Project Name: IMAP Client

Technical Documentation

Author: Slabik Yaroslav
Login: xslabi01

Contents

---

1. Introduction

This documentation covers the **imapcl** IMAP client developed as part of the ISA course project at the Faculty of Information Technology, Brno University of Technology. The goal was to implement a client for the IMAP4rev1 protocol (RFC 3501), enabling email downloading from a specified server and saving messages to a designated directory.

2. Assignment Description

The **imapcl** program provides:

- Connection to a specified IMAP server using IMAP4rev1.
- User authentication with provided credentials.
- Downloading messages from a specified mailbox (default "INBOX") and saving them to a specified directory.
- Displaying the number of downloaded messages on standard output.
- Ability to set additional parameters to modify functionality (e.g., use of SSL/TLS, port selection, downloading only new messages, downloading only headers, etc.).

3. Application Design

3.1 Program Architecture

The program is designed modularly with emphasis on clarity and maintainability. Main components:

- **Command Line Argument Parser**: Processes input parameters and configures program behavior.
- **Network Module**: Establishes connection to the server and communicates using IMAP.
- **Authentication Module**: Logs the user into the server.
- **Message Processing Module**: Handles downloading, processing, and saving messages.
- **SSL/TLS Module**: Ensures secure server connection.

3.2 Code Structure

Code is divided into functions, each serving a specific purpose:

- **initialize_ssl**, **create_context**, **cleanup_ssl**: Functions for OpenSSL operations.
- **connect_to_server**: Connects to the server.
- **send_command**, **receive_response**: Communicates with the server using IMAP commands.
- **login**, **select_mailbox**, **fetch_messages**: Implements main IMAP functions.
- **read_credentials**, **directory_exists**: Helper functions for file and directory operations.
- **base64_encode**, **base64_decode**, **decode_encoded_word**: Functions for message encoding and decoding.

4. Implementation Description

4.1 Command Line Argument Parsing

The program uses the **getopt** library for parameter parsing:

- **server**: Server address (required).
- **-p port**: Port number (default 143 or 993 with TLS).
- **-T**: Use SSL/TLS.
- **-c certfile**: Certificate file for SSL/TLS verification.
- **-C certaddr**: Certificate directory (default "/etc/ssl/certs").
- **-n**: Download only new messages.
- **-h**: Download only message headers.
- **-a auth_file**: Authentication file (required).
- **-b MAILBOX**: Mailbox name (default "INBOX").
- **-o out_dir**: Output directory (required).

4.2 Establishing Server Connection

Function **connect_to_server**:

- Retrieves server info using **getaddrinfo**.
- Creates a socket and establishes a TCP connection.
- If TLS is used, initializes SSL context and establishes a secure connection.
- Verifies server certificate using OpenSSL.

4.3 User Authentication

Function **login**:

- Receives server's greeting.
- Determines supported authentication methods.
- Logs in using LOGIN or PLAIN method.

4.4 Mailbox Selection

Function **select_mailbox**:

- Sends **SELECT** command to choose the mailbox.
- Processes response to get the message count.

4.5 Downloading and Saving Messages

Function **fetch_messages**:

- Determines which messages to download.
- Sends **FETCH** commands to download messages or headers.
- Processes responses including literals.
- Saves messages to the specified directory with necessary processing.

4.6 Working with SSL/TLS Certificates

- Loads certificates from file or directory.
- Sets server certificate verification mode.
- Verifies the certificate after establishing SSL connection.

5. User Manual

5.1 Authentication File Format

The file must contain:

```
username = your_username
password = your_password
```

5.2 Examples of Program Execution

1. Download all messages without TLS:

```
./imapcl server -a auth.txt -o zpravy
```

2. Download new messages with TLS and certificate:

```
./imapcl server -T -c cert.pem -n -a auth.txt -o zpravy
```

3. Download message headers:

```
./imapcl server -h -a auth.txt -o hlavicky
```

6. Application Testing

6.1 Description of Tests Performed

- **Connection tests** with and without TLS.
- **Authentication tests** with valid and invalid credentials.
- **Downloading various types of messages**.
- **Certificate handling tests**.
- **Saving messages** to different directories.

6.2 Testing Results

- **Connection** succeeded with correct settings.
- **Authentication** was successful with valid credentials.
- **Message downloading** worked in all modes.
- **Certificates** were correctly handled.
- **Saving messages** succeeded to accessible directories.

7. References

1. **RFC 3501** - IMAP4rev1
2. **RFC 5322** - Internet Message Format
3. **OpenSSL Documentation**
4. **Standard C/C++ Libraries**