protokolu IMAP nepracuje se zprávou přímo na serveru, ale zprávu si lokálně stáhne a v mnoha případech rovnou smaže zprávu z e-mailového serveru. POP3 umožňuje nezabezpečený přenos, zabezpečení pomocí SSL nebo TLS. Všechny tyto 3 případy jsou součástí tohoto projektu. Protokol POP3 je výhodný pro uživatele, kteří mají časově omezené nebo nestálé připojení k Internetu.[5] Nevýhodou tohoto protokolu je neschopnost přistoupit ke stejným zprávám z více zařízení, pokud se zprávy po prvním stažení smažou.

2 Návrh aplikace

Jak je již v předchozím textu uvedeno, aplikace bude klient, který bude z libovolného serveru s protokolem POP3 získávat zprávy. Klient má možnost stáhnout všechny zprávy ze schránky, nebo pouze nové zprávy, které ještě na toto zařízení nestahoval. Aplikace obsahuje možnost smazat zprávy, a to všechny, nebo pouze nové zprávy, v závislosti na kombinaci požadavku stahování pouze nových nebo všech zpráv. Aplikace poskytuje možnost nezabezpečeného spojení, zabezpečení pomocí SSL nebo pomocí modernějšího TLS.

3 Popis implementace

Jako implementační jazyk byl vybrán C++. Program je rozdělen na dvě dílčí úlohy, které oddělují jednotlivé fáze, kterým program prochází.

3.1 Zpracování parametrů

První dílčí úloha zajišťuje získání parametrů. Tato úloha je vypracována v souboru `args.cpp` a spustí se jako první po spuštění programu uživatelem. Úloha vezme uživatelem zadané parametry z příkazové linky a každý parametr zpracuje. Při zpracování jednotlivých parametrů ukládá informace do struktury `arguments`, která se poté posouvá do další úlohy.

Úloha kontroluje duplicitu parametrů a správnost zadaných informací. Pokud je nějaký parametr vyhodnocen jako nesprávný, je program ukončen s chybovou hláškou.

Číslo specifikovaného portu musí být číslo v rozmezí 0-65535, jinak je číslo portu vyhodnoceno jak nesprávné. Pokud číslo portu neni zadáno, je použit port 995 pro zabezpečení SSL, jinak port 110.[1] Program neumožňuje kombinaci přepínače pro SSL a TLS zároveň.

Při specifikování souborů nebo složky probíhá kontrola existence. Program nedovolí uživateli specifikovat neexistující cestu pro výstupní složku, aby se předešlo nechtěnému vytvoření nové složky.

Při získání neznámého parametru je zkontrolován předchozí parametr. Pokud aktualní parametr nepatří k předchozímu, je tento parametr vyhodnocen jako adresa serveru.

Po získání parametrů se získají informace ze souboru pro authorizaci uživatele, který byl zadán pomocí prepínače `-a`. Program hledá v souboru pomocí regulárního výrazu na prvních dvou řádcích údaje o uživatelském jméně a heslu ve tvaru `password/username = <string>`. Údaje o uživatelském jméně a heslu se předají pomocí struktury authInfo.

Pokud program nemá dostatek informací z příkazové linky, program je predčasně ukončen s chybovou hláškou.

Parametry		
Specifikace	**Volitelný**	**Význam**
<server>	Ne	Adresa POP3 serveru
-p <port>	Ano	Port pro připojení ke serveru
-T	Ano	Šifrování celé komunikace pomocí SSL
-S	Ano	Šifrování komunikace pomocí TLS
-c <certfile>	Ano	Soubor s certifikáty pro šifrování
-C <certdir>	Ano	Složka, kde se mají vyhledávat certifikáty
-d	Ano	Smazání získaných zpráv
-n	Ano	Získání pouze nových zpráv
-a <authfile>	Ne	Soubor s údaji uživatele
-o <outdir>	Ne	Složka pro ukládání zpráv

3.2 Komunikace

3.2.1 Navázání komunikace

Následující dílčí úloha se stará o samotnou komunikaci se serverem. Komunikace začíná připojením na server metodou `connectToServer`. Metoda pomocí údajů ze struktury `arguments` vytvoří adresu serveru i s portem. Pro připojení nezabezpečené i zabezpečené se používají objekty BIO z knihovny OpenSSL. [2]

Pokud je vyžadováno zabezpečení SSL, před připojením se provede konfigurace SSL připojení pomocí metody `initSecureComm`. To zahrnuje přidáni hlaviček pro SSL spojení, vytvoření SSLCTX struktury pro informace o SSL, získání a ověření certifikátů a provázání BIO objektu s SSL ukazatelem. Poté se již vytvoří a inicializuje samotné připojení na server. [2]

Po zkontrolování připojení se v případě zabezpečeného připojení také zkontroluje validita certifikátů. Pokud je vše v pořádku, zkontroluje se odpověď.

3.2.2 Kontrola odpovědi

Kontrola odpovědi probíhá dle formátu specifikovaného ve formátu RFC 1939 [3] v metodě `checkResponse`. Pomocí regulárního výrazu se zkontroluje, zda začátek odpovědi obsahuje `+OK` nebo `-ERR`. Metoda vrací false hodnotu pro jinou hodnuto jako +OK.

3.2.3 TLS

V případě zabezpečené komunikace pomocí TLS musíme po úspěšném připojení na server ve fázi autorizace zaslat příkaz STLS. Po přijetí příkazu STLS nastane podobně jako u připojení SSL inicializace SSL struktur a OpenSSL knihoven. Pro připojení vytvoříme nový BIO objekt s SSL konfigurací, který poté svážeme s BIO objektem, který byl použit pro navázání standartního nezabezpečeného připojení. Po provedení handshake zkontrolujeme platnost certifikátů a pokračujeme autorizací.

3.2.4 Autorizace

Po úspěšném připojení nastává fáze autorizace. Pro autorizaci se použije struktura authInfo, vytvořená při zpracování parametrů. Jako první klient pošle příkaz `USER` s uživatelským jménem. Pokud server pošle kladnou odpověď, klient pokračuje s příkazem PASS a heslem.

Každý příkaz klienta je ukončen CRLF. Pokud je heslo serverem přijato, je ukončena autorizační fáze a pokračuje se fází transakce.[3]

3.2.5 Transakce

Program se ve fázi transakce první příkazem `STAT` zeptá na stav e-mailové schránky. Díky tomuto příkazu se zjistí počet zpráv ve schránce.

Pokud schránka neobsahuje žádné zprávy, program se ukončí a tuto skutečnost oznámi uživateli hláškou. Pokud existuje alespoň jedna zpráva ve schránce, začnou se stahovat zprávy pomocí příkazu `RETR`. Pomocí BIO objektu se přes metodu `BIO_read` čte odpověď, dokud nenarazí na terminátor zprávy `CRLF.CRLF`. Po čtení zprávy se zkontroluje návratový kód, který se také oddělá ze samotné zprávy. Společne s koncem zprávy, což je `.CRLF`[4], se oddělá i bytestuffing. Bytestuffing nastává při víceřádkové odpovědi, kde řádek začíná tečkou.[3] Pro nezaměnitelnost s terminátorem zprávy se přidává další tečka, kterou je pro správnost zprávy třeba odstranit.

Po odstranění nadbytečných věcí ze zprávy se zjistí MessageID. MessageID je jedinečný identifikátor zprávy, pomocí kterého se zjišťuje novota zprávy. Seznam stažených zpráv se udržuje ve skrytém souboru `.downloaded`. Tento soubor je uložen na místě programu. Nevýhoda tohoto přístupu je vytváření dalšího souboru, ve kterém navíc uchováváme i identifikátory zpráv, ktere již byly smazány. Jiný přístup je ukládání MessageID do jména souboru, do kterého se zpráva uloží. Nevýhodou tohoto přístupu je nesrozumitelný název zprávy, pomocí kterého nelze rozeznat, o jakou zprávu se jedná. V souboru již stažených zpráv jsou jednotlivé identifikátory ve formátu `messageid : <messageid>`. Na každý identifikátor připadá vlastní řádek.

Podle parametrů spuštění a podle indikátoru novoty určíme, zda zprávu stáhnout a poté i smazat. Stažená zpráva se uloží do souboru s předmetem zprávy. Pokud již existuje taková zpráva, doplní se před název číselné odlišení od originální zprávy.

3.2.6 Ukončení komunikace

Po stažení a zpracování všech zpráv, které splňují vstupní parametry, se přechází do fáze ukončení spojení. Tato fáze je důležitá, aby server mohl přejít do fáze `UPDATE`[3], ve které plní požadavky na smazání zpráv. Pro vstup do této fáze se pošle serveru příkaz `QUIT`. Po ukončení komunikace se uvolní BIO struktura, vypíše se počet stažených zpráv a program je ukončen.

4 Návod na použití

Před spuštěním programu je potřeba zkompilovat zdrojový kód. Překlad se vykoná pomocí Makefile příkazem `make`. Je poskytnut také příkaz `clean`, který odstraní přeložený program a skrytý soubor obsahující informace o stažených zprávách. Po přeložení vznikne soubor `popcl`.

Program se spustí pomocí příkazu `./popcl <par>`, kde par jsou parametry. Seznam parametrů je popsán v kapitole Zpracování parametrů. Pro spuštění aplikace je potřeba specifikovat alespoň adresu POP3 serveru, soubor s údaji uživatele a výstupní složku. Pořadí parametrů je libovolné.

Příklad spuštění programu: `popcl 10.10.10.1 -p 1234 -T -n -o maildir -a cred`

Literatura

[1] Awati, R.: POP3 (Post Office Protocol 3). online, naštíveno 24. 10. 2021.
URL https://whatis.techtarget.com/definition/POP3-Post-Office-Protocol-3

[2] Ballard, K.: Secure programming with the OpenSSL API. online, naštíveno 21. 10. 2021.
URL https://developer.ibm.com/tutorials/l-openssl/

[3] J. Myers, M. R., Carnegie Mellon: Post Office Protocol - Version 3. online, naštíveno 20. 10. 2021.
URL https://datatracker.ietf.org/doc/html/rfc1939

[4] Resnick, P.: Internet Message Format. online, naštíveno 23. 10. 2021.
URL https://datatracker.ietf.org/doc/html/rfc5322

[5] S.L, S. S.: Email Protocols - POP3, SMTP and IMAP Tutorial. online, naštíveno 26. 10. 2021.
URL https://www.siteground.com/tutorials/email/protocols-pop3-smtp-imap/