VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

Sieťové aplikácie a správa sietí

POP3 server

19. novembra 2017

Adrián Tóth

Obsah

1 Úvod **2**

2 Pojmy **2**
2.1 Maildir 2
2.2 IMF 2
2.3 POP3 2

3 Návrh a implementácia **3**
3.1 Moduly 3
3.2 Inicializácia, Spracovanie parametrov, Naviazanie komunikácie 4
3.3 Obsluha klienta zodpovedným vláknom 4
3.4 Vynútené ukončenie servera pomocou SIGINT 4
3.5 Reset servera, Pomocné súbory 4
3.6 MD5 hash 5

4 Návod na použitie **5**

5 Prílohy **7**

1 Úvod

Projekt sa vzťahuje k predmetu *ISA - Sieťové aplikácie a správa sietí*, kde je úlohou naimplementovať ***POP3 server***, ďalej označovaný ako *server*. Server je naimplementovaný podľa RFC 1939 [2] ktorý využíva *BSD sockets* na komunikáciu.

Server pracuje so súbormi v jednom priečinku, ktorý sa nazýva maildir. Maildir [1] je priečinok, ktorý obsahuje ďalšie podpriečinky *new*, *cur* a *tmp*. Všetky súbory v priečinku maildir by mali byť vo formáte IMF podľa RFC 5322 [3].

Server využíva okrem maildiru ešte jeden súbor a to kvôli autentifikácii užívateľa v ktorom sú uložené prihlasovacie údaje. Potrebné informácie, ktoré si server potrebuje uchovať, ukladá do logovacích súborov ktoré sa uložia do toho priečinka v ktorom je server spustený.

2 Pojmy

2.1 Maildir

Maildir označuje priečinok ktorý server využíva k svojmu behu. Maildir je zadaný hneď pri spustení po parametri „-d“ a to cestou ku priečinku. Priečinok môže byť zadaný absolútnou cestou alebo relatívnou. Maildir slúži na ukladanie správ (súborov) užívateľa do podpriečinkov podľa určitých kritérii. Maildir obsahuje podpriečinky *cur*, *new* a *tmp*. Súbory sa nachádzajú v podpriečinku new, ak sú označené ako nová správa. Po pripojení klienta sa presunú do podpriečinku cur. Priečinok cur obsahuje všetky správy, ktoré boli najskôr v priečinku new ale po pripojení klienta sa presunuli do tohto priečinka. Odstránenie správ z priečinka cur je záležitosť klienta. Priečinok tmp slúži ako dočasné úložisko pre server.

2.2 IMF

Internet Message Format v skratke IMF definuje formát správy ako je uložená v súbore. Definuje správu ako sekvenciu znakov tvoriace riadky, ktoré sú ukončené s dvoma znakmi a to carriage-return a line-feed. Carriage-return označovaný ako CR je znak „\r“ a line-feed označovaný ako LF je znak „\n“. Takže správa je vlastne množina riadkov, ktoré sú sekvencia po sebe idúcich znakov, kde každý riadok je ukončený s dvoma znakmi CRLF.

2.3 POP3

Pojem POP3 označuje Post Office Protocol - Version 3 [2] čo je internetový protokol aplikačnej vrstvy používanie na správu emailov. Tento protocol definuje pomocou množín noriem a pravidiel spôsob komunikácie s pop3 serverom a funkcionalitu tohto servera na ktorom je spustený. Tento protokolom definovaný server slúži k správe e-mailov a ich prenos medzi serverom a klientom.

Server je riadený jedným rozsiahlejším konečným automatomatom. Protokol špecifikuje jednotlivé stavy konečného automatu a príkazy ktoré určujú prechody medzi nimi, konečný automat môžete vidieť v prílohe na obrázku číslo 1. Z obrázku je zrejmé že pri pripojení klienta je počiatočný stav *authorization* kde sa čaká na autentifikáciu pomocou príkazov APOP alebo USER a PASS. Následne po autentifikácii sa uzamkne priečinok maildir a ostatný klienti musia počkať až kým sa pripojený klient neodpojí. Po autentifikácii sa server nachádza v stave *transaction*. V tomto stave môže spravovať správy uložené na serveri pomocou príkazov RSET, STAT, LIST, UIDL, RETR, DELE. V tomto stave sú ešte ďalšie dva príkazy a to NOOP, ktorý nevykoná nič, a QUIT, ktorý ukončí spojenie s klientom a presunie server do stavu *update*. Server v stave *transaction* nevykoná žiadnu zmenu nad súbormi, ale po presune do stavu *update* vykoná všetky klientom zadané zmeny nad maildirom, napríklad vymaže súbor z disku. Na konci stavu *update* sa odomkne maildir ktorý už obsahuje zmeny vykonané odhláseným klientom a čaká sa na ďalšieho klienta ktorý sa bude chcieť pripojiť.

Server vždy reaguje na príkazy prijaté od klienta dvojakým spôsobom, a to buď odošle pozitívnu odpoveď vo

forme „+OK ...“ alebo negatívnu odpoveď „-ERR ...“. Negatívna odpoveď obsahuje popis vzniknutej chyby a popis pozitívnej odpovede je definovaný v protokole POP3 [2].

3 Návrh a implementácia

K implementácii bol zvolený viacparadigmatový programovací jazyk C++, pretože umožňuje objektovo orientované programovanie. V ďalších podkapitolách bude popísaná samotná implementácia POP3 servera, ktorá popisuje projekt z hľadiska implementácie funkcionality tohto servera, t.j. ako je program delený na logické časti, ktoré zaručujú určitú časť funkcionality servera a spolu nasledujúc po sebe tak zaručujú správny beh programu.

3.1 Moduly

Projekt bol rozdelený na moduly, ktoré tvoria určité logické celky.

- *constatns*

  Modul obsahujúci konštanty. Tu sa nachádza aj konštanta, ktorá reprezentuje znakovú sadu z ktorej sa generuje *unique-id*. Podľa RFC 1939 *unique-id* pozostáva z ASCII znakov od 0x21 do 0x7E. Server však podporuje všetky tieto znaky, okrem znaku 0x2F, t.j. znak „ / “. Znak „ / “ sa využíva ako oddeľovací znak v logovacích súboroch. V tomto module sa nachádzajú aj konštanty určujúce veľkosti, názvy.

- *argpar*

  Modul obsahujúci funkciu na parsovanie vstupných parametrov, funkciu na vypísanie „help“ správy na stdout a funkciu na načítanie prihlasovacích údajov zo súbora. Funkcia argpar v závislosti od parametrov inicializuje lokálnu premennú args v popseri. S premennou args sa často pracuje, pretože obsahuje všetky informácie, ktoré sú potrebné k behu programu. Premenná je objekt vytvorený z triedy Args definovaný v module datatypes.

- *checks*

  Modul obsahujúci funkcie, ktoré slúžia na kontrolu im predaných parametrov.

- *datatypes*

  Modul obsahujúci mnou definované enumeračné premenné, triedu Args spomenutú v predošlom bode „*argpar*“ a ešte jednu funkciu. Enumeračné premenné sa používajú v automate, kde označujú stav alebo príkaz, ktorý sa nachádza v module fsm. Trieda Args slúži na posun informácii medzi funkciami pričom je inicializovaná hned na začiatku spustenia programu funkciou argpar. Funkcia, ktorá sa nachádza v tomtot module slúži na transformáciu vstupného reťazca na výstupný k nemu vzťahujúcu sa enumeračnú hodnotu typu Command.

- *fsm*

  Modul nazýva ako „finite-state machine“, skrátene fsm, obsahuje samotnú implementáciu konečného automatu podľa RFC 1939. Modul reprezentuje funkcionalitu jedného vytvoreného vlákna, ktoré obsluhuje napojeného klienta. Obsluha klienta je naimplementovaná ako konečný automat vo funkcii „thread_main“. V tejto funkcii sa nachádza všetko potrebné, aby sa správne obslúžil pripojený klient, odpojil a aby sa vykonali ním požadujúce zmeny. Taktiež sa tu nachádzajú funkcie, ktoré vypomáhajú konečnému automatu.

- *logger*

  Modul obsahujúci všetky ostatné funkcie, ktoré sú prevažne volané z fsm a z argpar. Nachádzajú sa tu funkcie ktoré vykonávajú logovanie informácii do súboru a prácu nad týmito informáciami. Taktiež sa tu nachádza funkcia reset.

- *md5*

  Modul ktorý obsahuje oddelenú časť zodpovednej za generovanie MD5 hashu pre príkaz APOP. Tento hash sa generuje zo špecifického typu reťazca, ktorý sa nazýva podľa RFC 1939 ako „greeting banner“. Zodpovedná funkcia na generovanie tohto reťazca sa tu tiež nachádza.

- *popser*

  Hlavná časť programu, kde sa volajú všetky ostatné moduly. Tu sa nachádza aj hlavná funkcia „main“. V tejto časti, po spracovaní parametrov, sa vytvorí komunikácia a pri úspešnom pripojení klienta sa vytvorí vlákno, ktorému je predaná funkcia „thread_main“ a soket na ktorom sa pripojil klient. Taktiež sa tu nachádza odchytávanie signálu SIGINT, čo má za následok správne ukončenie celého servera.

3.2 Inicializácia, Spracovanie parametrov, Naviazanie komunikácie

Pri prvom spustení servera sa vytvoria pomocné globálne premenné, ktoré budú riadiť beh celého procesu. Proces využíva tri globálne premenné z ktorých sú dve premenné typu *bool* a zvyšná premenná typu *std::mutex*.

Prvým krokom je spracovanie parametrov a inicializácia objektu typu Args ktorý nesie všetky potrebné informácie obsiahnuté z parametrov a následne sa predáva ďalším funkciám. Túto incializáciu zabezpečuje funkcia *argpar* v module *argpar*.

Po spracovaní parametrov nasleduje vytvorenie paralelného neblokujúceho soketu. Táto časť je naimplementovaná vo funkcii *server_kernel*. Táto funkcia vytvorí soket, nastaví soket, a ak sa klient pripojí tak vytvorí vlákno zodpovedný za obslúšenie klienta a predá mu príslušný soket.

3.3 Obsluha klienta zodpovedným vláknom

V tejto časti spočíva jadro celého popsera. Základná funkcionalita ktorá je popísana v kapitole 2.3 a znázornená v prílohe na obrázku číslo 1 je v tejto časti. Táto funkcionalita je naimplementovaná vo funkcii „thread_main“ ktorá sa nachádza v module „fsm“. Pri naviazaní spojenia sa vytvorí thread a je mu predaný soket a táto funkcia. V tejto funkcii je naimplementovaný konečný automat pomocou jedného hlavného *switch-case* príkazu určeného na stavy automatu ktorý v každom jednotlivom stave obsahuje jeden *switch-case* príkaz ktorý je určený na príkazy v stave. Obsluha klienta sa tak odohráva v počiatočnom stave *authorization*, v ktorom sa čaká na autentifikáciu pomocou príkazu APOP alebo USER a PASS. Prepínač „-c“ určuje ktorá metóda autentifikácie sa vyžaduje. Po úspešnom pripojení klienta sa presunie obsah podpriečinka „new“ do „cur“, a klient môže pracovať so svojimi správami. Po odhlásení klienta pomocou príkazu QUIT sa server dostane do stavu *update* kde vykoná zmeny, ukončí spojenie, odomkne maildir a ukončí thread ktorý obsluhoval tohto klienta.

Klient je obsluhovaný serverom až kým nevyprší timeout, t.j. čas 10 minút. Po vypršaní 10 minú je klient automaticky odpojený, všetky jeho zmeny anulované a priečinok maildir odomknutý.

3.4 Vynútené ukončenie servera pomocou SIGINT

Pri spustení servera sa spustí funkcia *std::signal* ktorá odchytáva singal interrupt. Pri jeho chytení sa hodnota globálnej premennej *flag_exit* nastaví na *true* čo spôsobí korektné ukončenie servera, t.j. ukončenie všetkých vlákien a uzatvorenie všetkých spojení.

3.5 Reset servera, Pomocné súbory

Samotný reset servera je dosť komplikovaná záležitosť pri ktorej si musíme uvedomiť, že server môže byť spustený bez zadania cesty priečinka maildir, v ktorom je potrebné realizovať reset. Z implementačného hľadiska sa to vyriešilo logovaním informácii do pomocných súborov uložených do toho priečinka v ktorom

bol spustený server. Server využíva dva pomocné súbory.

Prvý súbor nazvývaný „log“ obsahuje plné cesty súborov v priečinku *new* ktoré boli presunuté do *cur*. Pri resete sa načíta z tohto súbora celý obsah a súbory ktoré boli presunuté z *cur* sa presunú naspäť do pôvodného priečinka t.j. do *new* priečinka. Súbory ktoré používateľ vymazal z disku pomocou príkazu DELE sa automaticky vymažú aj zo všetkých logovacích súborov.

Druhý súbor nazývaný „data“ je vytvorený kvôli obmedzenému počtu prístupu k správe. Pri presune súborov z *new* do *cur* sa vytvorí pre každý súbor „unique-id“ popísaný v kapitole 3.1 v module *constatns*. Taktiež sa tu získa veľkosť správy a tá sa prepočíta vzhľadom ku ukončeniu riadkov. Tieto informácie sa následne uložia do logovacie súbora „data“ a to vo formáte *[filename][/][unique-id][/][filesize]*.

3.6 MD5 hash

POP3 server na to aby mohol využívať autentifikáciu klienta pomocou zdielaného tajného kľúča bolo nutné naimplementovať MD5 hashovaciu funkciu.

Pri pripojení klienta server automaticky odošle reťazec nazývaný ako *greeting banner*, ktorý je vygenerovaný pre každého klienta osobitne. Tento *greeting banner* je reťazec vo forme:

[<][PID][.][TIME][@][HOSTNAME][>]

Greeting banner sa uloží dočastne na serveri a odošle sa klientovi pri pripojení. Klient musí využiť tento *greeting banner* v prípade prihlásenia za pomoci príkazu APOP, a to pridaním hesla priamo za tento *greeting banner* a zahashovaním tohto reťazca. Hash tohto reťazca (*greeting banner* + heslo) sa nazýva *digest* ktorý má formu:

[<][PID][.][TIME][@][HOSTNAME][>][PASSWORD]

Klient môže odoslať príkaz APOP spolu s menom a *digest*. Server prijatú správu overí zda ním vygenerovaný hash je rovnaký ako prijatý. Server pozná heslo ktoré očakáva a aj *greeting banner* takže si môže vygenerovať hash na porovnanie prijatého hashu. Ak sa prijatý a aj vygenerovaný hash rovnajú a súčastne sa jedná o správny *username*, server môže prehlásiť že prihlásenie klienta prebehlo úspešne.

4 Návod na použitie

Server sa spúšťa s parametrami ktoré určujú chovanie behu programu. Parametre sú na začiatku spustenia programu spracované a pri ich kontrole sú vyhodnotené. Parametre za ktorými nasleduje hodnota, treba uviesť správnu hodnotu inak bude program ukončený a na stderr sa vypíše chybové hlásenie.

Zoznam parametrov a ich význam:

- *-h*

  Parameter môže byť zadaný samostatne alebo s ľubovoľným iným parametrom ktorý popser podporuje. Tento parameter slúži na vypísanie krátkej pomocnej nápovedy na stdout.

- *-r*

  Parameter umožňuje resetovať t.j. znovuobnoviť počiatočný stav maildiru. Parameter môže byť zadaný samostatne čo zapríčiní len reset maildiru alebo môže byť zadaný so spúšťacími parametrami čo má za následok reset maildiru a okamžité spustenie behu servera.

- *-c*

  Parameter zmení chovanie autentifikácie. Používateľ sa štandardne autentifikuje pomocou príkazu APOP, ale pri zadanom parametri -c, autentifikácia prebieha za pomoci príkazov USER a následne za ním príkazom PASS.

- *-p port-number*

  Parameter nastaví číslo portu na ktorom sa má spustit beh servera. Za číslo portu sa považuje číslo z intervalu 0 - 65535.

- *-a authentification-file*

  Parameter a jeho hodnota určia cestu k autorizačnému súboru, ktorý obsahuje validné prihlasovacie údaje používateľa, ktoré sú od neho očakávané pri autentifikácii resp. pri prihlásení na server. Súbor má určitý vymedzený formát uvedený nižšie.

  ```
  username = foo
  password = bar
  ```

- *-d maildir*

  Parameter a jeho hodnota určia cestu k maildiru, t.j. k priečinku ktorý je typu maildir [1].

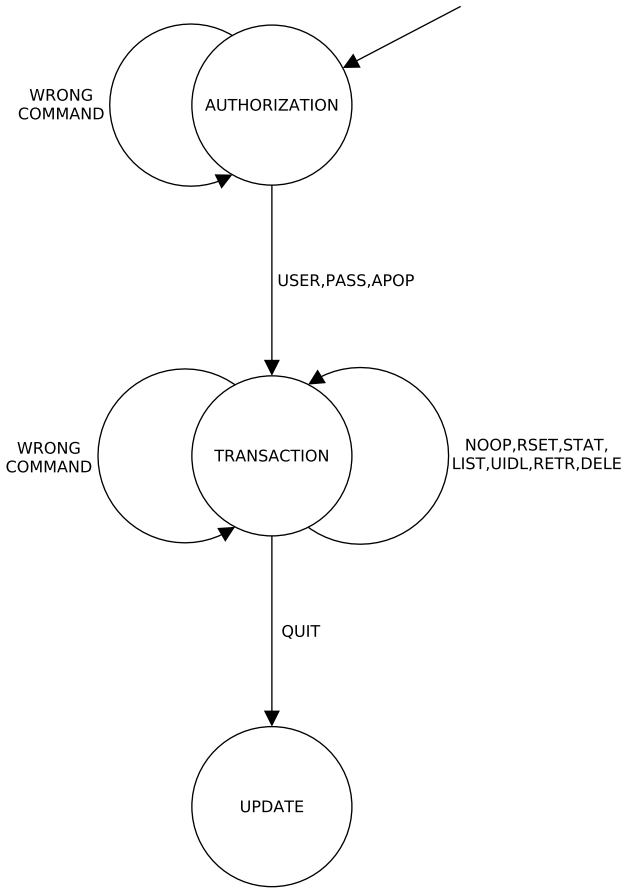
Kombináciou parametrov je možné rôzne ovplyvniť beh programu, ale napriek tomu že niektoré parametre sa na seba viažu tak si uvedieme ich možné kombinácie.

Možné kombinácie parametrov a ich význam pri spustení:

- Nápoveda:
  ./popser -h
- Iba Reset:
  ./popser -r
- Štandardné spustenie (autentifikácia používateľa prostredníctvom APOP):
  ./popser -a /foo/bar/authfile -d /foo/bar/maildir -p 55000
- Reset a štandardné spustenie (autentifikácia používateľa prostredníctvom APOP):
  ./popser -a /foo/bar/authfile -d /foo/bar/maildir -p 55000 -r
- Štandardné spustenie (autentifikácia používateľa prostredníctvom USER a PASS):
  ./popser -a /foo/bar/authfile -d /foo/bar/maildir -p 55000 -c
- Reset a štandardné spustenie (autentifikácia používateľa prostredníctvom USER a PASS):
  ./popser -a /foo/bar/authfile -d /foo/bar/maildir -p 55000 -c -r

Pri úspešnom spustení servera sa môžu napájať klienti na server. Server umožní prihlásenie len jedného a to prvého klienta ktorý sa úspšne prihlásil, všetci ostatní užívatelia ktorý sa prihlásia po ňom, sú odpojený a musia čakať až kým sa už prihlásený používateľ neodhlási. Klienti môžu využiť na pripojenie k serveru službu telnet kde uvedia host a port toho zariadenia, kde je spustení server. Napríklad: *telnet localhost 55000*.

5 Prílohy



Obr. 1: Konečný automat pop3 servera

Literatúra

[1] *Maildir*. [Online].
URL `https://cr.yp.to/proto/maildir.html`

[2] Myers, J.; Mellon, C.; Rose, M.: *Post Office Protocol - Version 3*. 1996, [Online].
URL `http://www.ietf.org/rfc/rfc1939.txt`

[3] P. Resnick, E.: *Internet Message Format*. 2008, [Online].
URL `http://www.ietf.org/rfc/rfc5322.txt`