ĐẠI HỌC QUỐC GIA TPHCM

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

KHOA CÔNG NGHỆ THÔNG TIN

Lập trình Socket

Đề tài: Email Client

Môn học: Mạng máy tính

Sinh viên thực hiện:
Phạm Hà Hiếu (22127114)
Lê Gia Huy (22127152)
Nguyễn Duy Bảo (22127031)

Giảng viên hướng dẫn:
ThS. Nguyễn Thanh Quân

Ngày 1 tháng 12 năm 2023

Mục lục

I. THÔNG TIN THÀNH VIÊN

MSSV	Họ và tên	Tài khoản Github
22127114	Phạm Hà Hiếu	hieupham19-hcmus
22127152	Lê Gia Huy	legiahuy
22127031	Nguyễn Duy Bảo	ndbao

Bảng 1: Thông tin thành viên

II. Cấu Hình Mail Server Mô Phỏng (Phiên Bản Không Mã Hóa)

Đồ án này sử dụng một Mail server mô phỏng không áp dụng kết nối mã hóa SSL/TLS. Mục tiêu là để quan sát và phân tích một cách minh bạch quá trình giao tiếp giữa Mail client và Mail server.

- **Mail Server Mô Phỏng:** Server được xây dựng sử dụng Java 8, có thể tải về từ link sau. Bạn có thể truy cập mã nguồn trên Github.
- **Hướng Dẫn Cài Đặt Java 8:** Để chạy server, cần cài đặt Java 8 (JDK 8u361). Bạn có thể tìm hướng dẫn cài đặt chi tiết tại đây.
- **Cách Thức Khởi Chạy:** Khởi chạy server bằng lệnh

```
java -jar test-mail-server-<version>.jar <args>
```

III. KỊCH BẢN GIAO TIẾP CỦA CHƯƠNG TRÌNH

1. Giao thức trao đổi

Giao thức trao đổi giữa Client và Server của tầng vận chuyển được sử dụng trong chương trình:

- **SMTP (Simple Mail Transfer Protocol):** Là giao thức chuẩn trong mô hình TCP/IP, chủ yếu được dùng để gửi thư điện tử (e-mail) trên Internet. Nó cung cấp khả năng thiết lập kênh kết nối giữa ứng dụng mail client (như Microsoft Outlook, Mozilla Thunderbird) và mail server, cho phép gửi thư từ client đến server.
- **POP3 (Post Office Protocol version 3):** Giao thức này hỗ trợ việc kết nối đến mail server để tải email xuống máy khách qua các ứng dụng mail client. Nó giúp người dùng lưu trữ và quản lý email một cách hiệu quả trên thiết bị cá nhân.

2. Cấu trúc thông điệp

Thư điện tử (email) được định dạng theo chuẩn MIME (Multipurpose Internet Mail Extensions), quy định cách thức hiển thị và truyền tải thông điệp email cùng các tệp đính kèm.

- **Content-Type: multipart/mixed; boundary="ID":** Xác định cấu trúc nội dung email, cho phép kết hợp nhiều loại nội dung (văn bản, hình ảnh, tệp đính kèm, v.v.).
- **MIME-Version: 1.0:** phiên bản của chuẩn MIME được sử dụng.
- **Message-ID: 123abc@gmail.com:** Là định danh duy nhất cho mỗi thông điệp email, hỗ trợ theo dõi và quản lý email hiệu quả.
- **User-Agent: PhamHaHieu:** Cung cấp thông tin về client hoặc ứng dụng dùng để tạo email.
- **Date: Sun, 12 Nov 2023 00:17:57 +0700:** Ghi nhận thời gian gửi email.
- **To, From, Cc, Bcc, Subject:** Các trường thông tin cơ bản của email, bao gồm người nhận, người gửi, người nhận bản sao, người nhận bản sao ẩn, và chủ đề của email.
- **To, From, Cc, Bcc, Subject:** Các trường thông tin cơ bản của email, bao gồm người nhận, người gửi, người nhận bản sao, người nhận bản sao ẩn, và chủ đề của email.
- **Boundary:** Mọi phần của email nằm giữa các boundary này bao gồm: boundary đầu tiên, đánh dấu sự bắt đầu của một phần trong multipart và boundary kết thúc của phần multipart. Boundary được khai báo trong tiêu đề Content-Type. Các tin nhắn chỉ chứa một phần không yêu cầu sử dụng boundary vì không có gì cần phải phân định rõ ràng.
- **Content-Transfer-Encoding:** Phương thức mã hóa được sử dụng để chuyển đổi dữ liệu từ văn bản gốc sang dạng có thể truyền qua mạng. Ví dụ: 7bit, 8bit, base64 (Được sử dụng nhiều trong việc mã hóa đính kèm trong email.), etc.

3. Lưu trữ và quản lý email

- Email được lưu về máy dưới định dạng .eml, cho phép lưu giữ toàn bộ nội dung email.
- Sử dụng cơ sở dữ liệu SQLite để quản lý các Message-ID cùng với trạng thái đã đọc hoặc chưa đọc, giúp xác định email nào đã được tải về.

4. Tổ chức Cơ Sở Dữ Liệu

Cơ sở dữ liệu được sử dụng trong đồ án này là SQLite3. Các thành phần chính của cơ sở dữ liệu bao gồm:

- **message-id:** Đây là trường dùng để lưu trữ các ID của email đã được tải về máy.
- **status:** Trường này biểu diễn trạng thái của email, với giá trị 1 đại diện cho email đã đọc, và 0 là email chưa đọc.

IV. Môi trường lập trình và các Thư viện hỗ trợ

1. Môi trường lập trình

Dự án này được phát triển trên nền tảng kiến thức thu được từ tài liệu *PythonNetBinder* của David M. Beazley [1], đóng vai trò là một hướng dẫn quan trọng trong việc xây dựng và triển khai ứng dụng.
Ngôn ngữ lập trình mặc định của đồ án là Python
Phiên bản: Python 3.11.6
Quản lý các phiên bản code tại: Github
Phần mềm đóng gói ứng dụng là: Pyinstaller

2. Các thư viện sử dụng

2.1. Thư viện chính (Phục vụ cho việc gửi và nhận email)

- **Thư viện socket:** Thư viện socket trong Python cung cấp các công cụ cần thiết để thiết lập kết nối mạng giữa các máy tính. Nó cho phép tạo ra các kết nối TCP/IP và UDP, là nền tảng quan trọng cho việc trao đổi dữ liệu qua mạng, bao gồm việc gửi và nhận email. Việc sử dụng socket giúp xây dựng cơ chế truyền thông cấp thấp, tạo điều kiện cho việc trao đổi thông tin giữa máy chủ và máy khách trong quá trình xử lý email.

- **Thư viện email:** Thư viện email trong Python cung cấp các công cụ để tạo, thao tác và phân tích cấu trúc của thư điện tử. Nó hỗ trợ việc tạo ra các thư điện tử với định dạng chuẩn, bao gồm cả phần đầu và nội dung của thư, cũng như việc xử lý các tệp đính kèm. Thư viện này giúp đơn giản hóa quá trình tạo và phân tích các email, là công cụ không thể thiếu trong việc phát triển các ứng dụng liên quan đến email.

- **Thư viện sqlite3:** Thư viện sqlite3 cung cấp giao diện để tương tác với cơ sở dữ liệu SQLite. SQLite là một hệ quản trị cơ sở dữ liệu quan hệ nhẹ, được lưu trữ trong một tệp tin đơn. Thư viện sqlite3 cho phép thực hiện các thao tác cơ bản như tạo, truy vấn, cập nhật và xóa dữ liệu trong cơ sở dữ liệu. Việc sử dụng sqlite3 phổ biến trong các ứng dụng nhỏ đến trung bình, đặc biệt hữu ích trong việc lưu trữ và truy xuất dữ liệu cấu hình, người dùng, hoặc log thông tin liên quan đến quá trình gửi và nhận email.

- **Thư viện os:** Thư viện os trong Python cung cấp một cách tiếp cận di động qua các hệ điều hành để sử dụng chức năng phụ thuộc vào hệ thống, như tương tác với hệ điều hành hoặc hệ thống tệp tin. Thư viện này cho phép thực hiện các thao tác như đọc, viết và xóa tệp, quản lý đường dẫn, và truy xuất thông tin môi trường. Trong bối cảnh gửi và nhận email, thư viện os có thể được sử dụng để quản lý các tệp đính kèm, tạo và quản lý thư mục tạm thời, hoặc truy xuất các biến môi trường liên quan đến cấu hình máy chủ email.

2.2. Thư viện phụ (Phục vụ cải thiện trải nghiệm người dùng)

- **Thư viện json:** Thư viện json trong Python cung cấp chức năng phân tích cú pháp (parse) và tạo ra dữ liệu JSON. JSON (JavaScript Object Notation) là một định dạng dữ liệu nhẹ, dễ đọc và ghi cho người dùng, dễ phân tích cú pháp và tạo ra cho máy tính. Thư viện json

được sử dụng rộng rãi trong việc trao đổi dữ liệu qua mạng, bao gồm việc gửi và nhận dữ liệu API, cấu hình, hoặc các thông tin liên quan đến email ở dạng chuẩn và dễ sử dụng.

- **Thư viện glob:** Thư viện glob cung cấp một phương thức để tìm kiếm tất cả các tệp tin trong một thư mục phù hợp với một mẫu cụ thể. Nó thường được sử dụng để lấy danh sách các tệp tin dựa trên tiêu chí nhất định (ví dụ: mở rộng tệp). Trong bối cảnh gửi và nhận email, glob có thể hữu ích trong việc xử lý các tệp đính kèm hoặc tìm kiếm các tệp log, cấu hình, hoặc dữ liệu liên quan đến email theo mẫu cụ thể trong một thư mục.
- **Thư viện hashlib:** Thư viện hashlib trong Python cung cấp các hàm băm (hashing), cho phép chuyển đổi một chuỗi dữ liệu thành một giá trị ngắn, cố định, thường được dùng để bảo mật thông tin. Trong đoạn mã ví dụ, 'hashlib' được sử dụng để tạo một chuỗi băm duy nhất từ một chuỗi ngẫu nhiên và thời gian hiện tại. Điều này giúp tạo ra một ID tin nhắn duy nhất cho mỗi email, tăng cường tính an toàn và khả năng truy xuất trong hệ thống quản lý email.
- **Thư viện datetime:** Thư viện datetime cung cấp các chức năng để làm việc với ngày và giờ. Trong đoạn mã, 'datetime' được sử dụng để lấy thời gian hiện tại, điều này đóng vai trò quan trọng trong việc tạo ra chuỗi băm duy nhất cho ID tin nhắn. Việc kết hợp thời gian hiện tại vào quá trình tạo ID giúp đảm bảo rằng mỗi ID được tạo ra là duy nhất và không lặp lại.
- **Thư viện random:** Thư viện random cung cấp các hàm để tạo ra dữ liệu ngẫu nhiên. Trong đoạn mã, 'random' được sử dụng để tạo một chuỗi ngẫu nhiên, là một phần của quá trình tạo ID tin nhắn. Sự ngẫu nhiên này tăng thêm độ duy nhất và an toàn cho ID tin nhắn, giúp ngăn chặn việc xung đột ID giữa các tin nhắn khác nhau.

V. MÔ TẢ HÀM VÀ CHỨC NĂNG CỦA CHƯƠNG TRÌNH

1. readConfig.py

File `readConfig.py` chứa hàm `read_config_file` dùng để đọc và phân tích cú pháp các file cấu hình.

1.1. Hàm `read_config_file(filepath)`

- **Mục Đích:** Đọc và trả về cấu hình từ file JSON, XML, hoặc TXT.
- **Tham Số:**
  - `filepath` - Đường dẫn đến file cấu hình.
- **Logic Chính:**
  1. Phân biệt định dạng file dựa trên phần mở rộng.
  2. Đọc và chuyển đổi file JSON thành từ điển Python.
  3. Đọc và xử lý file XML, hỗ trợ tag `General` và `Filters`.
  4. Đọc và xử lý file TXT với định dạng `key: value`.
  5. Xử lý các lỗi như `JSONDecodeError`, `ParseError`, và `FileNotFoundError`.

- **Kiểm Tra Các Key Cần Thiết:** Xác nhận sự tồn tại của các key quan trọng trong phần `general`.
- **Giá Trị Trả Về:** Từ điển Python chứa cấu hình hoặc `None` nếu có lỗi.

2. Send_Email.py

File `Send_Email.py` chứa các hàm hỗ trợ gửi email thông qua giao thức SMTP không mã hóa, với khả năng gửi nội dung đơn giản và tùy chọn file đính kèm.

2.1. Hàm `generate_message_id(userEmail)`

- **Mục Đích:** Tạo ID tin nhắn duy nhất dựa trên email người gửi và thời gian hiện tại.
- **Tham Số:**
  - `userEmail`: Email của người gửi, dùng để xác định tên miền trong ID tin nhắn.
- **Cách Hoạt Động:**
  1. Lấy thời gian hiện tại và tạo một chuỗi ngẫu nhiên.
  2. Kết hợp chuỗi ngẫu nhiên và thời gian để tạo một chuỗi hash MD5.
  3. Sử dụng hash này cùng với tên miền của email người gửi để tạo ID tin nhắn.
- **Giá Trị Trả Về:** ID tin nhắn dưới dạng chuỗi.

2.2. Hàm `send_email(...)`

- **Mục Đích:** Gửi email đến một hoặc nhiều người nhận, hỗ trợ file đính kèm và CC, BCC.
- **Tham Số:**
  - `host`: Tên máy chủ SMTP.
  - `smtp_port`: Cổng của máy chủ SMTP.
  - `userName`: Tên người gửi.
  - `userEmail`: Email của người gửi.
  - `userSubject`: Tiêu đề email.
  - `userContent`: Nội dung email.
  - `toEmails`: Danh sách email người nhận.
  - `ccEmails`: Danh sách email CC.
  - `bccEmails`: Danh sách email BCC.
  - `attachmentFilePaths`: Đường dẫn file đính kèm.
- **Logic Chính:**
  1. Tạo và kết nối socket với máy chủ SMTP.

2. Xác thực và cấu hình tin nhắn MIME.
3. Gửi email qua SMTP.
4. Đóng kết nối socket.

2.3. Lưu Ý và Hạn Chế

- Email gửi qua SMTP không mã hóa có thể không an toàn cho thông tin nhạy cảm.
- Giới hạn kích thước file đính kèm là 3 MB.
- Yêu cầu thông tin máy chủ SMTP chính xác.

2.4. Sử Dụng

Cần cung cấp đầy đủ thông tin người gửi, người nhận và máy chủ SMTP khi sử dụng script này.

3. filter.py

File `filter.py` bao gồm một hàm để lọc email dựa trên các bộ lọc cấu hình.

3.1. Hàm `filter(email, config)`

- **Mục Đích:** Lọc email và phân loại vào thư mục thích hợp dựa trên bộ lọc cấu hình.
- **Tham Số:**
  - `email`: Đối tượng email hoặc chuỗi chứa nội dung email.
  - `config`: Từ điển cấu hình chứa các bộ lọc.
- **Logic Chính:**
  1. Chuyển đổi email sang đối tượng `EmailMessage` nếu cần.
  2. Trích xuất chủ đề, người gửi và nội dung.
  3. Xử lý nội dung nếu email là nhiều phần.
  4. Kiểm tra và áp dụng bộ lọc dựa trên người gửi, chủ đề và nội dung.
- **Giá Trị Trả Về:** Thư mục tương ứng dựa trên bộ lọc phù hợp, hoặc 'Inbox' nếu không có bộ lọc nào khớp.

3.2. Sử Dụng

Cần cung cấp email cần lọc và cấu hình bộ lọc dưới dạng từ điển, với tiêu chí lọc rõ ràng và thư mục tương ứng cho các email khớp.

4. Receive_Email.py

File `Receive_Email.py` chứa các hàm để nhận và xử lý email từ máy chủ POP3.

4.1. Hàm `receive_email(...)`

- **Mục Đích:** Nhận và lọc email từ máy chủ POP3, lưu trữ chúng dựa trên bộ lọc cấu hình.
- **Tham Số:**
  - `host`: Tên máy chủ POP3.
  - `pop3_port`: Cổng của máy chủ POP3.
  - `user_email`: Email của người dùng.
  - `user_password`: Mật khẩu của người dùng.
  - `config`: Cấu hình bộ lọc email.
- **Logic Chính:**
  1. Kết nối và xác thực với máy chủ POP3.
  2. Nhận danh sách và UIDL của email.
  3. Tải và xử lý từng email.
  4. Lọc và lưu trữ email vào thư mục phù hợp.
  5. Ghi nhận ID của tin nhắn đã xử lý.

4.2. Các Hàm Hỗ Trợ

- `save_processed_id(msg_id)`: Lưu ID của email đã xử lý.
- `load_processed_ids()`: Tải các ID email đã xử lý.
- `extract_message_id(email_str)`: Trích xuất ID tin nhắn.
- `remove_metadata(email_str)`: Loại bỏ metadata khỏi phản hồi email.
- `process_email(raw_email_bytes)`: Xử lý email và trả về nội dung đầy đủ.

4.3. Sử Dụng

Để sử dụng script này, cần cung cấp thông tin đăng nhập và máy chủ POP3, script sẽ tự động nhận, xử lý, lọc và lưu trữ email.

5. EmailProcessor.py

File `EmailProcessor.py` chứa các hàm để xử lý và hiển thị thông tin từ các file email.

5.1. Hàm `print_email_details(email)`

- **Mục Đích:** Hiển thị chi tiết của email bao gồm người gửi, người nhận, CC, BCC, ngày, chủ đề và nội dung.
- **Tham Số:**
  - `email`: Đối tượng email cần hiển thị thông tin.

5.2. Các Hàm Hỗ Trợ

- `read_mail_from_file(path)`: Đọc và trả về nội dung email từ file.
- `remove_metadata(email_str)`: Loại bỏ metadata từ chuỗi email.
- `list_emails_in_folder(folder)`: Liệt kê các email trong thư mục.
- `pick_mail_in_folder(folder, index)`: Chọn và xử lý một email cụ thể trong thư mục dựa trên chỉ mục.

5.3. Sử Dụng

Để sử dụng `EmailProcessor.py`, cần định vị các file email và cung cấp cho script. Script sẽ đọc nội dung email, xử lý và hiển thị các thông tin chi tiết, bao gồm cả các file đính kèm.

6. client.py

File `client.py` là ứng dụng client email dựa trên dòng lệnh, kết hợp chức năng từ các module khác nhau để cung cấp trải nghiệm email đầy đủ.

6.1. Hàm `main()`

- **Mục Đích:** Chạy ứng dụng client email, xử lý nhận và gửi email, cung cấp giao diện người dùng dòng lệnh.
- **Logic Chính:**
  1. Đọc cấu hình từ `config.json`.
  2. Khởi tạo và chạy hàm tự động nhận email.
  3. Cung cấp menu tương tác người dùng.
  4. Xử lý các lựa chọn người dùng và gọi hàm tương ứng.

6.2. Các Hàm Hỗ Trợ

- `clear_screen()`: Xóa màn hình terminal.
- `wait_for_enter()`: Chờ người dùng nhấn Enter.
- `auto_run_function(interval, func, *args)`: Tự động chạy hàm sau khoảng thời gian.
- `get_existing_mail_folders()`: Lấy danh sách thư mục email.

6.3. Sử Dụng

Chạy script để tương tác qua menu dòng lệnh. Cần cấu hình thông tin email trong `config.json` trước khi sử dụng.

VI. ĐÁNH GIÁ MỨC ĐỘ HOÀN THÀNH

STT	Chức năng	Tiến độ	Người nhận trách nhiệm
1	Gửi Email với TO và CC	100%	Hà Hiếu, Gia Huy, Duy Bảo
2	Gửi Email với BCC	100%	Hà Hiếu, Gia Huy
3	Gửi attached file (<= 3MB)	100%	Hà Hiếu
4	Tải Email từ server về client	100%	Hà Hiếu, Gia Huy, Duy Bảo
5	Quản lý trạng thái Email	100%	Hà Hiếu
6	Xử lý lọc Email	100%	Hà Hiếu, Duy Bảo
7	Tự động tải Email theo cấu hình	100%	Hà Hiếu, Gia Huy
8	Tổ chức file config	100%	Hà Hiếu
9	Báo cáo	100%	Hà Hiếu, Gia Huy, Duy Bảo

Bảng 2: Bảng tiến độ dự án Email Client

Tài liệu

[1] David M. Beazley. Pythonnetbinder. http://www.dabeaz.com/python/PythonNetBinder.pdf, 2010.