ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA CÔNG NGHỆ THÔNG TIN

CTT010 – Nhập Môn CNTT 2

BÁO CÁO

DAMH_02: BẢO MẬT THÔNG TIN

GVHD: Hồ Tuấn Thanh & Lương Vĩ Minh

Nhóm thực hiện: Nhóm 19

Thành phố Hồ Chí Minh – 2019

LỜI CẢM ƠN

Được sự phân công của quý thầy cô trường Đại học Khoa Học Tự Nhiên thành phố Hồ Chí Minh, nhóm chung tôi bắt tay vào thực hiện đồ án có chủ đề **"Bảo mật thông tin"**.
Nhóm chúng tôi gồm bốn thành viên:

- Đoàn Đình Toàn – 18127231
- Huỳnh Đức Lê - 18127126
- Đặng Khánh Sơn – 18127197
- Huỳnh Nguyễn Nhật Quạng Huy – 18127106

Đồ án yêu cầu trình bày tổng quan về các hình thức mã hóa cơ bản và làm một ứng dụng mã hóa chuỗi từ người dùng đưa vào.
Nhân dịp bài luận cuối cùng của môn Thực Hành Nhập Môn Công Nghệ Thông Tin 2, chúng tôi xin bày tỏ lòng biết ơn đến thầy **Hồ Tuấn Thanh,** thầy **Lương Vĩ Minh** đã tận tình hướng dẫn, giúp đỡ chúng tôi kể từ ngày định hướng cho đến khi hoàn thành bài luận này! Đồng thời chúng tôi xin chân thành cảm ơn cô **Lê Thị Nhàn** đã dạy dỗ, trang bị kiến thức cho chúng tôi trong suốt quá trình học tập!

TP HCM, 29/03/2019

Đoàn Đình Toàn
Huỳnh Đức Lê
Đặng Khánh Sơn
Huỳnh Nguyễn Nhật Quang Huy

I. GIỚI THIỆU

Mã hóa là phương pháp để biến thông tin (phim , ảnh , văn bản , hình ảnh….) từ định dạng bình thường sang dạng thông tin không thể hiểu được nếu không có phương tiện giải mã.

Cryptography (hay crypto) – **mật mã học** – ngành khoa học nghiên cứu về việc giấu thông tin. Cụ thể hơn, mật mã học là ngành học nghiên cứu về những cách chuyển đổi thông tin từ dạng "có thể hiểu được" thành dạng "không thể hiểu được" và ngược lại. Cryptography giúp đảm bảo những tính chất sau cho thông tin:

- **Tính bí mật** (confidentiality): thông tin chỉ được tiết lộ cho những ai được phép.
- **Tính toàn vẹn** (integrity): thông tin không thể bị thay đổi mà không bị phát hiện.
- **Tính xác thực** (authentication): người gửi (hoặc người nhận) có thể chứng minh đúng họ.
- **Tính không chối bỏ** (non-repudiation): người gửi hoặc nhận sau này không thể chối bỏ việc đã gửi hoặc nhận thông tin.

Mật mã có rất nhiều ứng dụng trong thực tế như bảo vệ giao dịch tài chính (rút tiền ngân hàng, mua bán qua mạng), bảo vệ bí mật cá nhân… Nếu kẻ tấn công đã vượt qua tường lửa và các hệ thống bảo vệ khác thì mật mã chính là hàng phòng thủ cuối cùng cho dữ liệu của bạn.

Cần phân biệt khái niệm cryptography với khái niệm steganography (tạm dịch là giấu thông tin). Điểm khác nhau căn bản nhất giữa hai khái niệm này là: cryptography là việc giấu nội dung của thông tin, trong khi steganography là việc giấu sự tồn tại của thông tin đó.

Sau đây là các thuật ngữ tiếng anh chuyên ngành sẽ dùng trong báo cáo nhỏ này:

Cryptosystem (viết tắt của cryptographic system): hệ thống mã hóa thông tin, có thể là phần mềm như PGP, Ax-Crypt, Truecrypt… giao thức như SSL, IPsec… hay đơn giản là một thuật toán như DES.

Encrypt (encipher): mã hóa – quá trình biến đổi thông tin từ dạng ban đầu -> có thể hiểu được thành dạng không thể hiểu được, với mục đích giữ bí mật thông tin đó.

Decrypt (decipher): giải mã – quá trình ngược lại với mã hóa, khôi phục lại thông tin ban đầu từ thông tin đã được mã hóa.

Plaintext (cleartext): dữ liệu gốc (chưa được mã hóa).

Ciphertext: dữ liệu đã được mã hóa.

Cipher (hay cypher): thuật toán dùng để thực hiện quá trình mã hóa hay giải mã. Trong khuôn khổ bài viết này gọi tắt là thuật toán.

Key: chìa khóa – thông tin dùng cho qui trình mã hóa và giải mã.

Code: cần phân biệt code trong mật mã học với code trong lập trình hay code trong Zip code… Trong cryptography, code (mã) có ý nghĩa gần như là cipher (thuật toán). Chúng chỉ khác nhau ở chỗ: code biến đổi thông tin ở tầng nghĩa (từ, cụm từ) còn cipher biến đổi thông tin ở tầng thấp hơn, ví dụ chữ cái (hoặc cụm chữ cái) đối với các thuật toán cổ điển hay từng bit (hoặc nhóm bit) đối với các thuật toán hiện đại.

Cryptanalysis: nếu coi mật mã học là việc cất dữ liệu của bạn vào một cái hộp sau đó dùng chìa khóa khóa lại, thì cryptanalysis là ngành nghiên cứu những phương pháp mở hộp để xem dữ liệu khi không có chìa khóa.

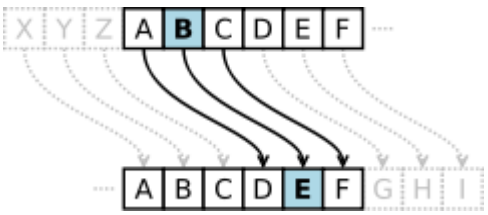
II. MẬT MÃ CAESAR

Mật mã Caesar là mật mã thân thuộc và gần gũi nhất với thế hệ trẻ Việt Nam thời xưa. Các mật thư, kí hiệu trong sổ tay Đội viên đều dựa trên loại mật mã này. Trong đồ án của chúng tôi có biểu diễn lại mật mã này trong sản phẩm.

1. Nguồn gốc:

Trong mật mã học, mật mã Caesar, còn gọi là mật mã dịch chuyển, là một trong những mật mã đơn giản và được biết đến nhiều nhất. Mật mã là một dạng của mật mã thay thế, trong đó mỗi ký tự trong văn bản được thay thế bằng một ký tự cách nó một đoạn trong bảng chữ cái để tạo thành bản mã. Vĩ dụ, nếu độ dịch là 3, A sẽ được thay bằng D, Ă sẽ được thay bằng C và cứ thế đến hết. Phương pháp được đặt tên theo Caesar, vị hoàng đế đã sử dụng nó thường xuyên trong công việc.

Bước mã hóa bằng mã Caesar thường được kết hợp với một mã phức tạp hơn, ví dụ như **mật mã Vigenère**, và hiện nay vẫn được dùng trong các ứng dụng hiện đại như ROT13. Cũng như các mật mã thay thế dùng một bảng mã khác, mã Xê da dễ dàng bị phá vỡ và không đáp ứng được yêu cầu an toàn thông tin trong truyền thông.



Hình 1.1 Mật mã Caesar

2. Cài đặt thuật toán:

Mỗi ngôn ngữ khác nhau sẽ có cách cài đặt khác nhau. Mình sẽ sử dụng Python để cài đặt thuật toán:

Mặc định bạn có thể dụng bảng chữ cái Tiếng Anh nhưng trong phần cài đặt này. Để thực tế hơn. Mình sẽ sử dụng bảng mã Tiếng Việt.

2.1 Bảng mã:

```
key =
'aáàạảãăắằặẳẵâấầậẩẫbcdđeéẹẻẽêếềệểễfghiíìịỉĩjklmnoóòọỏõôốồộổỗơớờ
ợởỡpqrstuúùụủũưứừựửữvwxyýỳỵỷỹAÁÀẠẢÃĂẮẰẶẲẴÂẤẦẬẨẪBCDĐEÉẸẺẼÊẾỀỆỂỄF
GHIÍÌỊỈĨJKLMNOÓÒỌỎÕÔỐỒỘỔỖƠỚỜỢỞỠPQRSTUÚÙỤỦŨƯỨỪỰỬỮVWXYÝỲỴỶỸ012345
6789`~!@#$%^&*()'
```

2.2 Thuật toán mã hóa:

Nếu như tinh ý, bạn dễ dàng nhận ra thuật toán mấu chốt ở đây là :

```
EK(i) = (i+k) mod N
```

Và cũng dễ dàng cài đặt một hàm như sau:

```
def encrypt(self, n, plaintext):
    """Encrypt the string and return the ciphertext"""
    result = ''
    for l in plaintext:
        try:
            i = (self.key.index(l) + n) % len(self.key)
            result += self.key[i]
        except ValueError:
            result += l
    return result
```

2.3 Hàm giải mã:

```
def decrypt(self, n, ciphertext):
    """Decrypt the string and return the plaintext"""
    result = ''

    for l in ciphertext:
        try:
            i = (self.key.index(l) - n) % len(self.key)
            result += self.key[i]
        except ValueError:
            result += l

    return result
```

III. MÃ HÓA ĐỐI XỨNG (MÃ HÓA KHÔNG CÔNG KHAI)

Là lớp thuật toán các mã hóa trong đó việc mã hóa và giải mã đều dùng chung cho 1 khóa (secret key).

1. Các loại thuật toán mã hóa đối xứng:

Thuật toán đối xứng có thể được chia ra làm hai thể loại, mật mã luồng (stream ciphers) và mật mã khối (block ciphers). Mật mã luồng mã hóa từng bit của thông điệp trong khi mật mã khối gộp một số bit lại và mật mã hóa chúng như một đơn vị. Cỡ khối được dùng thường là các khối 64 bit. Thuật toán tiêu chuẩn mã hóa tân tiến (Advanced Encryption Standard), được NIST công nhận tháng 12 năm 2001, sử dụng các khối gồm 128 bit.

Các thuật toán đối xứng thường không được sử dụng độc lập. Trong thiết kế của các hệ thống mật mã hiện đại, cả hai thuật toán bất đối xứng (asymmetric) (dùng chìa khóa công khai) và thuật toán đối xứng được sử dụng phối hợp để tận dụng các ưu điểm của cả hai. Những hệ thống sử dụng cả hai thuật toán bao gồm những cái như SSL (Secure Sockets Layer), PGP (Pretty Good Privacy) và GPG (GNU Privacy Guard) v.v. Các thuật toán chìa khóa bất đối xứng được sử dụng để phân phối chìa khóa mật cho thuật toán đối xứng có tốc độ cao hơn.

Một số ví dụ các thuật toán đối xứng nổi tiếng và khá được tôn trọng bao gồm Twofish, Serpent, AES (còn được gọi là Rijndael), Blowfish, CAST5, RC4, Tam phần DES (Triple DES), và IDEA (International Data Encryption Algorithm – Thuật toán mật mã hóa dữ liệu quốc tế).

Trong đồ án của nhóm mình, chúng mình chọn Fernet làm thuật toán đại diện cho kiểu mã hóa đối xứng này.

2. Tốc độ:

Các thuật toán đối xứng nói chung đòi hỏi công suất tính toán ít hơn các thuật toán khóa bất đối xứng (asymmetric key algorithms). Trên thực tế, một thuật toán khóa bất đối xứng có khối lượng tính toán nhiều hơn gấp hằng trăm, hằng ngàn lần một thuật toán khóa đối xứng (symmetric key algorithm) có chất lượng tương đương.

3. Hạn chế:

Hạn chế của các thuật toán khóa đối xứng bắt nguồn từ yêu cầu về sự phân hưởng chìa khóa bí mật, mỗi bên phải có một bản sao của chìa. Do khả năng các chìa khóa có thể bị phát hiện bởi đối thủ mật mã, chúng thường phải được bảo an trong khi phân phối và trong khi dùng. Hậu quả của yêu cầu về việc lựa chọn, phân phối và lưu trữ các chìa khóa một cách không có lỗi, không bị mất mát là một việc làm khó khăn, khó có thể đạt được một cách đáng tin cậy.

Để đảm bảo giao thông liên lạc an toàn cho tất cả mọi người trong một nhóm gồm n người, tổng số lượng chìa khóa cần phải có là $\frac{n(n-1)}{2}$.

Hiện nay người ta phổ biến dùng các thuật toán bất đối xứng có tốc độ chậm hơn để phân phối chìa khóa đối xứng khi một phiên giao dịch bắt đầu, sau đó các thuật toán khóa đối xứng tiếp quản phần còn lại . Vấn đề về bảo quản sự phân phối chìa khóa một cách đáng tin cậy cũng tồn tại ở tầng đối xứng, song ở một điểm nào đấy, người ta có thể kiểm soát chúng dễ dàng hơn. Tuy thế, các khóa đối xứng hầu như đều được sinh tạo tại chỗ.

Các thuật toán khóa đối xứng không thể dùng cho mục đích xác thực (authentication) hay mục đích chống thoái thác (non-repudiation) được.

IV. MÃ HÓA BẤT ĐỐI XỨNG (MÃ HÓA CÔNG KHAI)

Là thuật toán trong đó việc mã hóa và giãi mã dùng hai khóa khác nhau là pulic key (khóa công khai hay khóa công cộng) và private key (khóa riêng). Nếu dùng public key để mã hóa thì private key sẽ dùng để giải mã và ngược lại.

1. An toàn:

Về khía cạnh an toàn, các thuật toán mật mã hóa bất đối xứng cũng không khác nhiều với các thuật toán mã hóa đối xứng. Có những thuật toán được dùng rộng rãi, có thuật toán chủ yếu trên lý thuyết; có thuật toán vẫn được xem là an toàn, có thuật toán đã bị phá vỡ… Cũng cần lưu ý là những thuật toán được dùng rộng rãi không phải lúc nào cũng đảm bảo an toàn. Một số thuật toán có những chứng minh về độ an toàn với những tiêu chuẩn khác nhau. Nhiều chứng minh gắn việc phá vỡ thuật toán với những bài toán nổi tiếng vẫn được cho là không có lời giải trong thời gian đa thức . Nhìn chung, chưa có thuật toán nào được chứng minh là an toàn tuyệt đối (như hệ thống mật mã sử dụng một lần). Vì vậy, cũng giống như tất cả các thuật toán mật mã nói chung, các thuật toán mã hóa khóa công khai cần phải được sử dụng một cách thận trọng.

2. Ứng dụng:

Ứng dụng rõ ràng nhất của mật mã hóa khóa công khai là bảo mật: một văn bản được mã hóa bằng khóa công khai của một người sử dụng thì chỉ có thể giải mã với khóa bí mậtcủa người đó.

Các thuật toán tạo chữ ký số khóa công khai có thể dùng để nhận thực. Một người sử dụng có thể mã hóa văn bản với khóa bí mật của mình. Nếu một người khác có thể giải mãvới khóa công khai của người gửi thì có thể tin rằng văn bản thực sự xuất phát từ người gắn với khóa công khai đó.

Các đặc điểm trên còn có ích cho nhiều ứng dụng khác như: tiền điện tử, thỏa thuận khóa…

3. Hạn chế:

Tồn tại khả năng một người nào đó có thể tìm ra được khóa bí mật. Không giống với hệ thống mật mã sử dụng một lần (one-time pad) hoặc tương đương, chưa có thuật toán mã hóa khóa bất đối xứng nào được chứng minh là an toàn trước các tấn công dựa trên bản chất toán học

của thuật toán. Khả năng một mối quan hệ nào đó giữa 2 khóa hay điểm yếu của thuật toán dẫn tới cho phép giải mã không cần tới khóa hay chỉ cần khóa mã hóa vẫn chưa được loại trừ. An toàn của các thuật toán này đều dựa trên các ước lượng vềkhối lượng tính toán để giải các bài toán gắn với chúng. Các ước lượng này lại luôn thay đổi tùy thuộc khả năng của máy tính và các phát hiện toán học mới.

Khả năng bị tấn công dạng kẻ tấn công đứng giữa (man in the middle attack): kẻ tấn công lợi dụng việc phân phối khóa công khai để thay đổi khóa công khai. Sau khi đã giả mạo được khóa công khai, kẻ tấn công đứng ở giữa 2 bên để nhận các gói tin, giải mã rồi lại mã hóa với khóa đúng và gửi đến nơi nhận để tránh bị phát hiện. Dạng tấn công kiểu này có thể phòng ngừa bằng các phương pháp trao đổi khóa an toàn nhằm đảm bảo nhận thực người gửi và toàn vẹn thông tin. Một điều cần lưu ý là khi các chính phủ quan tâm đến dạng tấn công này: họ có thể thuyết phục (hay bắt buộc) nhà cung cấp chứng thực số xác nhận một khóa giả mạo và có thể đọc các thông tin mã hóa.

4. Khối lượng tính toán:

Để đạt được độ an toàn tương đương đòi hỏi khối lượng tính toán nhiều hơn đáng kể so với thuật toán mật mã hóa đối xứng. Vì thế trong thực tế hai dạng thuật toán này thường được dùng bổ sung cho nhau để đạt hiệu quả cao. Trong mô hình này, một bên tham gia trao đổi thông tin tạo ra một khóa đối xứng dùng cho phiên giao dịch. Khóa này sẽ được trao đổi an toàn thông qua hệ thống mã hóa khóa bất đối xứng. Sau đó 2 bên trao đổi thông tin bí mật bằng hệ thống mã hóa đối xứng trong suốt phiên giao dịch.

V. ĐỒ ÁN

Nhóm mình đã tạo ra một chương trình nhỏ, cho phép người dùng mã hóa các đoạn text nhập vào và cũng từ đó chỉ ra đặc điểm của các mô hình mã hóa đã trình bày ở trên. Mọi thông tin của đồ án tại https://github.com/t3bol90/Mini_Cryptography_4F.

1. Phân chia :

- Đồ án được chia làm hai giai đoạn: Giai đoạn 1 gồm quá trình tạo menu, lên bố cục cũng như chọn hàm và nghiên cứu lý thuyết của thuật toán. Song song với đó là xử lí phần hash, login và xử lí password (Vì những phần này dễ và đã từng làm ở những môn trước). Giai đoạn 2 gồm các bước đồng nhất các thuật toán lại với nhau thành một thể thống nhất và ghép vào menu, chạy test và fix bug.
- Công việc được phân chia như sau:

Thành viên	Giai đoạn 1	Giai đoạn 2	Giai đoạn 2.5 (Debug, test,…)	% Đánh giá
Sơn	Login, Sign in, Checkpass (hash)	Check lại login bằng hash	Fix bug	
Huy	Hàm đọc txt và trả ra text	Đồng nhất hàm Fernet()	Fix bug	
Lê	Làm menu cơ bản	Đồng nhất RSA	Fix bug	
Toàn	Thiết kế chương trình và viết báo cáo.	Shift và Loop menu.	Ghép code và kiểm tra bug	

2. Quá trình thực hiện:

- Hàm băm được chọn là SHA256 , thuật toán rất đơn giản. Khi người dùng tạo tài khoản mới, ta so sánh với các tài khoản cũ (bằng hash) và sau đó tạo mới cho họ. Vì không có hoạt động quên mật khẩu hay đổi mật khẩu nên cũng không có gì rườm rà.
- Ở hàm mã hóa đầu tiên, nhóm chọn thuật toán Shift cổ điển vì thuật toán này nhanh và dễ hình dung ra được. Và thuật toán này gánh được file 50MB kia trong thời gian có thể chờ đợi được.

- Thuật toán thứ hai chúng mình chọn là Fernet, một thuật toán đối xứng cơ bản của thư viện cryptography. Ưu điểm của thuật toán này là nhanh để chạy được bộ test 50MB như đề bài yêu cầu.
- Thuật toán cuối cùng là RSA, một thuật toán bất đối xứng thông dụng và có nhiều ứng dụng trong đời sống thực tiễn. Nhưng nhận ra ngay được giới hạn của thuật toán này như đã trình bày ở trên nên không thể đo được thời gian chạy của một file 50MB nó sẽ như thế nào vì keysize của RSA là giới hạn theo bộ đệm của môi trường. Ví dụ bộ đệm OAEP là 42 và ta dùng keysize cho RSA là 2048 bits thì số byte mã hóa được sẽ là 214 bytes. Con số này so với 50MB thì keysize phải lớn hơn gấp nhiều lần hiện tại.

3. Kết quả cuối cùng:

- Đồ án hoàn thành trong vòng 2 tuần với các chức năng :
    - Mã hóa chuỗi text người dùng nhập vào.
    - Mã hóa chuỗi text nhập từ file, nếu dưới 255 kí tự có thể in ra màn hình kết quả sau khi decrypt.
    - Mã hóa file 50MB. (Không mã hóa bằng RSA file 50MB được).
- Về kiến thức thu được:
    - Đã nắm được các khái niệm cơ bản của mã hóa thông tin.
    - Hiểu được các thuật ngữ chuyên ngành của ngành.
    - Lập kế hoạch hợp lí và làm việc nhóm tốt hơn đồ án trước!