VYSOKÉ UČENÍ TECHNICKÉ v BRNĚ
Fakulta Informačních technologií

Dokumentácia k projektu do predmetu ISA

POP3 server

Autor:
Marek Schauer xschau00

Obsah

Úvod

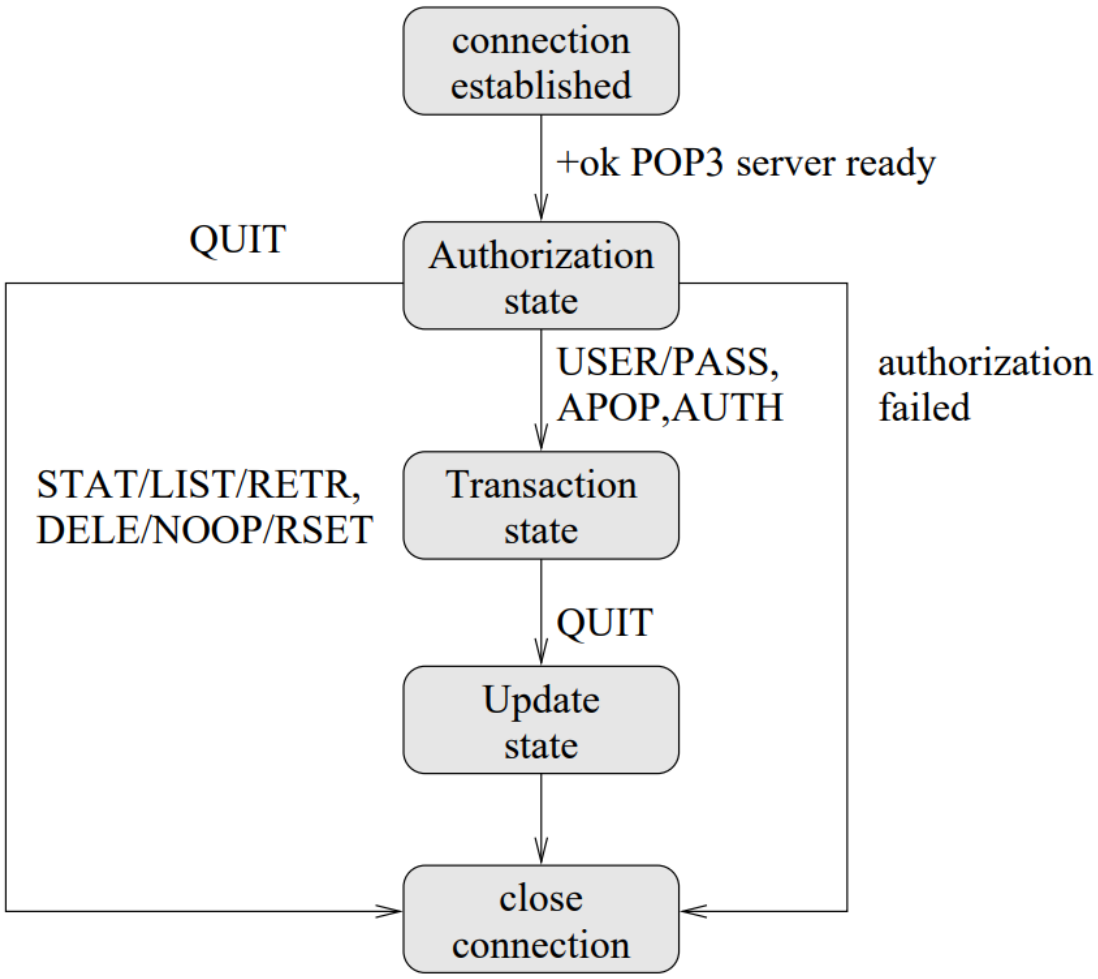
Táto dokumentácia pojednáva o procese implementácie konkurentného POP3 servera. Úlohou POP3 servera je poskytnúť rozhranie pre prijímanie elektronickej pošty podľa protokolu POP3 definovaného v RFC 1939 (J. Myers, M. Rose, 1996).

Server pracuje s emailmi uloženými vo formáte IMF (P. Resnick, Ed., 2008), ktoré sú uložené v adresárovej štruktúre typu Maildir (Bernstein).

Tento dokument je rozdelený do viacerých logických celkov, ktoré sa z rôznych uhľov pohľadu pozerajú na fungovanie a na proces implementácie POP3 servera, od teoretickej časti popisujúcej základný princíp fungovania protokolu POP3 až po praktickú časť popisujúcu náš spôsob implementácie konkurentného servera pracujúceho v súlade s protokolom POP3 definovaného v RFC 1939.

Protokol POP3

Protokol POP3 je aplikačný protokol pre prijímanie elektronickej pošty, ktorý využíva transportný protokol TCP. Spravidla funguje na porte 110, v nami vytvorenom serveri je číslo portu definovateľné pri jeho spúšťaní. Činnosť protokolu je popísaná stavovým automatom na Obrázku 1.



Obrázok 1, Popis činnosti protokolu POP3 stavovým automatom

Popis a spustenie programu

Program slúži ako server pre sťahovanie elektronickej pošty pomocou protokolu POP3. Program je implementovaný pre operačný systém CentOS Linux verzie 7.4.1708.

Preloženie súboru je možné zadaním príkazu `make`.

Preložený program je možné spustiť nasledovne:

```
./popser [-h] [-a PATH] [-c] [-p PORT] [-d PATH] [-r]
```

Parametre programu majú nasledovný význam:

- -h (help) - voliteľný parameter, pri jeho zadaní sa vypíše nápoveda a program sa ukončí
- -a (auth file) - cesta k súboru s prihlasovacími údajmi
- -c (clear pass) - voliteľný parameter, pri zadaní server akceptuje autentizačnú metódu, ktorá prenáša heslo v nešifrovanej podobe (inak prijíma iba heslá v šifrovanej podobe - hash)
- -p (port) - číslo portu na ktorom bude bežať server
- -d (directory) - cesta do zložky Maildir (napr. ~/Maildir/)
- -r (reset) - voliteľný parameter, server vymaže všetky svoje pomocné súbory a emaily z Maildir adresárovej štruktúry vráti do stavu, ako keby proces popser nebol nikdy spustený (netýka sa časových pečiatok, iba názvov a umiestnení súborov)

Autorizačný súbor musí byť v tvare

```
username = meno
password = heslo
```

Pre očakávané chovanie programu musia byť riadky ukončené znakom ‚\n' (ASCII hodnota znaku ‚\n' je 10).

Implementácia programu

Zdrojový kód nami vytvoreného servera je vytvorený v jazyku C++. Zdrojový kód sa nachádza v dvoch súboroch,

- `main.cpp` a
- `md5.h`.

Program využíva nasledujúce knižnice:

- `stdio.h`,
- `stdlib.h`,
- `string.h`,
- `iostream`,
- `fstream`,
- `unistd.h`,
- `sys/stat.h`,
- `netdb.h`,
- `err.h`,
- `ctype.h`,
- `sys/types.h`,
- `sys/socket.h`,

- `netinet/in.h`,
- `arpa/inet.h`,
- `ctime`,
- `fcntl.h`,
- `cstring`,
- `vector`,
- `dirent.h`,
- `mutex`,
- `map`,
- `csignal`,
- `errno.h`

Logika programu

Program je navrhnutý tak, aby fungoval ako konkurentný server pracujúci s neblokujúcimi soketmi. Vstupným bodom programu je funkcia `main()`, ktorá zabezpečuje vytvorenie schránky (soketu) a jej následné pripojenie na lokálny port. Ďalej je funkciou `listen()` vytvorená fronta prichádzajúcich požiadaviek a následne je funkciou `accept()` prijatá každá požiadavka, ktorá následne vyvolá vytvorenie novej schránky.

Po prijatí novej požiadavky sa prejde k vytvoreniu nového vlákna, ktoré je zodpovedné za komunikáciu s jedným konkrétnym klientom nášho servera a za spracovanie všetkých jeho požiadaviek. Pred vytvorením samotného vlákna sa naplní štruktúra `threadArgs`, ktorá v sebe nesie všetky informácie potrebné k správnemu fungovaniu daného vlákna. Táto štruktúra v sebe nesie nasledujúce informácie:

- Identifikátor schránky (soketu), s ktorou bude nové vlákno pracovať,
- parametre programu,
- vstupný stav konečného automatu implementujúceho protokol POP3 (táto položka štruktúry závisí od toho, či bol program spustený s parametrom -c alebo nie) a
- časová pečiatka, ktorá sa v danom vlákne pošle ako časť prvej správy poslanej klientovi.

Ukazateľ na túto štruktúru je do nového vlákna odovzdaný ako posledný argument funkcie `pthread_create()`, ktorá nové vlákno vytvára.

Po zavolaní funkcie `pthread_create()` je vytvorené nové vlákno obsluhujúce jedného konkrétneho klienta servera.

Nasledujúca časť dokumentácie popisuje kľúčové časti komunikácie s klientom POP3 servera.

Komunikácia s klientom POP3

Po spustení nového vlákna dané vlákno pošle klientovi uvítaciu správu vo formáte, ktorý je mierne v rozpore s RFC 1939. V RFC 1939 sa píše, že súčasťou uvítacej správy musí byť reťazec vo formáte

`<process-ID.clock@hostname>`,

kde

- process-ID je identifikátor procesu,
- clock je hodnota systémového času a
- hostname je doménové meno serveru.

Špecifikácia RFC 1939 vyžaduje, aby bol tento reťazec pre každého klienta jedinečný. Zadanie nášho projektu ale obmedzuje vytváranie nových procesov, ktoré je pre vytvorenie jedinečného reťazca v tomto formáte nevyhnutné. V dôsledku toho bude hodnota process-ID tohto reťazca vždy rovnaká a v prípade, že sa k serveru pripoja dvaja klienti v rovnakej sekunde, bude im pridelený zhodný reťazec.

Po odoslaní uvítacej správy je zahájené čakanie na prijatie správy od klienta. Toto čakanie je vykonávané pomocou funkcie `select()`, ktorá umožňuje nastavenie časového limitu, po ktorého uplynutí vráti číslo 0. Časový limit je v našom prípade nastavený na 10 minút. V prípade, že na server príde pred uplynutím časového limitu od daného klienta správa, funkcia `select()` vráti kladnú číselnú hodnotu a pridá zodpovedajúci soket do množiny soketov, ktoré sú pripravené na čítanie.

V prípade, že funkcia `select()` vráti nulu, detekujeme prekročenie časového limitu neaktivity klienta, v dôsledku čoho je klient následne od servera odpojený a dané vlákno je zrušené.

V prípade, že funkcia `select()` vráti kladnú hodnotu a pridala daný soket do množiny soketov, ktoré sú pripravené na čítanie, je celá daná správa prečítaná pomocou funkcie `recv()`.

Správa je následne spracovaná funkciou `process_message()`, ktorej sa venuje ďalšia časť.

Spracovanie prijatej správy

Prijatá správa je spracovaná funkciou `process_message()`. Táto funkcia je zavolaná po každom prijatí novej správy a je zodpovedná za správnu interpretáciu príkazu poslaného v správe, za správnu reakciu na tento príkaz a taktiež aj za vrátenie príslušnej odpovede na danú správu.

Vyhodnotenie prijatej správy

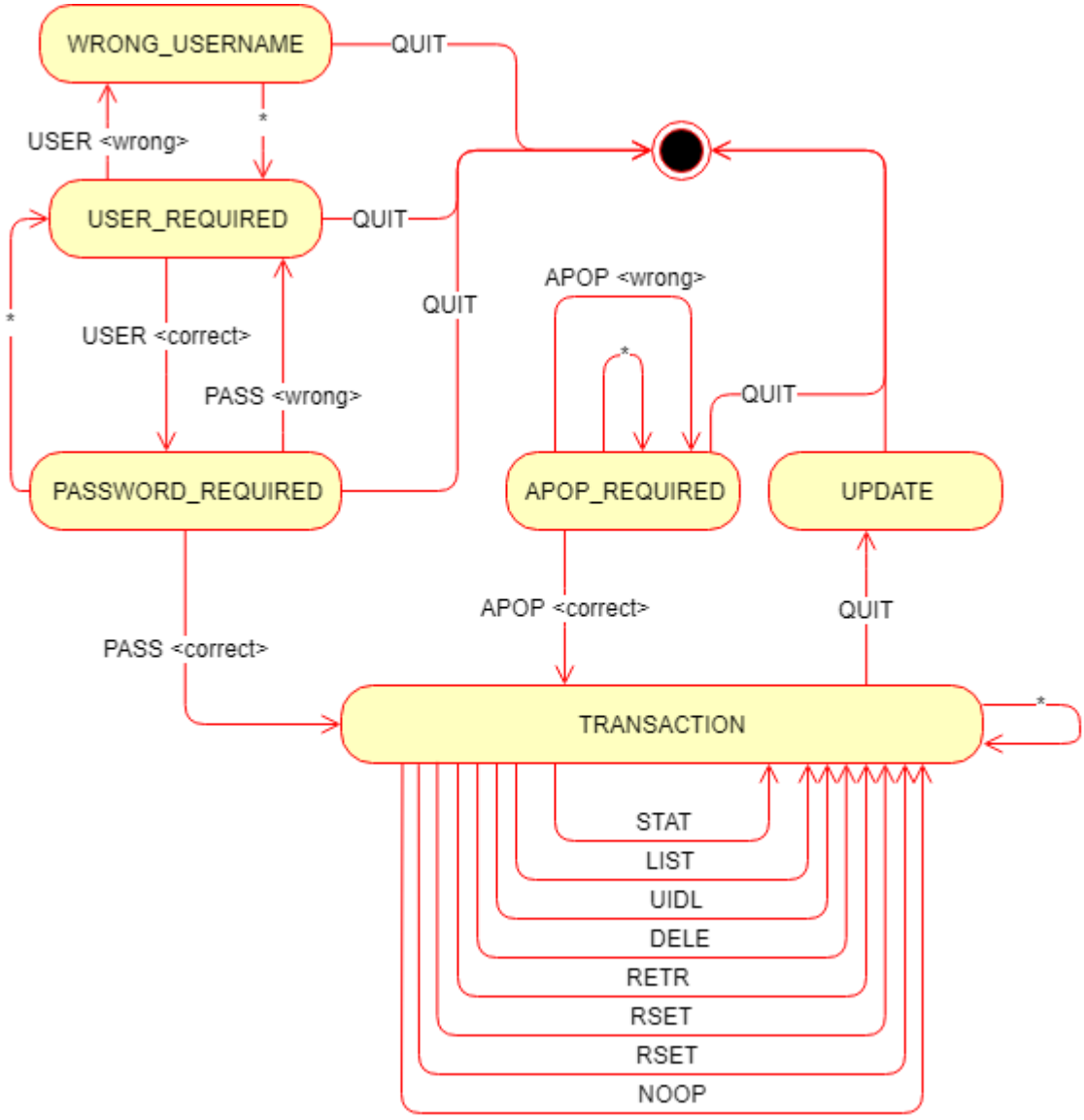
Za správne vyhodnotenie správy je zodpovedná funkcia `getCommand()`. Táto funkcia sa stará o to, aby sa z reťazca nesúceho správu od klienta stal príkaz s príslušnými argumentmi reprezentovaný štruktúrou `Command`. Štruktúra `Command` obsahuje meno príkazu a hodnoty 2 argumentov. V prípade, že správa od klienta obsahuje príkaz, ktorý nemá argumenty, sú položky tejto štruktúry reprezentujúce argumenty prázdne.

Reakcia a zostavenie odpovede na prijatú správu

Funkcia `process_message()` ďalej implementuje konečný stavový automat naznačený na Obrázku 2. Aktuálny stav konečného automatu je súčasťou štruktúry obaľujúcej informácie o danom vlákne predanej do tejto funkcie ako ukazateľ. Konečný automat má 6 stavov:

- USER_REQUIRED,
- WRONG_USERNAME,
- PASSWORD_REQUIRED,
- APOP_REQUIRED,
- TRANSACTION a
- UPDATE.

Prípustné príkazy v každom stave konečného automatu sú naznačené na obrázku.



Obrázok 2, Konečný stavový automat znázorňujúci fungovanie nami implementovaného POP3 servera

V prípade, že je parametrom -c zapnutá podpora prenosu hesla v nešifrovanej podobe, je počiatočný stav konečného automatu USER_REQUIRED, v opačnom prípade je to APOP_REQUIRED.

Stav WRONG_USERNAME je implementovaný kvôli tomu, aby sme prípadnému útočníkovi neprezrádzali, či uhádol správne meno užívateľa zasielané ako argument príkazu USER. Príkaz USER vždy vráti odpoveď „+OK“, pričom na pozadí servera môžu nastať dve situácie:

1. **Prípad, že užívateľ zadal zlé užívateľské meno**

   Server už teraz vie, že prihlásenie bude neúspešné a očakáva príkaz PASS, hoci k vyhodnoteniu správnosti hesla už nedôjde. Konečný automat sa presúva do stavu WRONG_USERNAME

2. **Prípad, že užívateľ zadal správne užívateľské meno**

   Server vyhodnotil, že klient zadal správne užívateľské meno a presunie sa do stavu PASSWORD_REQUIRED, v ktorom po nasledujúcej správe klienta kontroluje správnosť zadaného hesla.

Úspešné prihlásenie užívateľa je sprevádzané volaním funkcie `userAuthenticated()`, ktorá pripraví všetko potrebné pre prácu s zložkou Maildir. Kľúčovou časťou tejto funkcie je kontrolovanie, či už so zložkou Maildir nepracuje iné vlákno. Výlučný prístup k zložke Maildir je zabezpečený pomocou mutexu, ktorý je uzamknutý počas celej doby práce jedného klienta so zložkou Maildir až do momentu, keď klient odošle príkaz QUIT alebo klient zruší spojenie. V prípade, že je mutex uzamknutý, klient sa k zložke Maildir nedostane a je odhlásený. Spojenie s klientom nie je zrušené, klient sa v rámci konečného automatu ocitá v stave USER_REQUIRED.

Po spracovaní prijatej správy je v príslušnom stave konečného automatu vykonaná reakcia na daný príkaz a následne je zostavená príslušná odpoveď, ktorá je vrátená volajúcemu tejto funkcie ako štandardný reťazec.

Zaslanie odpovede klientovi

Zaslanie odpovede vrátenej funkciou `process_message()` je vykonávané pomocou funkcie `write()`. V prípade, že užívateľova ostatná správa bola „QUIT“ a odpoveď vrátená funkciou `process_message()` je reťazec „+OK“, je daná odpoveď odoslaná klientovi, následne server klienta odpojí a dané vlákno sa ukončí.

Ukončenie servera

Po tom, ako server prijme signál SIGINT je zavolaná funkcia `signalHandler()` zodpovedná za správne ukončenie servera. Táto funkcia vykonáva nasledujúce akcie (v príslušnom poradí):

- Odpojí všetkých klientov, ktorí sú na server pripojení,
- uzavrie všetky bežiace vlákna, ktoré boli serverom vytvorené,
- odomkne zámok, ktorý môže byť zamknutý,
- uvoľní pamäť, ktorá je vyhradená pre štruktúry nesúce informácie o bežiacich vláknach,
- ukončí program s návratovou hodnotou 0.

V prípade, že je pri spustení programu zadaný parameter `-r`, tesne pred ukončením programu je zavolaná funkcia `makeReset()`, ktorá je popísaná v ďalšej podkapitole.

Vrátenie stavu servera do pôvodného stavu

Vrátenie servera do stavu, ako keby nebol nikdy spustený, je implementované vo funkcii `makeReset()`. Hlavnou činnosťou tejto funkcie je presun súborov z adresára cur/ do adresára new/ v rámci zložky Maildir zadanej ako parameter programu.

Táto funkcia pracuje s pomocným súborom `log.txt`, do ktorého server počas svojho behu ukladá všetky informácie potrebné pre presun súborov medzi zložkami `cur/` a `new/`.

Štruktúra pomocného súbora

Pomocný súbor je obyčajný textový súbor, do ktorého je zapisované pri akejkoľvek zmene niektorého zo súborov v rámci zložky Maildir. Pri presune súboru reprezentujúceho daný email v rámci zložky Maildir je do pomocného súboru zapísaná štvorica riadkov reprezentujúcich nasledujúce informácie:

1. Názov súboru v rámci adresára Maildir,
2. veľkost daného súboru,
3. relatívna cesta od miesta spustenia servera k danému súboru v rámci zložky `Maildir/new/` a
4. relatívna cesta od miesta spustenie servera k danému súboru v rámci zložky `Maildir/cur/`.

V prípade, že niektorý zo súborov reprezentujúcich email je vymazaný, informácia o tomto súbore je vymazaná aj z pomocného súboru.

Záver

POP3 server patrí do oblasti sieťových aplikácií, konkrétne využíva prácu so soketmi, vláknami a rôzne ďalšie techniky, ktoré sú s implementáciou POP3 servera spojené.

V tomto dokumente sme sa venovali teoretickej časti popisujúcej základný princíp fungovania protokolu POP3 až po praktickú časť popisujúcu náš spôsob implementácie konkurentného servera pracujúceho v súlade s protokolom POP3 podľa RFC 1939.

Bibliografia

Bernstein, D. J. *Using maildir format.*

J. Myers, M. Rose. 1996. *RFC1939, Post Office Protocol - Version 3.* May 1996.

P. Resnick, Ed. 2008. *RFC5322, Internet Message Format.* s.l. : Qualcomm Incorporated, October 2008.