ĐẠI HỌC BÁCH KHOA TP. HỒ CHÍ MINH
KHOA KHOA HỌC & KỸ THUẬT MÁY TÍNH

MÃ HÓA TẬP TIN CÁ NHÂN

Assignment 1

Họ và tên	MSSV
Bùi Phi Long	1712001

Ho Chi Minh, 5/2020

Mục lục

1 Tóm tắt

Báo cáo trình bày sơ lược lý thuyết, kết quả phân tích đề bài, quá trình hiện thực với những vấn đề gặp phải, tổng hợp các kết quả và kết luận để đưa đến một góc nhìn tổng quan hơn về điểm mạnh và điểm yếu của hệ thống.

2 Giới thiệu

2.1 Ứng dụng SET

Mã hóa là phương pháp giúp bảo vệ dữ liệu quan trọng cho người dùng, là những loại dữ liệu chỉ để dành cho những người được phép đọc. **SET** (Secret things) ra đời nhằm mục đích giúp cho người dùng không có chuyên môn cũng có thể thực hiện mã hóa dữ liệu. Đáp ứng nhu cầu về tốc độ và tính an toàn.

2.2 Phạm vi ứng dụng

SET ứng dụng các giải thuật mã hóa nổi tiêu biểu. Mục đích là mang lại trải nghiệm tốt nhất dành cho người dùng khi sử dụng, đáp ứng hầu hết nhu cầu về các loại tập tin cần được mã hóa.

Các định dạng tập tin đã được kiểm chứng được đề cập tại mục: **6.1**

3 Nội dung công việc

3.1 Cơ sở lý thuyết

Dưới đây là sơ lược về lý thuyết của các thuật toán mã hóa được ứng dụng:

3.1.1 DES (Data Encryption Standard)

DES là một phương pháp bảo mật được FIPS (Tiêu chuẩn xử lý Thông tin Liên Bang Hoa Kỳ) chọn làm chuẩn chính thức vào năm 1976.

Đặc điểm:

> DES là thuật toán mã hóa khối, độ dài mỗi khối là 64bit (gọi là plaintext)

> Khóa dùng trong DES có độ dài là 64bit (gọi là K)

> DES xuất ra bản mã 64bit (gọi là ciphertext)

> DES là thuật toán được thiết kế để chạy trên phần cứng

Quá trình sinh khóa: Khóa trong DES có độ dài 64bit, tuy nhiên chỉ có 56bit được

sử dụng để tính toán khóa vòng, 8 bit còn lại dùng cho việc kiểm tra
(1) Phân tách chuỗi khóa: Phân tách chuỗi khóa thành hai phần, đoạn C0(28bit) và D0 (28bit)
(2) Hoán vị chuỗi khóa: Mỗi nhánh C và D sẽ có một bảng hoán vị mở đầu, cần thực hiện hoán vị trước khi sinh khóa vòng.
(3) Vòng lặp: Tại mỗi lần lặp, lấy được một khóa tương ứng cho vòng lặp tại quá trình mã hóa. Các chuỗi C và D được lấy bằng cách thức hiện phép dịch cho chuỗi C và D của vòng lặp trước.
(4) Hoán vị sinh khóa: Sinh khóa vòng cho mỗi bước.

Quá trình mã hóa:
(1) Hoán vị khởi tạo block: Ở bước này, các bit trong plaintext sẽ bị thay đổi vị trí dựa vào bảng IP của DES, **chỉ thay đổi vị trí, không thay đổi giá trị**.
(2) Phân tách chuỗi hoán vị: Sau khi được hoán vị, chuỗi sẽ được phân tách ra thành hai đoạn, mỗi đoạn có chiều dài 32bit L(1→32), R(33→64).
(3) Bắt đầu quá trình lặp: Với mỗi chuỗi L và R được phân tách, thực hiện vòng lặp 16 lần. Sau mỗi vòng lặp, chuỗi L sẽ là chuỗi R của vòng lặp trước đó, chuỗi R hiện tại sẽ được tính bằng hàm mã hóa f(R,K) và chuỗi L của vòng trước đó.

$$\mathrm{Ln{+}1} = \mathrm{Rn}$$
$$\mathrm{Rn{+}1} = \mathrm{Ln} \oplus f(\mathrm{Rn},\mathrm{Kn{+}1})$$

(4) Hàm mã hóa f(R,K): Sử dụng lần lượt các quá trình mở rộng, XOR, hoán vị để trả về một chuỗi có độ dài 32bit.
(5) Hoán vị khởi tạo đảo: Bước cuối cùng để lấy được chuỗi mã hóa 64bit

3.1.2 3DES (Triple Data Encryption Standard0

DES là một thuật toán mã hóa tốt, tuy nhiên trong trường hợp hacker đánh chặn được một cặp (ciphertext,plaintext). Với không gian khóa có độ lớn 2^56 thì các siêu máy tính có thể giải quyết được vấn đề phá khóa, do đó 3DES ra đời.
3DES thực chất chỉ là việc chúng ta thực hiện 3 lần thuật toán DES với các khóa khác nhau, nâng độ phức tạp bảo mật lên tới 2^{112}.

3.1.3 AES (Advanced Encryption Standard)

AES là một phương pháp mã hóa được công bố vào năm 1998 và được sử dụng trên phạm vi toàn thế giới.
Đặc điểm:

> AES là thuật toán mã hóa khối, độ dài mỗi khối là 128bit

> Khóa dùng trong AES có thể có độ dài là 128bit, 192bit hoặc 256bit.

> AES xuất ra bản mã có độ dài 128bit

> AES được thiết kế để chạy trên phần cứng mà không sử dụng quá nhiều vùng nhớ.

> AES xây dựng các chuỗi đầu vào và khóa như một ma trận, qua đó các thao tác được thực hiện trên cột của ma trận.

Quá trình sinh khóa: Quá trình sinh khóa trong AES ứng dụng quá trình tạo khóa vòng của Rijndeal, thao tác trên cột với các phép dịch và kết hợp nhân ma trận, tạo ra một khóa mới

Quá trình mã hóa: Sau quá trình sinh khóa, mỗi khóa sẽ được ứng dụng vào công việc mã hóa theo các bước sau.

Một số thủ tục chính:

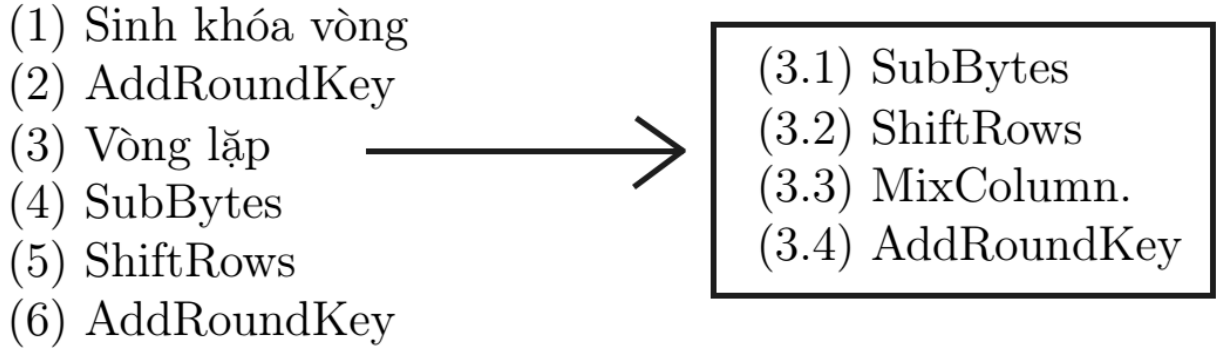
• AddRoundKey: Các khóa con được kết hợp cùng với các khối, XOR các từng bit của khóa với khối dữ liệu tại cùng vị trí trong mỗi ma trận, tạo ra dữ liệu tại vị trí tương ứng trong ma trận mới.

• SubBytes: Các bytes được thay thế thông qua một bảng tra (S-box). Bảng S-box được xây dựng mang tính chất phi tuyến để chống lại các tấn công đặc tính đại số. Tạo sự an toán cho thuật toán mã hóa

• ShiftRows: Các hàng tương ứng trong khối dữ liệu được dịch một khoảng (dịch trái), khoảng này được đo bằng số thứ tự của hàng đó: Khoảng = Chỉ số hàng - 1.

• MixColumns: Mỗi cột từ khối dữ liệu đầu thực hiện phép nhân ma trận để tạo nên một cột mới cho khối dữ liệu đầu ra, góp phần tạo nên tính khuếch tán cho thuật toán.

Các bước mã hóa



3.2 Ngôn ngữ lập trình

Ngôn ngữ lập trình sử dụng: Python
Các gói hỗ trợ: Tkinter (Tạo giao diện) và PyCryptodome (Nhân thuật toán).

4 Phân tích yêu cầu hệ thống

Mục đích sử dụng của SET là mã hóa các tập tin dữ liệu cho người dùng. Do đó, cần có một giao diện tương tác phù hợp, một phương pháp mã hóa hỗ trợ đầy đủ và cần thiết cho khách hàng có thể sử dụng.

Định dạng tập tin cần mã hóa có thể là bất kì loại: hỉnh ảnh, âm thanh, video,.... Hệ thống không thể làm mà chỉ hướng đến một trong số các loại tập tin trên, mà phải là tất cả. Mỗi loại tập tin sẽ có một cấu trúc khác nhau, bắt buộc phải tìm được một cách chung nhất cho tất cả các loại. **Thao tác ở mức bytes(bit)** sẽ đáp ứng được nhu cầu về tính đa dạng đầu vào.
Các định dạng đã được kiếm chứng được liệt kê tại phần **6.1**

4.1 Yếu tố bên trong

4.1.1 Thuật toán

SET là một ứng dụng chạy trên máy của người dùng, các thao tác từ mã hóa đến giải mã đều trên máy tính cá nhân. Kéo theo đó là việc các thông tin cần thiết cho bộ mã hóa, giải mã (những thông tin tuyệt mật) đôi khi có thể vì sự sơ ý của khách hàng mà để lộ ra, làm cho một cách mã hóa bị nắm bắt cấu trúc. Khi người dùng nhận được cảnh báo thì có thể lựa chọn sang một loại thuật toán mã hóa khác được SET cung cấp để đảm bảo tính an toàn. Tuy nhiên, trường hợp này được đảm bảo xảy ra với tỉ lệ cực kì thấp,các điều kiện tiêu chuẩn phải đáp ứng ở mức độ cao nhằm mục đích hạn chế trường hợp lộ thông tin.

4.1.2 Cấu trúc đầu vào

Yêu cầu đầu vào của các thuật toán trên là dữ liệu gốc và khóa:
Dữ liệu gốc:

SET cung cấp hai loại dữ liệu gốc đầu vào: Đơn lẻ hoặc Thư mục. Hai loại dữ liệu gốc trên đều không phụ thuộc định dạng tập tin được chọn, ngoài ra, để đảm bảo sự dễ thao tác cho người dùng. Các tập tin dữ liệu gốc sẽ không được dồn nén. Mỗi tập tin có một khóa riêng biệt nhằm mục đích an toàn.

Khóa:

SET cung cấp hai loại khóa đầu vào: Người dùng tự thiết lập hoặc hệ thống tự sinh khóa. Trường hợp người dùng tự thiết lập sẽ có rất nhiều ràng buộc cần đáp ứng, do đó tính năng tự sinh khóa ra đời đáp ứng nhu cầu của khách hàng không chuyên.

Cần phải xác định được cặp đối tượng: Khóa $\leftrightarrow$ Tập tin. Cấu trúc tập tin khóa được mô tả rõ tại mục **4.3.3**

Bởi vì SET sử dụng nhiều thuật toán mã hóa, mỗi thuật toán mã hóa lại có yêu cầu khác biệt về dữ liệu khóa đầu vào, và mỗi loại thuật toán đó cũng có các ràng buộc cụ thể, các ngoại lệ không cho phép vẫn có thể xảy ra.

4.1.3 Cấu trúc đầu ra

Dữ liệu đầu ra:

Dữ liệu đầu ra mã hóa cần được lưu vào một ví trị cụ thể được xác định bởi người dùng, SET yêu cầu người dùng chọn thư mục để lưu trữ dữ liệu đầu ra. Ngoài ra, tập tin mã hóa cũng sẽ có những ký tự tương tự như tập tin đầu vào để người dùng nhận biết.

Khóa đầu ra:

Trường hợp này chỉ xảy ra khi người dùng muốn hệ thống sinh khóa tự động, người dùng phải lựa chọn thư mục lưu trữ tập tin khóa được sinh. Tên của tập tin khóa sẽ là thời gian mà người dùng chạy giải thuật.

4.1.4 Xác thực kết quả mã hóa

Trong quá trình sử dụng, có thể các tập tin mã hóa dữ liệu bị thay đổi ở một vùng nào đó. Dù là vô tình hay hữu ý thì hệ thống mã hóa đều không được phớt lờ, và cũng để đảm bảo cho người dùng sử dụng một tập tin mã hóa đạt chất lượng tốt. Cần đưa vào các yếu tố xác thực, SET có hai yếu tố xác thực tính án toàn của tập tin giải mã.

> Cờ hiệu (Flag): Có chiều sài là một số bytes nhất định, được đưa vào file mã hóa nhằm mục đích nói rõ quan điểm: "Chỉ những người nắm bắt được đâu là đối tượng đã mã hóa, thì họ mới có thể giải mã".

> Băm (Hash): Một phương thức xác thực nữa cũng được SET sử dụng cho việc đảm bảo tính bất biến, đó là băm các dữ liệu đầu vào (chưa mã hóa) và ghi vào trong tập tin mã hóa.

Tóm lược:

Như vậy, trong mỗi tập tin mã hóa sẽ bao gồm các loại dữ liệu: (Cờ hiệu, Băm, Dữ liệu gốc), tiếp theo sẽ là phân tích tính đảm bảo an toàn khi tập tin mã hóa được phân bổ như trên.

• Cờ hiệu: Trường hợp hacker cố gắng giải mã nhưng nắm bắt sai vị trí của cờ hiệu, tập tin được giải mã sẽ bị loạn và không tạo ra được tập tin gốc của người dùng.

• Chuỗi băm: Nếu chuỗi băm thay đổi thì chứng tỏ dữ liệu đã bị thay đổi, không còn an toàn nên sẽ cảnh báo người dùng, người dùng có thể tiếp tục sử dụng dữ liệu đã bị thay đổi đó hoặc không.

Trường hợp hacker sửa đổi tập tin mã hóa, tùy thuộc vào vị trí sửa đổi sẽ vi phạm hai lỗi ở trên: Thay đổi vào vị trí của cờ hiệu sẽ không thể giải mã, thay đổi vào vị trị chuỗi băm hoặc nội dung sẽ làm sai lệch băm.

4.2 Yếu tố bên ngoài

Các yếu tố bên ngoài ảnh hướng lớn đến hiệu suất làm việc của ứng dụng. Có những tập tin đặt nặng vấn đề thời gian thực thi hơn tính an toàn tuyệt đối. Mặt khác, lại có những tập tin cần tính tuyệt đối đó. Yêu cầu đặt ra là phải sử dụng khả năng của máy tính sao cho đạt hiểu quả tốt nhất.

4.2.1 Không gian vùng nhớ

Không gian vùng nhớ chính của máy tính luôn là hữu hạn, nhỏ hơn rất nhiều so với vùng nhớ thứ cấp. Việc có một tập tin dữ liệu có kích thước lớn hơn số lượng vùng nhớ khả dụng là trường hợp hoàn toàn có thể xảy ra, kéo theo sự sụt giảm tốc độ thực thi (đối với hệ điều hành tự động phân bổ vùng nhớ thứ cấp hỗ trợ) hoặc gây đổ vỡ ứng dụng. Khi quản lý vùng nhớ không tốt sẽ gây ảnh hưởng xấu đến kết quả, mục tiêu đặt ra thuật toán phải đáp ứng được nhu cầu vùng nhớ khả dụng.

4.2.2 Thời gian xử lý

Thời gian là một yếu tố có thể kéo dài được. Tuy nhiên, khi thời gian quá lâu cũng sẽ gây ảnh hưởng đến người dùng, hệ thống cần báo cáo tiến độ mã hóa đối với người dùng. Mặt khác, với những tập tin yêu cầu mã hóa nhanh, ứng dụng cũng phải đáp ứng. Do đó, yêu cầu về tối ưu hóa ứng dụng là quan trọng.

4.2.3 Độ chính xác và tính bất biến

Tất cả các yếu tố về thời gian và không gian có tối ưu đến đâu, mục tiêu quan trọng nhất của ứng dụng mã hóa là phải đảm bảo tính chính xác và khó phá vỡ của tập tin. Đòi hỏi một quá trình kiểm thử đầy đủ cho các trường hợp gây lỗi hệ thống.

4.3 Yêu cầu cấu trúc dữ liệu

4.3.1 Trạng thái đầu vào

SET là ứng dụng bao gồm nhiều trạng thái đầu vào bắt buộc cho việc thực thi bao gồm:
> Thuật toán: Yêu cầu đầu tiên và quan trọng phục vụ việc thực thi.
> Phương thức: Có hai dạng trong vấn đề bảo mật là mã hóa và giải mã.
> Tập tin đầu vào:
+ Hình thức tập tin: Kiểu dữ liệu đầu vào có thể là một tập tin hoặc một thư mục.
+ Đường đẫn: Đường dẫn đến tập tin/thư mục đã được chọn.
> Khóa đầu vào:
+ Hình thức khóa: Xác nhận khóa sẽ được hệ thống tạo hay do người dùng tự tạo.
+ Đường dẫn: Đường dẫn đến tập tin/thư mục lưu trữ khóa
> Thư mục lưu trữ: Cung cấp một thư mục cho việc lưu trữ dữ liệu đầu ra.

4.3.2 Tập tin đầu vào

Để tối ưu việc sử dụng vùng nhớ khả dụng, các tập tin được đọc lần lượt, không nén, không đọc tổng thể. Qua đó vùng nhớ khả dụng sẽ đủ để đáp ứng các nhu cầu làm việc với tập tin có kích thước lớn tiệm cận chính vùng nhớ đó. Bởi lý do trên, danh sách tập tin đầu vào được lưu thành các đường dẫn, ví dụ:
File: `E:\University\192\CANS\Assignment\Ass1\SETApplication\Test\trytoLearn`
Folder: `E:\University\192\CANS\Assignment\Ass1\SETApplication\Test`

4.3.3 Khóa đầu vào

Cấu trúc khóa trong tập tin khóa có dạng: Name=Key, các giá trị khi được ghi vào tập tin được hoán vị, khiến người thường không thể đọc. Dưới đây là ví dụ về câu trúc.

```
key.txt     =12345678
music.mp3   =abcdefgh
```

Chiều dài khóa

Thuật toán	DES	3DES	AES
Khóa	8(bit)	24(bit)	16(bit)

Khóa có thể là bất kì biểu diễn bytes nào, khi đọc từ tập tin khóa. Chiều dài của khóa sẽ được tính từ ngay sau dấu '=' đến hết dòng. Mỗi loại thuật toán đều có chiều dài khóa khác nhau, đảm bảo chiều dài khóa (không thừa không thiếu).

Số lượng khóa

Có ba kiểu số lượng khóa trong SET:

> Chỉ có 1 khóa dành cho tất cả: ví dụ cấu trúc "all=12345678", đối tượng all ám chỉ tất cả các tập tin trong danh sách mã hóa hoặc giải mã đều sử dụng khóa là 12345678.

> Số lượng khóa bằng số lượng danh sách đầu vào: Cấu trúc khóa "filename=key", đảm bảo yêu cầu về số lượng SET mới có thể thực hiện các giải thuật rõ ràng, tường minh và chính xác.

> Trường hợp cây thư mục, các tập tin cùng tên cũng phải được mã hóa theo khóa riêng.

4.4 Các loại vi phạm

Các loại vi phạm cần được xử lý và điều chỉnh để thông báo lỗi đến người dùng, hoặc dừng chương trình nếu vi phạm đạt ở mức cao.

4.4.1 Vi phạm về trạng thái

Trường hợp người dùng chưa điền đầy đủ các trạng thái thông tin nhưng vẫn yêu cầu thực thi thuật toán. Hệ thống buộc phải tạm ngưng và trả về yêu cầu cho người dùng.

4.4.2 Vi phạm về đọc ghi tập tin

Vi phạm về tập tin nói đến vấn đề khi đọc và ghi các tập tin dữ liệu rời vào trường hợp tập tin đó bị xóa ngang khi đang quá trình xử lý, hoặc đã bị đưa thông tin giả.

Python: Except FileNotFoundError

4.4.3 Vi phạm về khóa trong mã hóa

Trong mã hóa, mỗi giải thuật sẽ có độ dài khóa khác nhau, **Tuy nhiên, trường hợp này hệ thống sẽ xử lý thay cho người dóaùng.**

Python: Except ValueError

4.4.4 Vi phạm về cấu trúc tập tin khóa

Tập tin khóa có những cấu trúc đòi hỏi người dùng phải tuân theo nếu muốn tự sinh khóa hoặc do một tên không tồn tại trong danh sách đầu vào.Mọi sai lệch về cấu trúc khóa đều kéo theo sự sai lệch của một chuỗi các hoạt động mã hóa khác. **Python: Except ValueError** Cấu trúc của một tập tin khóa được mô tả tại phần: **4.3.2**

4.4.5 Vi phạm về số lượng khóa

Ngoài việc có cấu trúc, số lượng các khóa cũng phải tuân theo các quy tắc. Sai phạm về số lượng khóa sẽ dẫn đến việc hệ thống không thể tìm ra tập tin hoặc mã hóa không đúng.

Python: Except ValueError

Quy tắc về số lượng khóa được mô tả tại phần: **4.3.2**

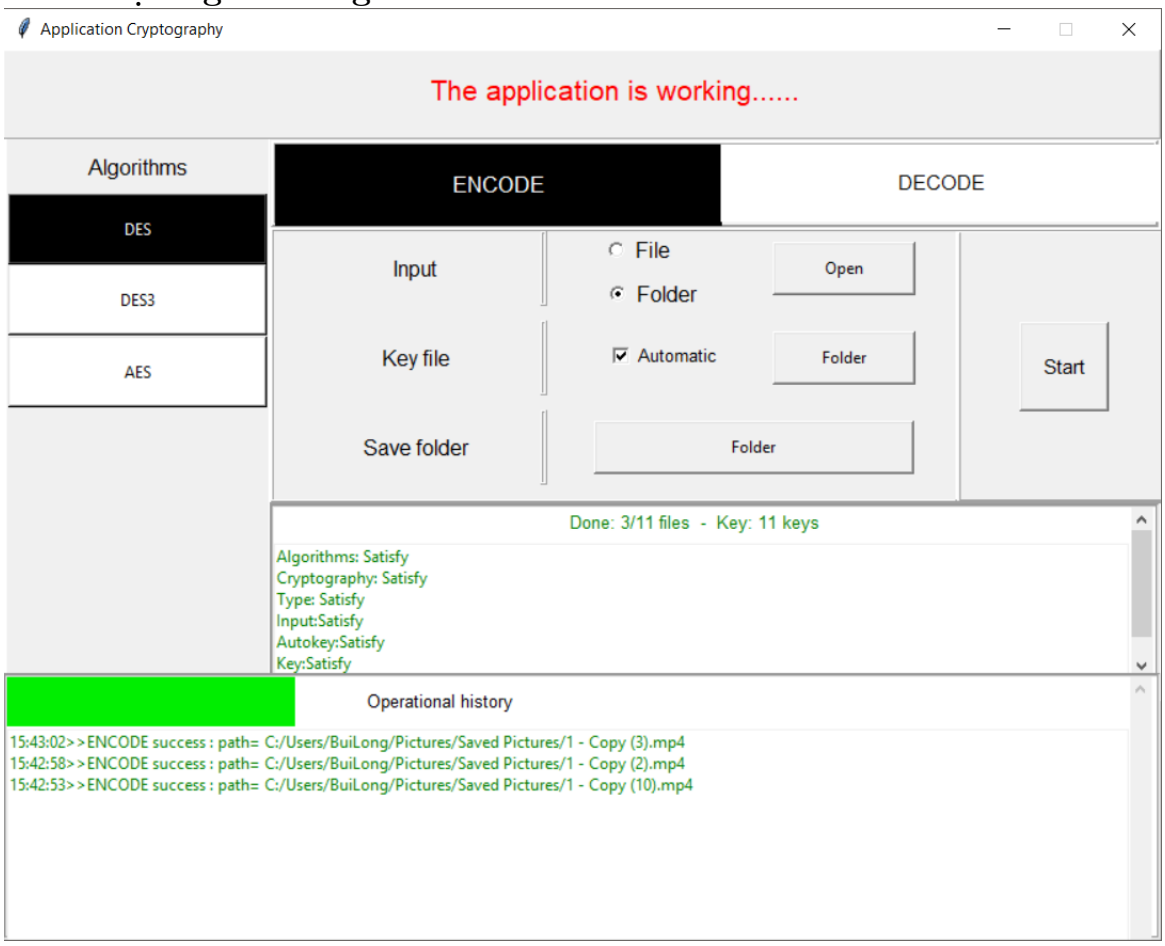
5 Hiện thực

Mô tả sơ lược về quá trình hiện thực ứng dụng SET, các ván đề gặp phải và cách giải quyết.

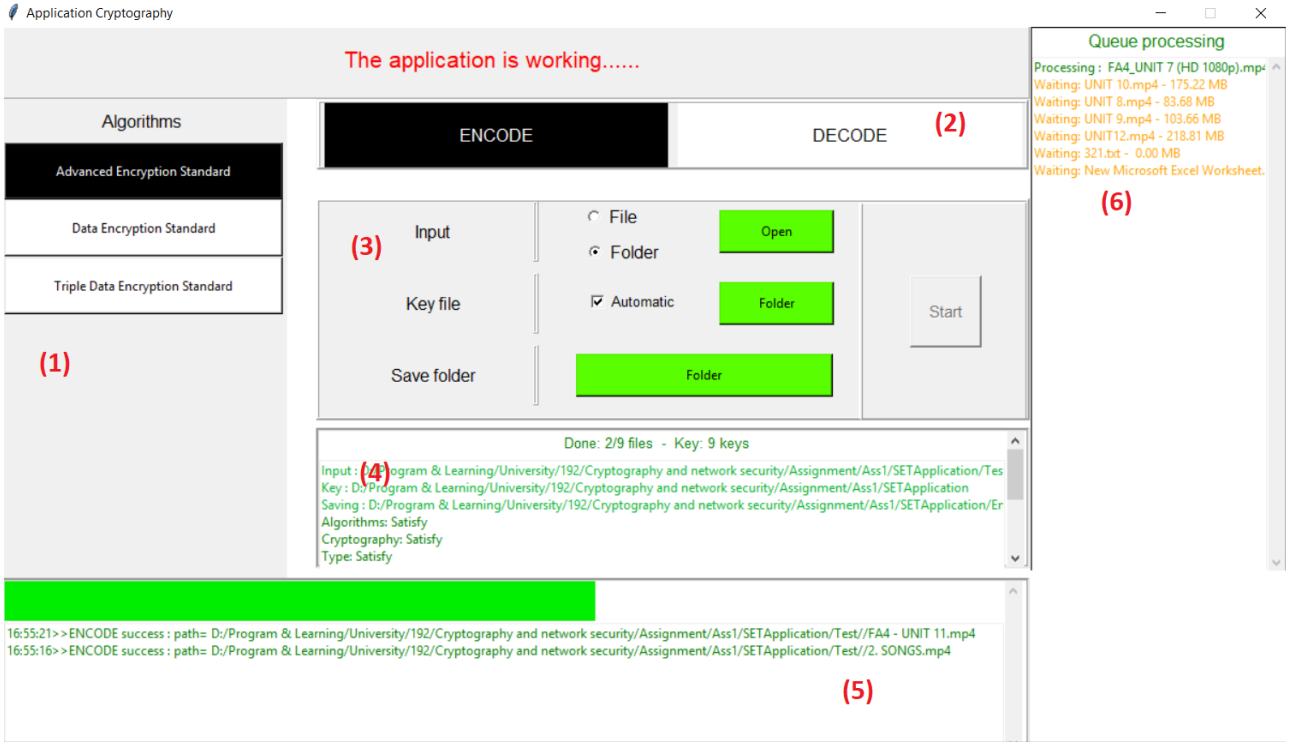
5.1 Giao diện người dùng

Sử dụng gói Tkinter của python để xây dựng giao diện cho người dùng, đồng thời cũng để tương tác với nhân của thuật toán.

Giao diện người dùng:



Mô tả giao diện



(1) Khu vực lựa chọn thuật toán mã hóa muốn sử dụng.

(2) Vùng lựa chọn hình thức thực thi, mã hóa hoặc giải mã.

(3) Vùng trung tâm chính của ứng dụng, bao gồm:

> Danh sách đầu vào, có hai RadioButton cho người dùng chọn định dạng muốn mã hóa: theo từng file hoặc tất cả file trong folder.

> Khóa mã hóa: Checkbox lựa chọn hệ thống tự tạo mã khóa và lựa chọn thư mục lưu file hoặc mở tập tin chứa khóa.

> Thư mục lưu: Lựa chọn thư mục lưu trữ các kết quả sau thực thi.

> Start: Bắt đầu gửi dữ liệu cho hệ nhân thuật toán xử lý (gọi là **ClientData**).

(4) Vùng thông báo trạng thái của quá trình thực thi: Lỗi hoặc thành công.

(5) Vùng lịch sử: Bao gồm lịch sử hoạt động và trạng thái hoàn thành tiến trình.

Các khu vực cho phép người dùng cập nhật thông tin trong thực thi, cùng với việc đảm bảo các state (mục **4.3.1**) được đáp ứng đầy đủ. (6) Vùng chứa danh sách hàng đợi chờ thực thi, vùng này chỉ xuất hiện khi hệ thống đang xử lý yêu cầu mã hóa/ giải mã.

5.2 Cấu trúc dữ liệu

Trạng thái điều khiển: Trạng thái quản lý các yêu cầu từ người dùng.

[Algorithms,CryptoType,InputType,InputUrl,KeyControl,KeyUrl,SavingUrl]

Cấu trúc danh sách tập tin: Là một danh sách, chứa bên trong một đối tượng.

[url1,url2,url3,...]

Cấu trúc danh sách khóa: Là một danh sách, chứa bên trong một đối tượng.

[(name1,url1),(name2,url2),(name3,url3),...]

Các cấu trúc được mô tả tại mục **4.3** Vấn đề

5.3 Thuật toán

5.4 Thư viện mã hóa

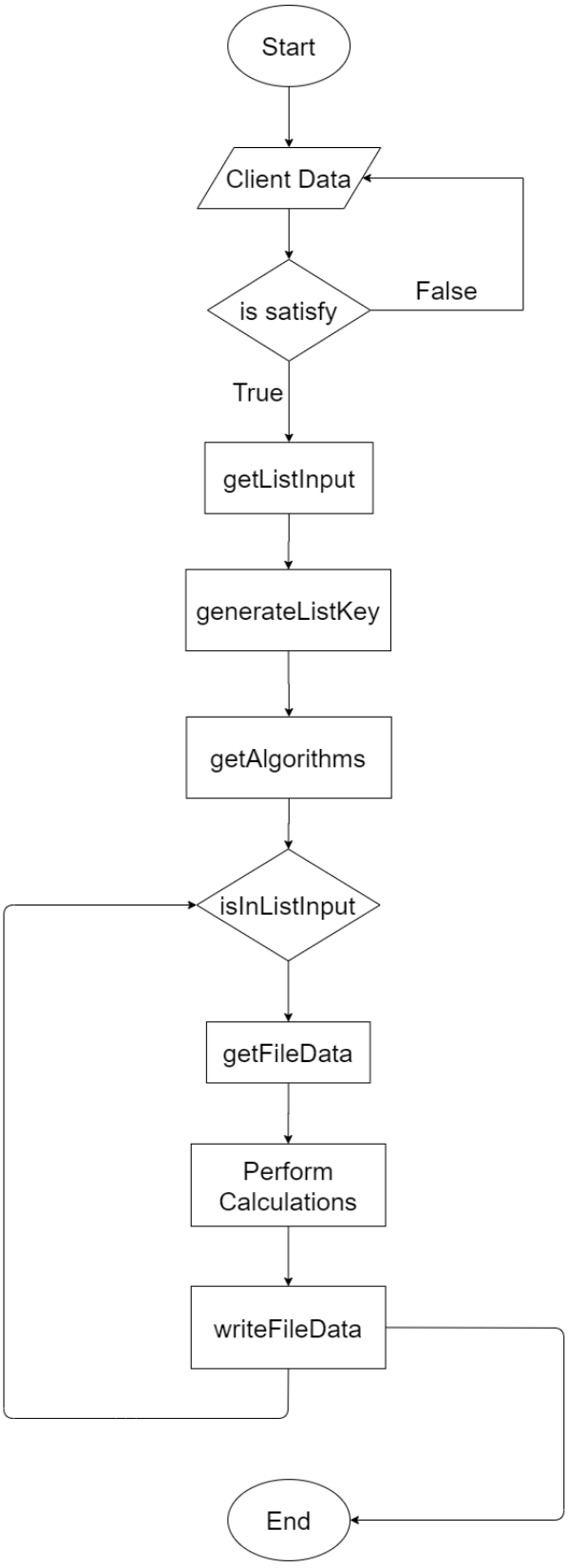
Thư viện PyCryptodome được ứng dụng làm gốc mã hóa. Mô tả cơ bản về quá trình làm việc ứng dụng thư viện như sau:
Phân loại: Trong quá trình ứng dụng thư viện, chỉ có hai thao tác chính để tương tác là mã hóa(Encrypt) và giải mã (Decrypt). Hai thao tác trên khác nhau về đầu vào và đầu ra. Cụ thể.

Name	Input	Exception	Output
Encrypt	PlainData(Bytes), Key(Bytes)	4.4.3	HashData(Bytes), CipherData(Bytes), Flag(Bytes)
Decrypt	CipherData(Bytes), Flag(Bytes), Key(Bytes),	4.4.3	PlainData(Bytes), newHash(Bytes)

Như đã phân tích ở trên, các giá trị đầu ra của bước mã hóa sẽ là giá trị đầu vào của bước giải mã, các giá trị được ghi trong cùng một tập tin mã hóa.
- Cờ định danh giải mã đúng (Flag).
- Kết quả băm so sánh(HashData & newHash).
- Kết quả đầu ra(CipherData, PlainData).

5.5 Luồng thực thi chính

Nhân quá trình bắt đầu làm việc khi người dùng gọi lệnh hành động. Thư viện PyCryptodome được ứng dụng để làm nhân của thuật toán mã hóa (AES, DES, 3DES).



> **ClientData**: Dữ liệu được lựa chọn bởi người dùng

> **isStatisfy**: Đây là một phần mở đầu quan trọng, Xác định người dùng đã cung cấp các thông tin yêu cầu. Trường hợp không thỏa yêu cầu: quay lại yêu cầu người dùng cung cấp đủ thông tin.

> **getListInput**: Lấy ra danh sách các tập tin cần được mã hóa. Vì vấn đề không gian vùng nhớ, chỉ lấy ra các đường dẫn (path) đến thư mục. Dữ liệu trong file được đọc khi cần thiết, mục đích tiết kiệm vùng nhớ. Danh sách tập tin cần có trước để so điều kiện danh sách khóa.

> **generateListKey**: Tạo ra danh sách các khóa cho việc mã hóa. Thực hiện quá trình sinh khóa tự động hoặc đọc tập tin chứa khóa.

> **getAlgorithms**: Lấy loại thuật toán mã hóa tương ứng với lựa chọn của người dùng, API của PyCryptoDome được gọi trong chương trình con này, hướng đối tượng ứng dụng giải quyết vấn đề.

> Lặp với từng đường dẫn(path) trong danh sách đầu vào:

(1) Lấy dữ liệu tập tin theo đường dẫn.

(2) Lấy khóa từ danh sách khóa.

(3) Chạy thuật toán mật mã.

(4) Ghi ngược kết quả ra tập tin mới.

Xử lý trích xuất chương trình con:
Hệ thống nhân cũng có những ngoại lệ (Exception) xảy ra khi các điều kiện đầu vào bị vi phạm. Sau đây là bảng chân trị thể hiện sự tương quan giữa đầu vào và đầu ra khi vào các trương chình con.

Name	Input	Exception	Output
isStatisFy	userState(List)	None	control(True/False)
getListInput	path(String)	4.4.2	dataField(List)
generateListKey	path(String), lengDataField(int)	4.4.2, 4.4.4, 4.4.5	keyField(List)
getAlgorithms	algori(Int)	None	algorithms(Function)
getFileData	path(String)	4.4.2	fileData(bytes)
getKeyValue	nameObject(String)	4.4.4	key(bytes)
performCalculations	fileData(bytes), key(bytes)	4.4.3	cryptoData(bytes)
writeFileData	cryptoData(bytes)	4.4.2	None

Giá trị trả về của chương trình còn được thực hiện trước, trải qua một số thao tác nhỏ sẽ trở thành giá trị đầu vào của chương trình con thực hiện sau, tạo thành một thể thống nhất từ trạng thái đầu vào (người dùng) đến trạng thái kết thúc. Trong quá trình thực thi, mỗi khi có một trạng thái tạo thành mã lỗi thuộc phạm vi của ngoại lệ. Chương trình đáp ứng ngay yêu cầu dừng, sau đó đặt mã lỗi vào luồng gửi về phía giao diện thông báo.

5.6 Vấn đề hiện thực

5.6.1 Luồng thực thi

Sử dụng tkinter cùng python core để gây dựng nên SET. Khi hệ thống thuật toán đang trong quá trình tính toán, phần giao diện người dùng bị cứng do không còn luồng thực thi nào cho giao diện nữa. Yêu cầu đặt ra là cần phải phân tách được giao diện và nhân thuật toán ra thành hai phần chạy trên hai luồng riêng biệt.

5.6.2 Tối ưu hóa yêu cầu

Đánh giá quá trình hiện thực, trường hợp giải mã cả một thư mục khi nén các tập tin lại sẽ giúp đơn giản cho quá trình mã hóa. Tuy nhiên, khi cần mã hóa một tổ hợp quá nhiều tập tin có kích thước lớn, hệ thống không thể đáp ứng. Do đó thuật toán cần thay đổi và làm việc tuần tự với từng tập tin.

6 Phân tích và đánh giá

6.1 Kết quả

SET đã đáp ứng được các yêu cầu đặt ra trong quá trình phân tích, các quy định về tính đồng nhất và an ninh cho quá trình mã hóa và giải mã,

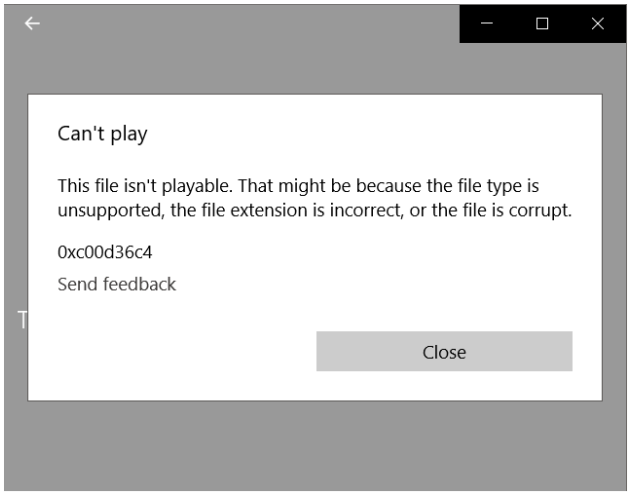
Thay đổi dữ liệu tập tin mã hóa: Kiểm thử trường hợp thay đổi dữ liệu tập tin mã hóa, thay đổi một bytes nào đó trong file.
Kết quả thông báo người dùng:

Please becareful, you files had been changed

Thay đổi khóa: Kiểm thử trường hợp khóa được sử dụng một cách ngẫu nhiên. Thay đổi khóa sẽ dẫn đến việc dữ liệu gốc bị thay đổi sau giải mã, kéo theo đó là kết quả băm cũng bị thay đổi:

Please becareful, you files had been changed

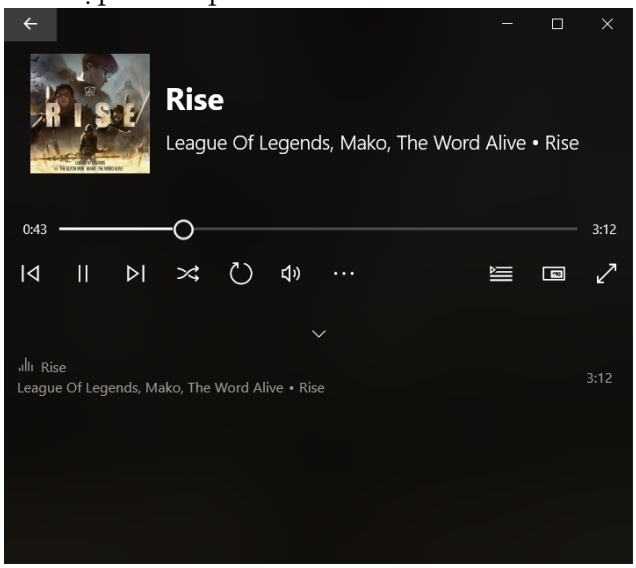
Hai tập tin mp4 giải mã đúng khóa và giải mã sai khóa:

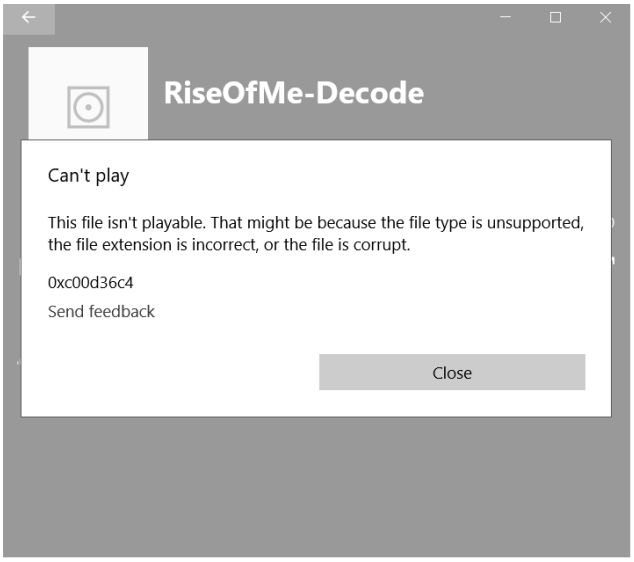


Thay đổi cấu trúc thư mục mã hóa: Trường hợp cấu trúc thư mục lưu các tệp mã hóa bị thay đổi (tên tập tin, xóa, thêm,....) đồng thời cũng phải cập nhật lại trong tập tin chứa khóa. Nếu điều này không được thực hiện, mã lỗi được gửi về người dùng thông báo sự thay đổi đó.

Nắm bắt sai cấu trúc: Mỗi tập tin mã hóa đều có một cấu trúc tổ chức chặt chẽ, nắm bắt sai lệch vị trí, dù chỉ là 1 bit cũng đưa ra kết quả không chính xác. Yêu cầu về cấu trúc xác thực tập tin được nhắc tới tại mục **4.1.3**

Hai tập tin mp3 nắm bắt sai cấu trúc:





Các định dạng tập tin đã được kiểm thử:

.png	.jpeg	.jpg	.3fr	.ari	.arw	.bay	.crw
.cr2	.cap	.data	.dcs	.dcr	.dng	.drf	.eip
.erf	.fff	.iiq	.k25	.kdc	.mdc	.mef	.mos
.mrw	.nef	.nrw	.obm	.orf	.pef	.ptx	.pxn
.r3d	.raf	.raw	.rwl	.rw2	.rwz	.sr2	.srf
.srw	.tif	.x3f	.gif	.psd	.pdf	.txt	.mp4
.mp3	.mov	.wmv	.avi	.flv	.wma	.wav	.flac
.aac	.ogg	.aiff	.alac	.c	.cpp	.java	.py

6.2 Đánh giá hiệu năng

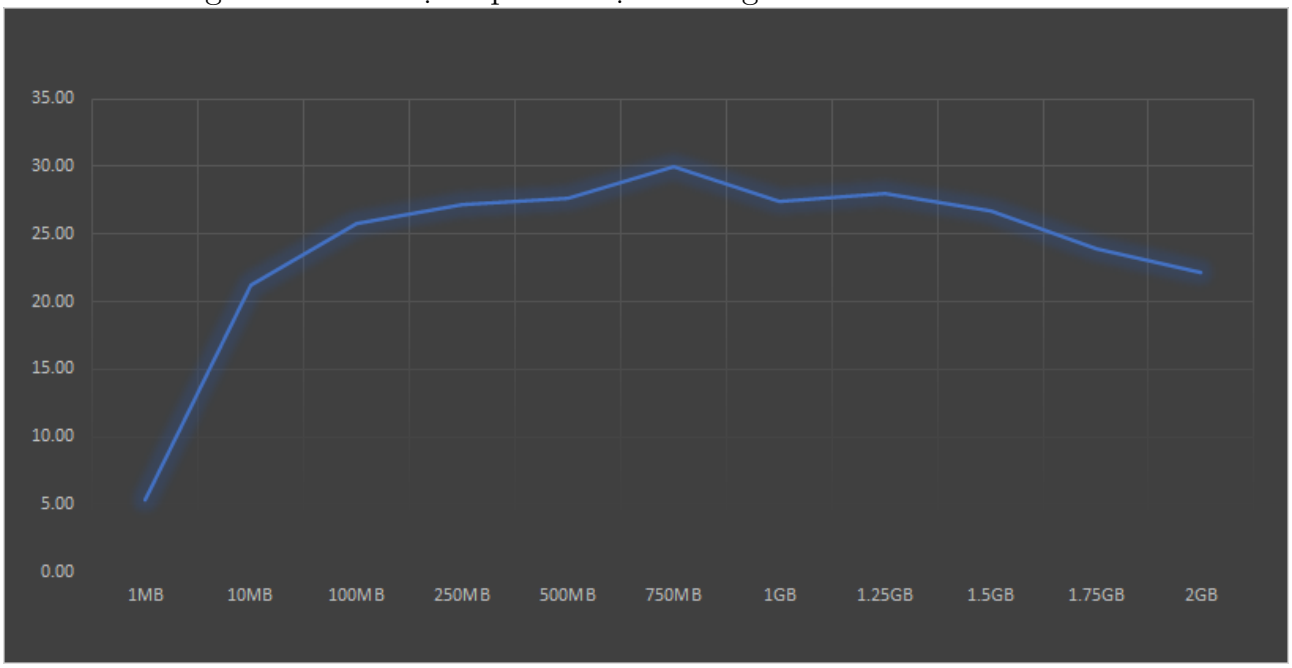
Đảm bảo tính an toàn, trong sáng cho người sử dụng, đáp ứng được phần lớn các yêu cầu của người dùng về mã hóa và bảo mật.

Khả năng ứng dụng:

SET đáp ứng được khả năng mã hóa với không giới hạn định dạng tập tin, đáp ứng nhu cầu mã hóa của người dùng. Các giải thuật mã hóa được ứng dụng trong SET đều được kiểm chứng cực kỳ kỹ lưỡng bởi các tổ chức quốc tế, trong đó có những giải thuật không thể tấn công trực tiếp vào kết quả mã hóa như AES mà chỉ có thể tấn công vùng biên. Ngoài ra còn có những giải thuật đáp ứng cho tốc độ mã hóa.

Thời gian: là một tiêu chí phụ, đánh giá thời gian được thực hiện tương đối.

Tốc độ thực thi chương trình phụ thuộc vào nhiều yếu tố: Hiệu năng máy tính, kích thước tập tin, không gian vùng nhớ, ... Do đó, chúng ta không có ràng buộc nào có thể xác định được mức thời gian. Tuy nhiên, khi tập tin có độ lớn nhất định, tốc độ thực thi của chương trình sẽ ổn định quanh một khoảng.



Tại hai đầu của đồ thị có hiện tượng tăng mạnh và giảm mạnh về tốc độ thực thi, vấn đề này rơi vào các trường hợp công việc thực hiện trên nhân thuật toán nó là nhỏ so với công việc đọc/ghi dữ liệu từ bộ nhớ thứ cấp.

Không gian: ứng dụng đầy đủ khả năng của hệ thống 64bit, hoàn toàn không có giới hạn về bộ nhớ có thể thao tác bằng thuật toán: "Bộ nhớ chính cung cấp được bao nhiêu, tập tin đưa vào mã hóa có thể lớn bấy nhiêu"

Tính chính xác: Tính chính xác đã được nhắc đến nhiều trong phần kết quả, tuy nhiên kết quả sẽ không khách quan khi chỉ sử dụng hệ thống hash của PyCryptodome. Do đó, ứng dụng của bên thứ 3 "Cryptool" sẽ được sử dụng để chứng minh tính toàn vẹn.

Kết quả hiển thị:

Difference between the hash values of the original and of the modified file

```
00000000#00000000#00000000#00000000#00000000#00000000#00000000#00000000#
00000000#00000000#00000000#00000000#00000000#00000000#00000000#00000000#
0.00% of the bits differ (0 of 128).
Longest identical bit sequence: offset 0, length 128.
```

SET đưa ra kết quả với 0.00% bits sai lệch, tỉ lệ chính xác đạt mức cao nhất đáp ứng nhu cầu của phân tích tại mục **4.1.3**

6.3 Hạn chế

Giới hạn về không gian: SET hoạt động tốt trong trường hợp tập tin đầu vào có kích thước vùng nhớ nhỏ hơn bộ nhớ chính. Tuy nhiên, trường hợp ngược lại SET hoạt động không hiệu quả hoặc có thể bị crash.
Hạn chế của thời gian: Khoảng thời gian thực thi có thể rất lớn với những bài toán có kích thước lớn. Ví dụ, Tập tin có kích thước 2GB cần khoảng 90s thực thi.
Tính tương đối của tốc độ thực thi: Tốc độ thực thi được tính toán một cách tương đối trên mỗi máy tính mà ứng dụng được khởi chạy, bởi chúng ta không thể can thiệp vào nhân của PyCryptodome để lấy chính xác.
Khả năng tương tác người dùng: Giao diện người dùng đạt ở mức dễ hình dùng, dễ điều khiển nhưng chưa thực sự gây ấn tượng với người dùng. Các thức thiết kế cũng chưa đặt tới mục tiêu phản ứng nhanh trên nhiều kích thước màn hình.
Số lượng thuật toán: Set mới chỉ ứng dụng 3 trong số nhiều thuật toán mã hóa tốt hiện nay, điều này chưa thực sự mang đến nhiều sự lựa chọn.

7 Hướng phát triển

Các vấn đề của ứng dụng SET đã được nêu lên ở mục **6.3**, sau đây là những định hướng cải thiện trong thời gian tới.
Không gian: Phân tích và xây dựng một hệ thuật toán cho phép cắt các tập tin có kích thước quá lớn ra và mã hóa lần lượt, đảm bảo tính nhất quán cho hệ thống. Hướng tới phục vụ cá nhân, doanh nghiệp, gia nhập vào thị trường.
Thời gian: SET hiện đang sử dụng hai luồng: chạy giao diện đồ họa và thực thi thuật toán. Trong tương lai, SET sẽ được ứng dụng để chạy với nhiều luồng và quy trình hơn, đạt tốc độ cao trong việc giải mã bằng phương pháp tính toán song song, hỗ trợ cho vấn đề **Không gian** và **Thực thi nhiều tập tin**. Khoảng thời gian phát triển tối ưu hơn cho người dùng.
Tốc độ: Trong tương lai, các thuật toán mã hóa sẽ được SET tự phát triển, kéo theo đó là thông tin về tốc độ thực thi ở từng thời điểm đạt mức chính xác.
Giao diện người dùng: SET ngày một phát triển hệ thống tương tác người dùng. Ngoài việc tạo nên một giao diện khả quan hơn, còn mang mục đích mở rộng chức năng trong các phiên bản sắp tới.
Thuật toán: Cùng với mục đích mở rộng chức năng, số lượng thuật toán cũng được SET nghiên cứu phát triển mở rộng.

8 Kết luận

Sau cùng, Secret Things là ứng dụng được tạo dựng phục vụ mã hóa và đã đáp ứng được yêu cầu của việc mã hóa. Tuy hẵng còn nhiều điểm chưa hoàn hảo trong hệ thống, song SET vẫn đang trong quá trình phát triển từ bên trong lẫn bên ngoài. Ở thời điểm hiện tại, SET vẫn đang là một mầm non.

Tài liệu tham khảo

[1] AES Algorithms:
"**https://en.wikipedia.org/wiki/Advanced_Encryption_Standard**",
Last access: 5/5/2020.

[2] DES Algorithms:
"**https://en.wikipedia.org/wiki/Data_Encryption_Standard**",
Last access: 5/5/2020.

[3] Triple DES Algorithms: "**https://en.wikipedia.org/wiki/Triple_DES**",
Last access: 5/5/2020.

[4] Tkinter Tutorial:
"**https://www.tutorialspoint.com/python/python_gui_programming.htm**",
Last access: 5/5/2020.

[5] PyCryptodome Documentation:"**https://pycryptodome.readthedocs.io/**",
Last access: 5/5/2020

Phụ lục

A Cấu trúc

Trong thư mục được gửi, bao gồm các tập tin sau:
» set.py: File gốc của hệ thống, dùng để quản lý giao diện và nhân xử lý.
» ui.py: File chứa giao diện người dùng của hệ thống.
» alg.py: File chứa nhân thuật toán của hệ thống.
» setx64.exe: File thực thi, xây dựng trên nền tảng 64bit

B Hướng dẫn sử dụng

Sử dụng file thực thi: Trong phần submit có đóng gói sẵn file setx64.exe, là file thực thi riêng trên hệ điều hành windows.

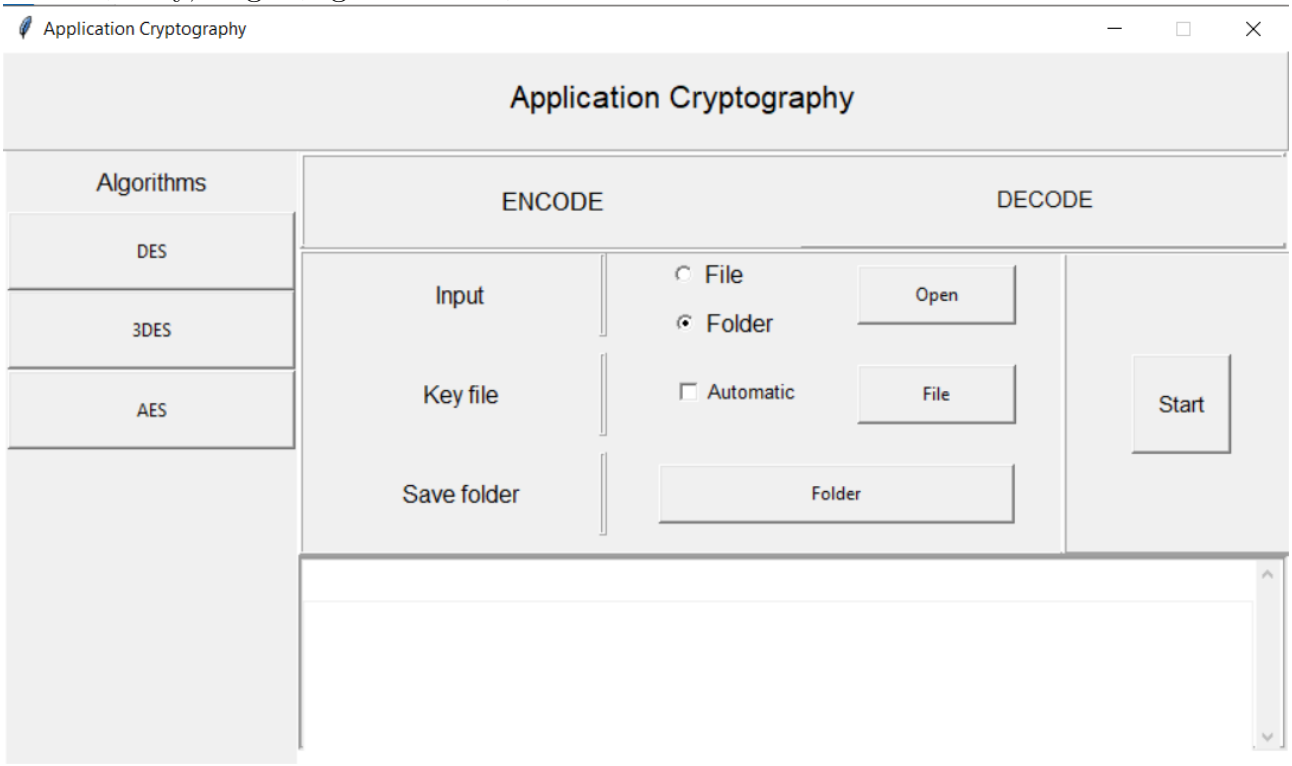
Compile source: Biên dịch hệ thống source code để lấy ra dữ liệu sử dụng.

Các bước:

1» Tải về thư viện PyCryptoDome: *pip install pycryptodome*

2» Chạy lệnh biên dịch: *python3 set.py*

Tại đây, ứng dụng sẽ hiển thị như sau.



Cấu trúc giao diện đã được phân tích tại mục **5.1**: Trong quá trình sử dụng, vì tốc tộ và thời gian thực thi được tính một cách tương đối, lần đầu tiên trong mỗi phiên làm việc của ứng dụng (mỗi lần chạy thực thi) sẽ được sử dụng làm lần căn chỉnh tốc độ đầu tiên. Do đó thanh trạng thái hoàn thành công việc sẽ không di chuyển.

Các yêu cầu về trạng thái được liệt kê tại mục **4.3.1** có thể thực hiện không theo thứ tự, chỉ cần làm trước khi nhấn Start. Trường hợp chưa đủ trạng thái nhấn Start thì sẽ nhận được thông báo lỗi.

Đợi tới khi hệ thống ngừng thông báo đang làm việc là lúc để kiểm tra kết quả thực thi.