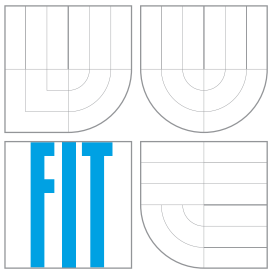
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

KLIENT IMAP S PODPOROU TLS

SEMESTRÁLNÍ PROJEKT
TERM PROJECT

AUTOR PRÁCE
AUTHOR

FRANTIŠEK ŠUMŠAL

BRNO 2016

Obsah

1 Úvod **2**

2 Implementace **3**
2.1 IMAP . . . 3
2.1.1 LOGIN . . . 4
2.1.2 SELECT . . . 4
2.1.3 FETCH . . . 4
2.1.4 SEARCH . . . 5
2.2 SSL/TLS . . . 6
2.2.1 Certifikáty . . . 7
2.3 Příklady použití . . . 7
2.3.1 Stažení všech zpráv z výchozí schránky (INBOX) - nešifrované . . . 7
2.3.2 Stažení všech zpráv ze schránky Trash - nešifrované . . . 7
2.3.3 Stažení hlaviček z výchozí schránky - nešifrované . . . 7
2.3.4 Stažení nových zpráv z výchozí schránky - nešifrované . . . 7
2.3.5 Stažení všech zpráv ze schránky Sent - šifrované . . . 8
2.3.6 Přechozí příklad se specifikováním umístění certifikátů . . . 8

Literatura **9**

Kapitola 1

Úvod

Cílem projektu byla implementace klienta komunikujíhího skrze IMAPv4 protokol [1]. Výsledný klient má zvládat i šifrovanou komunikaci s využitím SSL/TLS protokolu.

Kapitola 2

Implementace

Vlastní implementace proběhla v jazyce C++. Zadání omezuje použití knihoven na určitou podmnožinu, kterou tvoří standardní knihovny C/C++, knihovny pro práci se sítí a sockety a nakonec knihovna OpenSSL, pro implementaci šifrování.

2.1 IMAP

Pro nešifrovaný protokol IMAP jsou třeba BSD sockety a několik příkazů pro vlastní komunikaci se serverem. Během komunikace je třeba dodržovat určitou syntaxi pro požadavky a pro zpracování odpovědí. Každý požadavek začíná identifikátorem požadavku, za kterým následuje vlastní příkaz a jeho parametry. Server na tuto žádost odpoví zprávou, která dodržuje jeden ze dvou formátů:

1. *untagged response*
   Začíná hvězdičkou (*) a značí, že odpověď není kompletní:

   ```
   b SELECT INBOX
   * FLAGS (\Answered \Flagged \Deleted \Seen \Draft)
   * OK [PERMANENTFLAGS (\Answered ...] Flags permitted.
   * 2 EXISTS
   ```

2. *tagged response*
   Začíná identifikátorem z původního požadavku a obsahuje informaci o stavu provedeného požadavku:

   ```
   a LOGIN isa isa
   a OK [CAPABILITY IMAP4rev1 ...] Logged in
   ```

Provedení požadavku může skončit následujícími stavy:

OK	požadavek byl proveden úspěšně
BAD	syntaktická chyba v požadavku
NO	chyba při vykonávání požadavku

2.1.1 LOGIN

Příkaz login umožňuje autentizaci pomocí uživatelského jména a hesla. Syntaxe je následující:

```
LOGIN username password
```

Po úspěné autentizaci server vrací *OK*. Při chybném uživatelském jménu či heslu je vrácen stavový kód *NO*. Neplatné argumenty způsobí navrácení *BAD*.

```
a LOGIN isa ias
a NO [AUTHENTICATIONFAILED] Authentication failed.

b LOGIN isa isa
b OK [CAPABILITY IMAP4rev1 LITERAL+ SASL-IR ...] Logged in
```

2.1.2 SELECT

Pro výběr aktivní schránky slouží příkaz *SELECT*:

```
SELECT mailbox
```

Tento příkaz, mimo stavových kódů, povinně vrací i několik dalších informací. V tomto projektu byla využita pouze informace *EXISTS*, která určuje celkový počet emailů v dané schránce. Úspěšný přechod do dané schránky je dán stavem *OK*, neexistenci schránky oznamuje stav *NO*. Neplatné parametry jsou indikovány stavem *BAD*.

```
c SELECT nonexistent
c NO Mailbox doesn't exist: nonexistent (0.000 + 0.000 secs).

d SELECT INBOX
* FLAGS (\Answered \Flagged \Deleted \Seen \Draft)
* OK [PERMANENTFLAGS (\Answered \Flagged ...)] Flags permitted.
* 2 EXISTS
* 0 RECENT
* OK [UIDVALIDITY 1475686917] UIDs valid
* OK [UIDNEXT 3] Predicted next UID
* OK [HIGHESTMODSEQ 7] Highest
d OK [READ-WRITE] Select completed (0.000 + 0.000 secs).
```

2.1.3 FETCH

Stažení emailu lze provést příkazem *FETCH*:

```
FETCH mail-uid arguments
```

Argumentů pro přikaz *FETCH* existuje celá řada, projekt však využívá jen dvou:

BODY[] stažení kompletního těla emailu

BODY[HEADER] stažení pouze hlavičky

Tělo emailu odpovídá formátu *Internet Message Format* [3] [4]. *OK* indikuje úspěšné získání emailu, *NO* značí chybu během přenosu/získávání dat a *BAD* je navrácen v případě neplatných argumentů.

```
e FETCH 1 BODY[]
* 1 FETCH (BODY[] {834}
Return-Path: <fsumsal@localhost.localdomain>
X-Original-To: isa@localhost.localdomain
Delivered-To: isa@localhost.localdomain
...
From: Frantisek Sumsal <fsumsal@localhost.localdomain>
Message-Id: <201610051706.u95H6nT3006867@localhost.localdomain>
Date: Wed, 05 Oct 2016 19:06:49 +0200
To: isa@localhost.localdomain
Subject: Test
User-Agent: Heirloom mailx 12.5 7/5/10
MIME-Version: 1.0
Content-Type: text/plain; charset=us-ascii
Content-Transfer-Encoding: 7bit

AHOJ
)
e OK Fetch completed (0.001 + 0.000 secs).

f FETCH 1 BODY[HEADER]
* 1 FETCH (BODY[HEADER] {828}
Return-Path: <fsumsal@localhost.localdomain>
X-Original-To: isa@localhost.localdomain
Delivered-To: isa@localhost.localdomain
...
From: Frantisek Sumsal <fsumsal@localhost.localdomain>
Message-Id: <201610051706.u95H6nT3006867@localhost.localdomain>
Date: Wed, 05 Oct 2016 19:06:49 +0200
To: isa@localhost.localdomain
Subject: Test
User-Agent: Heirloom mailx 12.5 7/5/10
MIME-Version: 1.0
Content-Type: text/plain; charset=us-ascii
Content-Transfer-Encoding: 7bit

)
f OK Fetch completed (0.001 + 0.000 secs).
```

2.1.4 SEARCH

Posledním využitým IMAP příkazem je *SEARCH* pro vyhledávání emailů vyhovující určitým podmínkám:

```
SEARCH criteria
```

Kritérií pro vyhledávání opět existuje celá řada, pro projekt je však důležité jen kritérium *UNSEEN*, pomocí kterého se dají vyhledat všechny nepřečtené zprávy. Odpověd serveru poté obsahuje unikátní identifikátory takto označených zpráv, které lze poté využít s pří-

kazem *FETCH* pro jejich stažení. Společně s identifikátory je vrácen stav *OK*, pokud vše proběhlo v pořádku, *NO* při chybě ve vyhledávacích kritériích nebo *BAD* při neplatných argumentech.

```
g STORE 1:2 -FLAGS (\Seen)
* 2 FETCH (FLAGS ())
g OK Store completed (0.001 + 0.000 secs).
h SEARCH UNSEEN
* SEARCH 1 2
h OK Search completed (0.001 + 0.000 secs).
k FETCH 1:2 BODY[]
...
k OK Fetch completed (0.003 + 0.000 + 0.002 secs).
l SEARCH UNSEEN
* SEARCH
l OK Search completed (0.001 + 0.000 secs).
```

2.2 SSL/TLS

Pro implementaaci šifrování pomocí protokolů SSL/TLS byla využita knihovna OpenSSL [2]. Pro využití šifrování pro protokol IMAP je možné bez problémů využít stávající implementaci, jen je třeba povýšit spojení se serverem na šifrované. Toho se dá dosáhnout využitím několika funkcí OpenSSL, které vytvoří nový kontext, nastaví povolené šifry a cesty k certifikátům, pro validaci serveru a poté s využitím stávajícího nešifrovaného spojení provede jeho povýšení na šifrované.

Pro následnou práci se socketem je třeba místo standardních funkcí *read()* and *write()* použít jejich protějšky podporující šifrování - *SSL_read()* a *SSL_write()*.

Aby byla práce se socketem "unifikovaná"a kód byl tedy přehlednější, byly obě verze zmíněných funkcí obaleny pomocnou funkcí, které zajistí jednotné volání. Místo předávání čísla socketu bylo třeba vytvořit pomocnou strukturu, která uchovává informace o šifrovaném i nešifrovaném spojení.

```
typedef struct {
    bool tls = false;        /**< TLS state (enabled/disabled) */
    unsigned int cnt = 0;    /**< Request number */
    int mail_count = -1;     /**< Mail count for currently selected
                                  mailbox */
    int sd = -1;             /**< Socket descriptor */
    SSL *tlsd = NULL;        /**< OpenSSL socket descriptor */
} connection_data_t;

ssize_t socket_read(connection_data_t *conn, void *buf, size_t nbyte)
{
    if(conn->tls) {
        return SSL_read(conn->tlsd, buf, nbyte);
    } else {
        return read(conn->sd, buf, nbyte);
    }
}

ssize_t socket_write(connection_data_t *conn, void *buf, size_t nbyte)
```

```
{
    if(conn->tls) {
        return SSL_write(conn->tlsd, buf, nbyte);
    } else {
        return write(conn->sd, buf, nbyte);
    }
}
```

2.2.1 Certifikáty

Aktuální implementace ověřuje validitu serverového certifikátů během povyšování připojení na šifrované. Z toho důvodu je nutné mít na cílové stanici uložené vhodné certifikáty, se kterými se dá ověřit důvěryhodnost daného serveru.

Pokud tyto certifikáty daná stanice nemá nebo chce-li uživatel použít certifikáty v nevýchozím umístění, je možné použít patřičné parametry klienta. V každém případě musí být certifikáty uložené ve formátu PEM. V případě souboru je situace jednoduchá, stačí spustit klienta s parametrem *-c* a cestou k souboru s certifikátem či certifikáty. U adresářů se situace poněkud komplikuje. Knihovna OpenSSL očekává soubory v adresářích uložené pod specifickým názvem. Jednou z možností je tento názvev ručně zjistit pomocí dostupných utilit a soubor vhodně přejemenovat nebo lze použít utilitu *c_rehash*, která pro každý certifikát nalezený v adresáří vygeneruje správný název a vytvoří symbolický odkaz s tímto názvem na původní soubor. Poté je adresář připraven k použití s klientem s parametrem *-C*.

2.3 Příklady použití

Pro výpis všech podporovaných funkcí stačí klienta spustit bez parametrů. Jelikož veškerá manipulace se schránkami vyžaduje autentizaci, je potřeba vytvořit konfigurační soubor, který bude obsahovat údaje pro přihlášení. Tento soubor má následující formát:

```
username = uzivatelske_jmeno
password = heslo
```

2.3.1 Stažení všech zpráv z výchozí schránky (INBOX) - nešifrované

```
$ ./imapcl imap.server.tld -a auth.conf -o download/
Downloaded 692 messages from mailbox INBOX
```

2.3.2 Stažení všech zpráv ze schránky Trash - nešifrované

```
$ ./imapcl imap.server.tld -a auth.conf -o download/ -b Trash
Downloaded 96 messages from mailbox Trash
```

2.3.3 Stažení hlaviček z výchozí schránky - nešifrované

```
$ ./imapcl imap.server.tld -a auth.conf -o download/ -h
Downloaded 692 message headers from mailbox INBOX
```

2.3.4 Stažení nových zpráv z výchozí schránky - nešifrované

```
$ ./imapcl imap.server.tld -a auth.conf -o download/ -n
Downloaded 17 new messages from mailbox INBOX
```

2.3.5 Stažení všech zpráv ze schránky Sent - šifrované

```
$ ./imapcl imap.server.tld -T -a auth.conf -o download/ -b Sent
Downloaded 51 messages from mailbox Sent
```

2.3.6 Přechozí příklad se specifikováním umístění certifikátů

```
$ ./imapcl imap.server.tld -T -C certs/ -a auth.conf -o download/ -b Sent
Downloaded 51 messages from mailbox Sent
```

Literatura

[1] Crispin, M.: *INTERNET MESSAGE ACCESS PROTOCOL - VERSION 4rev1*. RFC 3501, Březen 2003.
URL https://www.rfc-editor.org/rfc/rfc3501.txt

[2] OpenSSL Software Foundation.: *OpenSSL ssl library documentation*. https://www.openssl.org/docs/man1.0.1/ssl/, 2016, [Online; navštíveno 18.10.2016].

[3] Resnick, P., Ed.: *Internet Message Format*. RFC 2822, Duben 2001.
URL https://www.rfc-editor.org/rfc/rfc2822.txt

[4] Resnick, P., Ed.: *Internet Message Format*. RFC 5322, Říjen 2008.
URL https://www.rfc-editor.org/rfc/rfc5322.txt