TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI

KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO BÀI TẬP LỚN

HỌC PHẦN AN TOÀN VÀ BẢO MẬT THÔNG TIN

ĐỀ TÀI

MÃ HOÁ DỮ LIỆU DES VÀ ỨNG DỤNG THUẬT TOÁN CHIA SẺ BÍ MẬT SHAMIR TRONG PHÂN PHỐI KHOÁ DES

GVHD: TS. Lê Thị Anh

Lớp: 20231IT6001001 Khóa: 16

Thành viên nhóm 11:

Phạm Đức Thông 2021605538

Võ Quang Minh 2021606993

Nguyễn Viết Hải 2020603093

Vũ Trọng Tấn 2021607675

Hà Nội – Năm 2023

LỜI CẢM ƠN

Chúng em xin gửi lời cảm ơn chân thành nhất đến quý thầy cô đã hướng dẫn và hỗ trợ chúng em trong quá trình hoàn thành báo cáo bài tập lớn môn học.

Với sự tận tâm và kiến thức của mình, cô đã cung cấp cho chúng em những kiến thức cơ bản về An toàn bảo mật thông tin và giúp chúng em áp dụng những kiến thức đó vào thực tế. Qua quá trình học tập, chúng em đã được trải nghiệm các công cụ và phương pháp lập mã và thám mã hiện đại, từ đó ứng dụng vào báo cáo bài tập lớn của mình.

Bài tập lớn này đã giúp chúng em áp dụng những kiến thức và kỹ năng đã học. Chúng em đã học được nhiều điều về mật mã học. Qua việc đối mặt với những thách thức và khó khăn trong quá trình hoàn thành bài tập lớn này, chúng em đã rèn luyện và phát triển kỹ năng tư duy logic, sáng tạo và giải quyết vấn đề.

Cuối cùng, chúng em xin gửi lời cảm ơn đến tất cả những người thân yêu đã động viên và hỗ trợ chúng em trong suốt quá trình thực hiện bài tập lớn này.

MỤC LỤC

DANH SÁCH HÌNH

DANH MỤC BẢNG

MỞ ĐẦU

1. Lý do chọn đề tài

Khi học học phần An toàn và bảo mật thông tin, chúng em muốn tìm hiểu và giải thích được một số khái niệm cơ bản liên quan đến quá trình mã hóa và giải mã thông tin. Tiếp theo đó là hiểu và phân tích được một số hệ mật mã được giảng dạy trong học phần, ví dụ như Hệ mật mã DES và 3DES. Đưa ra được những ưu nhược điểm của nó, từ đó nâng cao tính bảo mật bằng thuật toán chia sẻ bí mật Shamir trong phân phối khoá DES; sau đó ứng dụng vào dự án phù hợp. Đề tài nghiên cứu dưới đây đã phù hợp với mong muốn của chúng em khi học học phần này.

2. Mục tiêu nghiên cứu

Mục tiêu cần đạt được sau khi hoàn thành bài tập lớn:

- Giải thích và chỉ ra được các khái niệm của bản về mật mã học;

- Giải thích và chỉ ra được các bước hoạt động của hệ mật DES;

- Giải thích và chỉ ra được các bước hoạt động của thuật toán chia sẻ bí mật Shamir;

- Ứng dụng DES, thuật toán chia sẻ bí mật Shamir vào một dự án cụ thể.

3. Đối tượng nghiên cứu

- Hệ mật DES, thuật toán chia sẻ bí mật Shamir và ứng dụng của hệ mật DES, thuật toán chia sẻ bí mật Shamir.

4. Kết quả mong muốn đạt được

Kết quả sau khi hoàn thành bài tập lớn cần đạt được là:

- Giải thích, và chỉ ra được các khái niệm của bản về mật mã học;

- Giải thích, chỉ ra được các bước hoạt động của hệ mật DES;

- Giải thích và chỉ ra được các bước hoạt động của thuật toán chia sẻ bí mật Shamir;

- Ứng dụng DES, thuật toán chia sẻ bí mật Shamir vào một dự án cụ thể.

5. Cấu trúc quyển báo cáo

Ngoài phần Mở đầu và phần Kết luận, nội dung bài tập lớn gồm 3 chương sau:

Nội dung chương 1: Tổng quan về mật mã học;

Nội dung chương 2: Chuẩn mã hóa dữ liệu DES;

Nội dung chương 3: Thuật toán chia sẻ bí mật Shamir;

Nội dung chương 4: Chương trình và kết quả thực hiện;

CHƯƠNG 1:
TỔNG QUAN VỀ MẬT MÃ HỌC

Nội dung chương 1 cung cấp các kiến thức cơ bản của mật mã học, trình bày về phân loại mật mã.

1.1 Giới thiệu về mật mã học

Mật mã đã xuất hiện và được ứng dụng từ hàng nghìn năm trước ở La Mã, Ả Rập,...; tuy nhiên mật mã khi đấy chủ yếu được dùng trong lĩnh vực quân sự. Ngày nay, nhờ sự phát triển của khoa học kĩ thuật, mật mã được ứng dụng sâu hơn, rộng hơn không chỉ trong các lĩnh vực quân sự, quốc phòng, an ninh mà còn trong các lĩnh vực phi quân sự như thương mại điện tử, ngân hàng,...

1.1.1 Khái niệm mật mã, thám mã

Mã hoá *(Cryptography)*: nghiên cứu các thuật toán và phương thức để đảm bảo tính bí mật và xác thực của thông tin. Các sản phẩm cơ bản: các hệ mật mã, các hàm băm, các hệ chữ ký điện tử, các cơ chế phân phối, quản lý khoá và các giao thức mật mã.

Thám mã *(Cryptanalysis-codebreaking):* nghiên cứu các phương pháp phá mã hoặc tạo mã giả. Sản phẩm của lĩnh vực này là các phương pháp thám mã, các phương pháp giả mạo chữ ký, các phương pháp tấn công các hàm băm và các giao thức mật mã.

1.1.2 Mục đích của mật mã

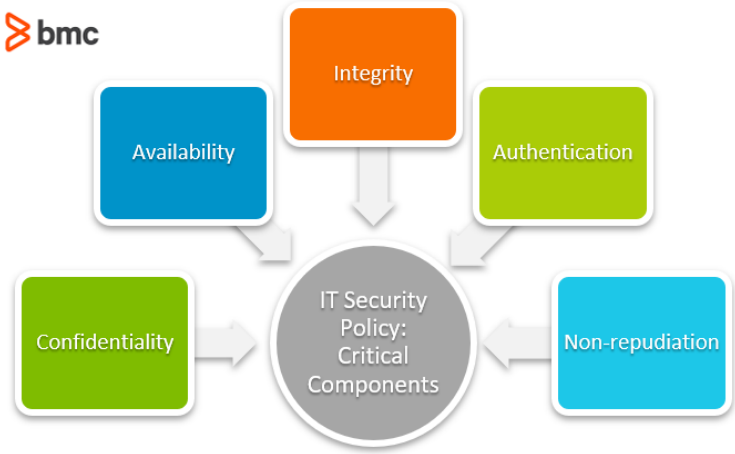
Tính bảo mật *(Confidentiality)*: Chỉ người nhận dự định mới có thể truy cập và đọc thông tin, vì vậy các cuộc hội thoại và dữ liệu vẫn được giữ kín.

Tính toàn vẹn của dữ liệu (Integrity of data): Mật mã đảm bảo rằng dữ liệu được mã hóa không thể bị sửa đổi hoặc giả mạo trên đường truyền từ người gửi đến người nhận mà không để lại dấu vết có thể theo dõi. Ví dụ về điều này là chữ ký số.

Tính sẵn sàng *(Availability)*: Đảm bảo truy cập và sử dụng thông tin kịp thời và đáng tin cậy. Mất tính sẵn sàng là sự gián đoạn truy cập hoặc gián đoạn sử dụng thông tin hoặc gián đoạn sử dụng hệ thống thông tin.

Xác thực *(Authentication)*: Danh tính và đích đến (hoặc nguồn gốc) được xác minh.

Chống chối bỏ *(Non-repudiation)*: Người gửi phải chịu trách nhiệm về tin nhắn của mình vì sau này họ không thể phủ nhận rằng tin nhắn đã được truyền đi. Ví dụ chữ ký số và theo dõi email là những ví dụ về điều này.



Hình 1 Mục đích của mật mã

1.2 Các khái niệm, mô hình truyền tin cơ bản trong mật mã học

1.2.1 Các khái niệm cơ bản

Các khái niệm cơ bản thường dùng trong mật mã:

+ *Thông điệp (Message)*: là một thực thể vật lý mang thông tin cần trao đổi. Ví dụ: lá thư, bức ảnh, video, đoạn ghi âm,…

+ *Bản rõ (Plaintext)***:** là thuật ngữ chỉ thông điệp rõ ràng có thể dễ dàng hiểu được. Bản rõ là đầu vào của thuật toán mã hoá.

+ *Bản mã (Ciphertext):* là thuật ngữ chỉ thông điệp sau khi đã được mã hoá, có thể đọc được nhưng không thể hiểu được. Bản mã là đầu ra của thuật toán mã hoá.

+ *Khóa (Key):* thông tin tham số dùng để mã hóa. Là thành phần quan trọng trong mã hóa và giải mã. Tương tự như khóa vật lý, chỉ khi có khóa mới có thể "mở" (giải mã) bản mã thành bản rõ và ngược lại.

+ *Mã hóa (Encryption):* Là quá trình biến đổi thông tin từ dạng bản rõ sang bản mã bằng khóa hoặc không cần khóa.

+ *Giải mã (Decryption):* Là quá trình ngược lại với mã hoá, nhằm biến đổi thông tin từ dạng bản mã sang bản rõ.

1.2.2 Hệ thống mật mã

Hệ thống mật mã *(Cryptosystem)*: là một bộ năm (P, C, K, E, D) thoả mãn điều kiện:

+ Tập nguồn *P (plaintext)* là tập bản rõ.

+ Tập đích *C (ciphertext)* là tập bản mã.

+ Tập *K (key)* là tập khoá (giữ tuyệt mật).

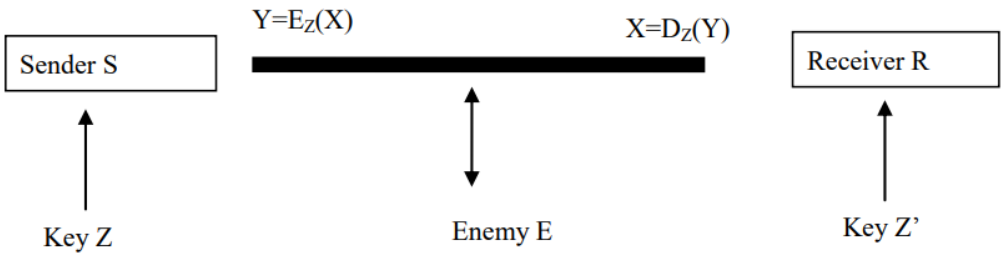
+ *E (encryption)* là quy tắc mã hoá, được gọi là phép lập mật mã (có thể không cần giữ bí mật).

+ *D (decryption)* là quy tắc giải mã, được gọi là phép giải mã (có thể không cần giữ bí mật).

Với mỗi khoá $k \in K$ tồn tại luật mã hoá $e_k \in E: P \rightarrow C$ và luật giải mã $d_k \in D: C \rightarrow P$ sao cho:

$$d_k(e_k(x)) = x, \forall\, x \in P$$

1.2.3 Mô hình truyền tin



Hình 2 Mô hình truyền tin cơ bản trong mật mã

Người phát S (sender) muốn gửi một thông điệp (message) X tới người nhận R (receiver) qua một kênh truyền tin (communication channel). Kẻ thù E (enenmy) muốn lấy/nghe trộm thông tin X. Thông tin X là ở dạng đọc được, còn gọi là bản rõ (plaintext). Để bảo mật, S sử dụng một phép biến đổi mã hoá (encryption), tác động lên X, để chế biến ra một bản mã Y (cryptogram, hay ciphertext), không thể đọc được. Ta nói bản mã Y đã che giấu nội dung của bản rõ X bản đầu. Giải mã (decryption) là quá trình ngược lại cho phép người nhận thu được bản rõ X từ bản mã Y.

Sơ đồ mô hình nói trên cũng thể hiện một điều hết sức cơ bản là toàn bộ tính bảo mật của cơ chế phụ thuộc vào tính mật của khóa, chứ không phải là tính mật của thuật toán mã hoá hay giải mã (encryption và decryption).

Như vậy, khóa giữ vai trò trung tâm trong mô hình truyền tin mật.

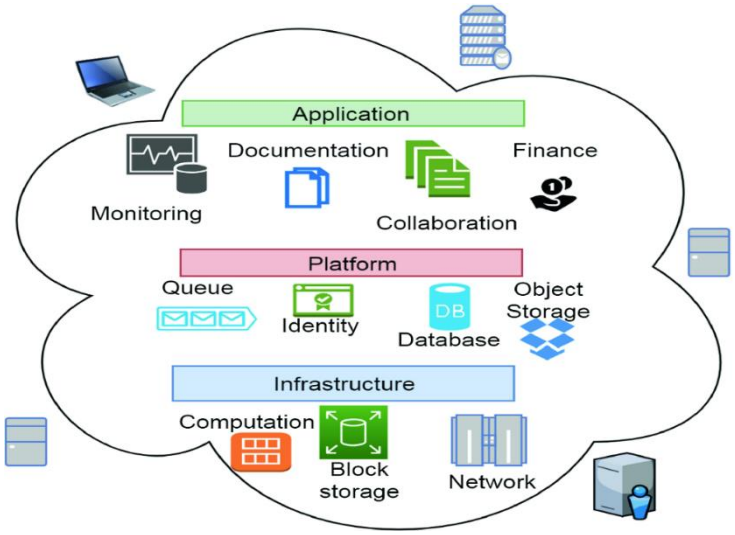
1.3 Ứng dụng của mật mã học

Lưu trữ mật khẩu và dữ liệu: Ngăn chặn dữ liệu bị rò rỉ thiệt hại nếu ổ cứng bị đánh cắp.

Giao tiếp bảo mật: Thường thấy trong giao tiếp giữa người dùng (client) và máy chủ (server).

Thương mại điện tử: Bao gồm thanh toán điện tử, tiền tệ điện tử, v.v.

Chữ ký điện tử (digital signature) và nhiều loại dấu vết điện tử khác (digital footprint).



Hình 3 Ứng dụng của mật mã học

1.4 Phân loại hệ mật mã

Phân loại dựa trên hai yếu tố: thời gian và cách thức truyền khoá.

1.4.1. Dựa theo thời gian

Hệ mã cổ điển: là một dạng của mật mã học đã được sử dụng trong lịch sử phát triển của loài người nhưng ngày nay đã trở nên lạc hậu do các phương thức mã hóa này quá đơn giản và những kẻ tấn công có thể dễ dàng bẻ khóa thông qua nhiều phương thức như tấn công vét cạn (ví dụ như dùng máy tính thử hết mọi trường hợp) hay dựa trên tấn công thống kê (dựa trên tần suất xuất hiện của các chữ cái).

Ví dụ: Caser, Vigenere,…

Hệ mã hiện đại: là những phương pháp mã hóa thông tin được phát triển để đáp ứng các yêu cầu bảo mật cao hơn, đặc biệt là trong bối cảnh công nghệ thông tin ngày nay. Những hệ mã hiện đại thường sử dụng các thuật toán mạnh mẽ và có tính bảo mật cao, khả năng chống lại các kỹ thuật tấn công hiện đại.

Ví dụ: AES, DES, RSA,…

1.4.2. Dựa theo cách thức truyền khóa

1.4.2.1. Mã hóa khóa bí mật

a) Khái niệm

Mã hóa khóa bí mật, còn gọi là mã hóa đối xứng hay mã hóa khóa riêng, là sử dụng một khóa cho cả quá trình mã hóa (được thực hiện bởi người gửi thông tin) và quá trình giải mã (được thực hiện bởi người nhận).

b) Đặc điểm

Sử dụng cùng một khóa để mã hóa và giải mã: Đây là đặc điểm chính của mã hóa đối xứng. Điều này làm cho mã hóa đối xứng trở nên đơn giản và hiệu quả hơn mã hóa bất đối xứng.

An toàn: Mã hóa đối xứng được coi là an toàn nếu khóa bí mật được bảo mật cẩn thận. Nếu khóa bí mật bị lộ, thì bất kỳ ai cũng có thể mã hóa và giải mã thông tin đã được mã hóa.

Tốc độ: Mã hóa đối xứng thường nhanh hơn mã hóa bất đối xứng.

c) Ưu điểm:

Hiệu Suất Cao: Mã hóa và giải mã đơn giản và nhanh chóng, phù hợp cho các ứng dụng yêu cầu hiệu suất cao.

Quản Lý Khóa Dễ Dàng: Quản lý một khóa duy nhất dễ dàng hơn so với quản lý cặp khóa công khai và bí mật.

d) Nhược điểm:

Vấn Đề Truyền Khóa: Cần phải giải quyết vấn đề truyền khóa an toàn giữa người gửi và người nhận.

Khả năng bị tấn công: Mã hóa đối xứng có thể bị tấn công bằng các phương pháp khác nhau, chẳng hạn như tấn công *brute-force*, tấn công *man-in-the-middle*.

1.4.2.2. Mã hóa công khai

a) Khái niệm

Mã hóa công khai (còn được gọi là mật mã bất đối xứng) là một hệ thống sử dụng các cặp khóa để mã hóa và giải mã thông tin. Một khóa trong cặp là khóa công khai, có thể được phân phối rộng rãi mà không ảnh hưởng đến bảo mật. Khóa thứ hai trong cặp là khóa riêng chỉ được chủ sở hữu biết.

b) Đặc điểm

Khóa công khai có thể được chia sẻ công khai mà không ảnh hưởng đến bảo mật. Điều này cho phép hai bên trao đổi thông tin an toàn mà không cần phải trao đổi khóa chung trước đó.

Khóa riêng chỉ được chủ sở hữu biết. Điều này đảm bảo rằng chỉ có chủ sở hữu mới có thể giải mã thông tin đã được mã hóa bằng khóa công khai của họ.

Mã hóa công khai dựa trên các thuật toán toán học phức tạp. Điều này làm cho việc bẻ khóa trở nên rất khó khăn, ngay cả với các máy tính mạnh nhất hiện có.

c) Ưu điểm:

Mức độ bảo mật cao: Mã hóa công khai dựa trên các thuật toán toán học phức tạp, khiến việc bẻ khóa trở nên rất khó khăn, ngay cả với các máy tính mạnh nhất hiện có. Điều này làm cho mã hóa công khai trở thành một giải pháp bảo mật hiệu quả cho các ứng dụng nhạy cảm.

An Toàn Truyền Khóa: cho phép hai bên trao đổi khóa chung bí mật mà không cần phải chia sẻ khóa chung đó trước đó. Điều này có thể được thực hiện bằng cách sử dụng một thuật toán trao đổi khóa công khai, chẳng hạn như Diffie-Hellman.

Tính linh hoạt: Mã hóa công khai có thể được sử dụng trong nhiều ứng dụng khác nhau, bao gồm chữ ký điện tử, trao đổi khóa bảo mật và https. Điều này làm cho mã hóa công khai trở thành một công nghệ quan trọng được sử dụng trong nhiều lĩnh vực khác nhau.

d) Nhược điểm:

Tốc độ chậm: Mã hóa công khai có thể chậm hơn mã hóa đối xứng, một loại mã hóa sử dụng cùng một khóa để mã hóa và giải mã thông tin. Điều này là do mã hóa công khai yêu cầu hai khóa khác nhau, một khóa công khai và một khóa bí mật.

Khả năng bị tấn công: Mặc dù mã hóa công khai được coi là an toàn, nhưng nó vẫn có thể bị tấn công. Một số loại tấn công mã hóa công khai phổ biến bao gồm tấn công *brute-force*, tấn công *man-in-the-middle*.

1.4.2.3. So sánh

Giống nhau:

+ Đều sử dụng các thuật toán toán học để mã hóa và giải mã thông tin.
+ Đều được sử dụng để mã hoá dữ liệu nhằm bảo vệ thông tin khỏi bị truy cập trái phép.

Khác nhau:

Bảng 1 So sánh mã hóa đối xứng và mã hóa bất đối xứng

Sự khác biệt chính	Mã hóa đối xứng	Mã hóa bất đối xứng
Kích thước văn bản mật mã	Văn bản mật mã nhỏ hơn so với tệp văn bản thuần túy ban đầu.	Văn bản mật mã lớn hơn so với tệp văn bản thuần túy gốc.
Kích thước dữ liệu	Được sử dụng để truyền dữ liệu lớn.	Được sử dụng để truyền dữ liệu nhỏ.
Độ dài khóa	Kích thước khóa 128 hoặc 256-bit.	Kích thước khóa RSA 2048-bit hoặc cao hơn.
Số lượng khóa	Mã hóa đối xứng sử dụng một khóa duy nhất để mã hóa và giải mã.	Mã hóa bất đối xứng sử dụng hai khóa để mã hóa và giải mã
Kỹ thuật	Là một kỹ thuật cũ.	Là một kỹ thuật mã hóa hiện đại.
Bảo mật	Sử dụng một khóa duy nhất để mã hóa và giải mã nên có khả năng bị xâm phạm trong quá trình truyền khoá.	Hai khóa được tạo riêng biệt để mã hóa và giải mã giúp loại bỏ khả năng mất an toàn khi chia sẻ khóa.
Tốc độ	Mã hóa đối xứng là kỹ thuật nhanh.	Mã hóa bất đối xứng chậm hơn về tốc độ.
Ứng dụng	Trao đổi dữ liệu an toàn, xác thực, lưu dữ liệu an toàn.	Chữ ký số, trao đổi khóa, bảo mật, https.

CHƯƠNG 2:
CHUẨN MÃ HÓA DỮ LIỆU DES

Nội dung chương 2 trình bày về cách thức hoạt động của DES, cách xây dựng thuật toán mã hóa DES và các ví dụ minh họa.

2.1 Giới thiệu mã hóa DES

DES là chuẩn mã hóa dữ liệu đầu tiên trên thế giới, được NIST công nhận vào năm 1976. Sau khi được công nhận, DES được ứng dụng và sử dụng rộng rãi trên thế giới. Tuy nhiên do nhưng yếu điểm của mình, vào năm 2002 DES đã bị thay thế bằng AES.

Mặc dù DES không còn là thuật toán tiêu chuẩn mã hóa dữ liệu và được sử dụng nữa, tuy nhiên DES vẫn đóng vai trò quan trọng trong mật mã bởi nó là nền tảng cho các thuật toán mã hóa tiếp theo. Hiểu rõ mã hóa DES giúp nắm được những kiến thức cơ bản trong mã hóa, và tạo tiền đề để nắm bắt các phương pháp mã hóa khác dễ dàng hơn.

2.2 Lịch sử hình thành

Là chuẩn mã hóa dữ liệu đầu tiên do NIST công nhận, lịch sử hình thành của DES diễn ra như sau:

Năm 1960, IBM đã thiết lập một dự án nghiên cứu về mật mã máy tính do Horst Feistel đứng đầu.

Năm 1971, kết quả của dự án là thuật toán LUCIFER, được bán cho LIoyd’s of London để sử dụng trong hệ thống phân phối tiền mặt. LUCIFER là một mã hóa khối hoạt động trên các khối 64 bits, sử dụng kích thước khóa là 128 bits. Trước những kết quả khả quan của dự án LUCIFER tạo ra, IBM đã nỗ lực phát triển sản phẩm mã hóa thương mại có thể bán trên được thị trường với lý tưởng có thể được thực hiện trên một con chip duy nhất. Nỗ lực do Walter Tuchman và Carl Meyer đứng đầu, nó không chỉ là nỗ lực của các nhà nghiên cứu của IBM mà còn có tư vấn đến từ NSA. Các nhà nghiên cứu của IBM kết hợp với sự tư vấn đến từ NSA dưới sự lãnh đạo của Walter Tuchman và Carl Meyer, đã nỗ lực tạo ra một phiên bản tinh chỉnh của LUCIFER có khả năng chống thám mã tốt hơn và có kích thước khóa giảm còn 56 bits để phù hợp với chip đơn.

Năm 1974, IBM giới thiệu thuật toán LUCIFER theo lời kêu gọi thiết kế một thuật toán mã hóa có thể đáp ứng được các tiêu chuẩn nghiêm ngặt của Văn phòng tiêu chuẩn Quốc gia (NBS - National Bureau of Standard), hiện nay

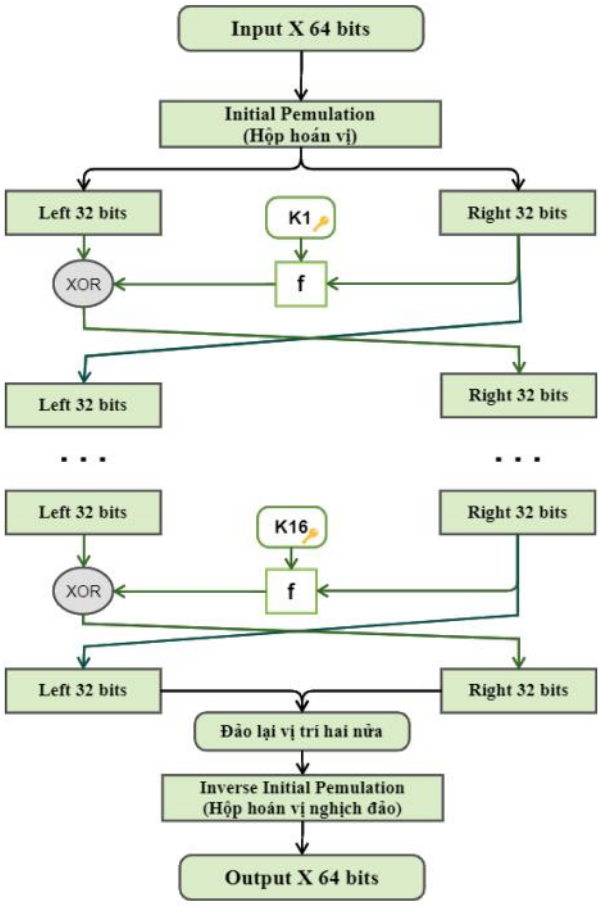
đã đổi tên thành Viện Tiêu chuẩn và Công nghệ Quốc gia Hoa Kỳ (National Institute of Standards and Technology - NIST).

Năm 1976, dự án của IBM được NIST thông qua với tên gọi của mã hóa là Tiêu chuẩn Mã hóa dữ liệu (DES).

2.3 Thuật toán mã hóa DES

Mã hóa DES là mã hóa khóa bí mật (hay mã hóa đối xứng). Mã hóa khóa bí mật phân loại thành: mã hóa khối và mã hóa dòng; DES thuộc mã hóa khối. DES sử dụng khóa 64 bits (trong đó có 8 bits để kiểm tra chẵn lẻ) xử lí thông tin theo từng khối 64 bits. Thuật toán mã hóa DES gọi là DEA (Data Encryption Algorithm - thường được sử dụng giữa DES và DEA). Đầu vào của thuật toán là các khối thông tin 64 bits và đầu ra là khối thông tin 64 bits đã được mã hóa.

Sơ đồ thuật toán mã hóa DES.



Hình 4: Thuật toán mã hóa DES

Theo sơ đồ, ta thấy quá trình mã hóa DES diễn ra qua ba giai đoạn:

Giai đoạn 1:

Khối thông tin bản rõ 64 bit x thực hiện phép hoán vị IP thành khối 64 bit x_0 mới. Ta có:

$$x_0 = IP(x)$$

Bảng 2 IP

Bảng IP							
58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

Phép hoán vị IP : Theo bảng IP, ta có bit thứ nhất của chuỗi x_0 tương ứng là bit thứ 58 của x, bit thứ hai của chuỗi x_0 tương ứng là bit thứ 50 của x,... tương tự đến bit thứ 64 của x_0 là bit thứ 7 của x.

Sau đó khối thông tin được chia làm hai phần bao gồm nửa trái L_0 32 bit và nửa phải R_0 32 bit: $x_0 = L_0R_0$.

Giai đoạn 2:

Mã hóa DES thực hiện 16 vòng lặp với bộ 16 khóa 48 bit K_i , sinh ra từ khóa k cho trước thông qua hàm tạo khóa. Ở mỗi vòng lặp với $i = \overline{1,16}$ thực hiện:

$$L_i = R_{i-1}$$

$$R_i = L_i \oplus F(R_{i-1}, K_i)$$

Với:

- K_i là khóa thứ i sinh ra từ khóa k cho trước.

- F là hàm Feistel biến đổi R_{i-1} thành một chuỗi 32 bit mới (chi tiết cách thức hoạt động của hàm F được trình bày ở phần tiếp theo).

- $\oplus$ là phép XOR trên hai dãy bit.

Giai đoạn 3:

Tiến hành đảo vị trí của nửa trái và nửa phải thu được sau vòng lặp thứ 16 ở giai đoạn hai. Sau đó thực hiện phép hoán vị nghịch đảo IP^{-1} ta thu được khối thông tin 64 bit được mã hóa y: $y = IP^{-1}(R_{16}L_{16})$.

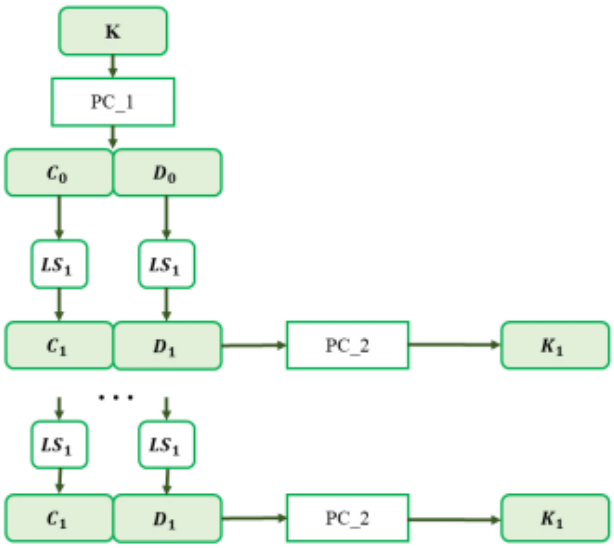
Bảng 3 IP $^{-1}$

Bảng IP^{-1}							
40	8	48	16	56	24	64	32
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28
35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26
33	1	41	9	49	17	57	25

Phép hoán vị IP^{-1} : Cách đọc bảng tương tự hoán vị IP.

2.4 Hàm sinh khóa con

Hàm sinh khóa tạo ra 16 khóa con 48 bit từ khóa k cho trước. Đầu vào của thuật toán là khóa K 64 bit; đầu ra của thuật toán là 16 khóa con 48 bit. Quá trình tạo khóa con được mô tả dưới hình sau:



Hình 5 Sơ đồ hàm sinh khóa

Theo sơ đồ, các bước thực hiện hàm sinh khóa con là:

Bước 1: các bit của khóa đầu vào K, được đánh số từ 1 đến 64, sau thực hiện khối PC_1 bỏ đi các bit thứ 8, 16, 24, 32, 40, 48, 56, 64 ta được chuỗi 56 bit.

Bảng 4 PC_1

PC_1						
57	49	41	33	25	17	9
1	58	50	42	34	26	18
10	2	59	51	43	35	27
19	11	3	60	52	44	36
63	55	47	39	31	23	15
7	62	54	46	38	30	22
14	6	61	53	45	37	29
21	13	5	28	20	12	4

Bước 2: Chuỗi 56 bit thu được, được chia làm hai chuỗi 28 bit C_0 và D_0 sao cho PC_1(K) = C_0D_0. Thực hiện 16 vòng lặp. Với $i = \overline{1,16}$, tại mỗi vòng lặp thực hiện:

$$C_i = LS_1(C_{i-1})$$

$$D_i = LS_1(D_{i-1})$$

Với: LS_1 là phép dịch trái 1 bit với $i = 1, 2, 9, 16$ và dịch trái 2 bit với các vòng còn lại.

Sau đó, thực hiện khối PC_2 với C_iD_i thu được khóa con 48 bit K_i.

Bảng 5 PC_2

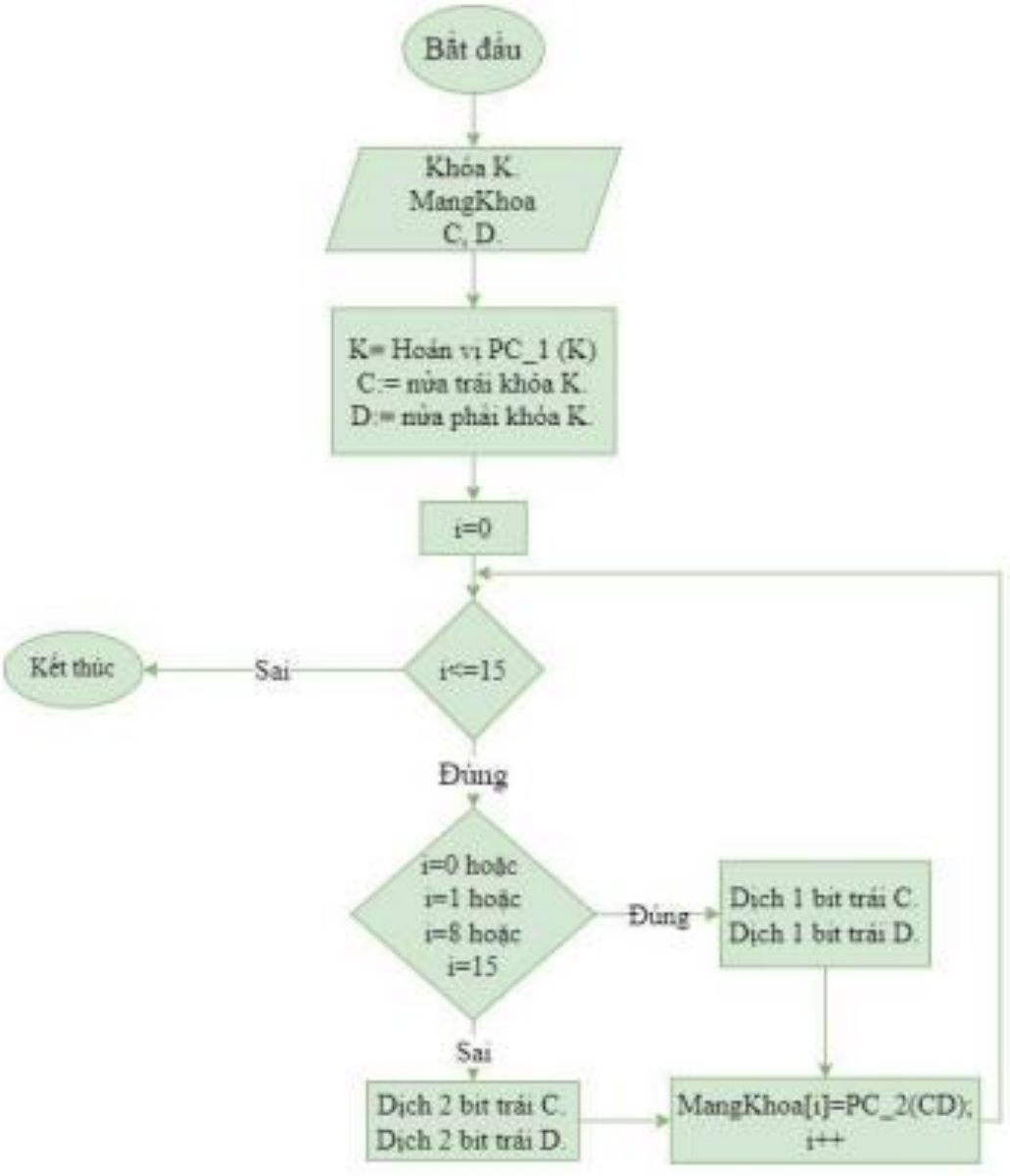
PC_2					
14	17	11	24	1	5
3	28	15	6	21	10
23	19	12	4	26	8
16	7	27	20	13	2
41	52	31	37	47	55
30	40	51	45	33	48
44	49	39	56	34	53
46	42	50	36	29	32

*** Xây dựng hàm sinh khóa**

Sơ đồ khối hàm sinh khóa K:

• Input: Khóa K (đầu vào được biểu diễn ở hệ Hexa có 16 kí tự) được chuyển đổi sang nhị phân có độ dài 64 bit.

• Output: 16 khóa con K_i , i = 1, 16 có độ dài 48 bit được lưu vào mảng MangKhoa.

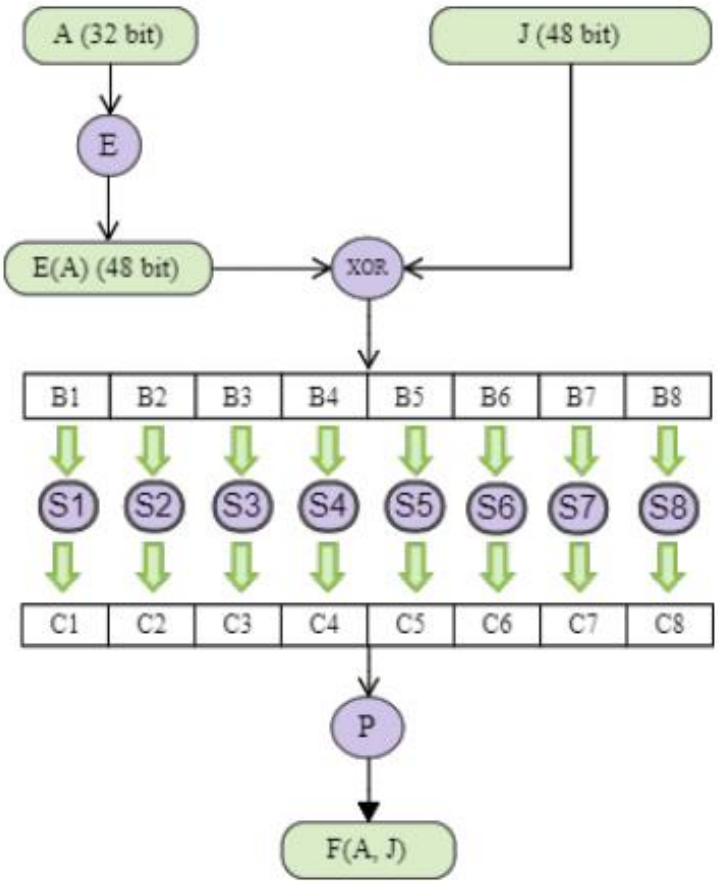


Hình 6 Sơ đồ khối hàm sinh khóa con

2.5 Hàm Feitel (F)

Hàm Feistel được coi là trái tim của thuật toán mã hóa DES. Ở giai đoạn 2 của thuật toán mã hóa, hàm F được sử dụng để biến đổi chuỗi nửa phải (R) sang một chuỗi mới có độ dài 32 bit bằng cách sử dụng khóa con 48 bits sinh bởi khóa K. Đầu vào của hàm F gồm: chuỗi 32 bits và chuỗi khóa con gồm 48 bits. Kết quả đầu ra của hàm F là chuỗi 32 bits.

Sơ đồ hoạt động của hàm F được biểu diễn qua hình dưới đây, mô tả cách thức hoạt động của hàm F với A là chuỗi 32 bit đầu vào; J là khóa con 48 bits sinh bởi khóa K.



Hình 7 Sơ đồ hàm Feitel

Các bước thực hiện hàm F:

Bước 1: Chuỗi 32 bits A đầu vào, đi qua phép mở rộng E thành chuỗi E(A) có 48 bits.

Bảng 6 Mở rộng E

Bảng mở rộng E					
32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	33

Sau đó, thực hiện XOR hai chuỗi E(A) và J (khóa con) thành chuỗi 48 bit B.

Bước 2: Chia chuỗi B thành 8 chuỗi con 6 bit: B_1, B_2, B_3, B_4, B_5, B_6, B_7, B_8, với $B = B_1B_2B_3B_4B_5B_6B_7B_8$. Sử dụng 8 hộp S-box ($S_i$, $i = \overline{1,8}$) biến đổi 8 chuỗi con 6 bit B_i , $i = \overline{1,8}$ thành 8 chuỗi 4 bit C_i :

$$C_i = S_i(B_i)$$

theo quy tắc với mỗi i, $i = \overline{1,8}$:

Xác định giá trị trên hàng r, cột c của ma trận S_i với r là giá trị thập phân của số nhị phân gồm bit đầu và bit cuối của B_i ; c là giá trị thập phân của số nhị phân gồm 4 bit giữa còn lại trong B_i.

C_i là chuyển đổi nhị phân của giá trị trên.

Danh sách 8 hộp S-box:

Bảng 7 Hộp S_1

Hộp S_1															
14	4	13	1	2	15	11	8	3	10	3	12	5	9	1	7
1	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

Bảng 8 Hộp S_2

Hộp S_2															
15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
3	13	4	13	15	2	8	14	12	0	1	10	6	9	11	5
0	14	7	14	10	4	13	1	5	8	12	6	9	3	2	15
13	8	10	8	3	15	4	2	11	6	7	12	0	5	14	9

Bảng 9 Hộp S_3

Hộp S_3															
10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
13	6	4	9	8	5	3	0	11	1	2	12	5	10	14	7
1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12

Bảng 10 Hộp S_4

Hộp S_4															
7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14

Bảng 12 Hộp S_5

Hộp S_5															
2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3

Bảng 11 Hộp S_6

Hộp S_6															
12	1	10	15	9	2	6	8	0	13	3	4	14	7	15	11
10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
4	3	2	12	9	5	15	10	11	14	11	7	6	0	8	13

Bảng 13 Hộp S_7

Hộp S_7															
4	11	12	14	15	0	8	13	3	12	9	7	5	10	6	1
13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12

Bảng 14 Hộp S_8

Hộp S_8															
13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
1	5	13	8	10	3	7	4	12	5	6	11	0	14	9	2
7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

Bước 3: Chuỗi 32 bit C thu được là: C = $C_1C_2C_3C_4C_5C_6C_7C_8$. Thực hiện khối P với C thu được đầu ra chuỗi 32 bit của hàm F:

$$F(A, J) = P(C)$$

Bảng 15 Bảng hoán vị P

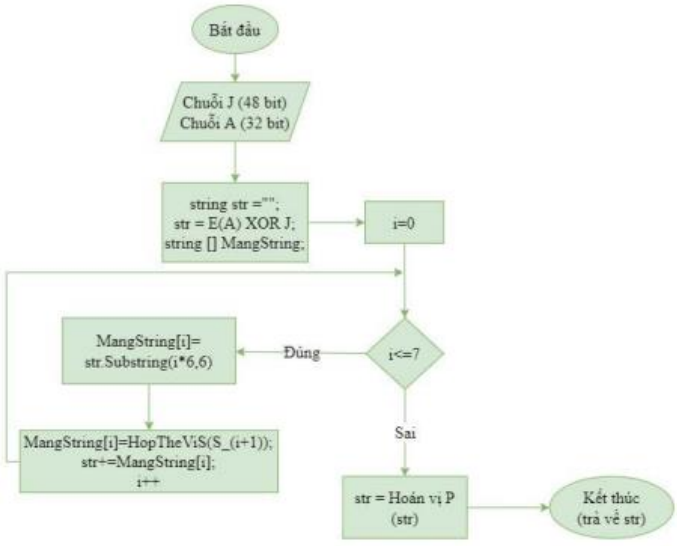
Bảng P			
16	7	20	21
29	12	28	17
1	15	23	26
5	18	31	10
2	8	24	14
32	27	3	9
19	13	30	6
22	11	4	25

***Xây dựng hàm Feistel(F)**

• Input: Chuỗi đầu vào A có 32 bit, khóa con J đầu vào 48 bit.

• Output: Chuỗi 32 bit F(A, J).

Sơ đồ khối của hàm F là:



***Hàm mở rộng E**

Hình 8 Sơ đồ khối hàm Feitel

• Input: Chuỗi A 32 bit, mảng chứa hộp E.

• Output: Chuỗi đầu ra 48 bit

***Hàm thế vị S-box**

• Input: Chuỗi 6 bit, mảng chứa hộp thế vị S.

• Output: Chuỗi 4 bit.

***Hàm hoán vị P**

• Input: Chuỗi 32 bits.

• Output: Chuỗi 32 bits.

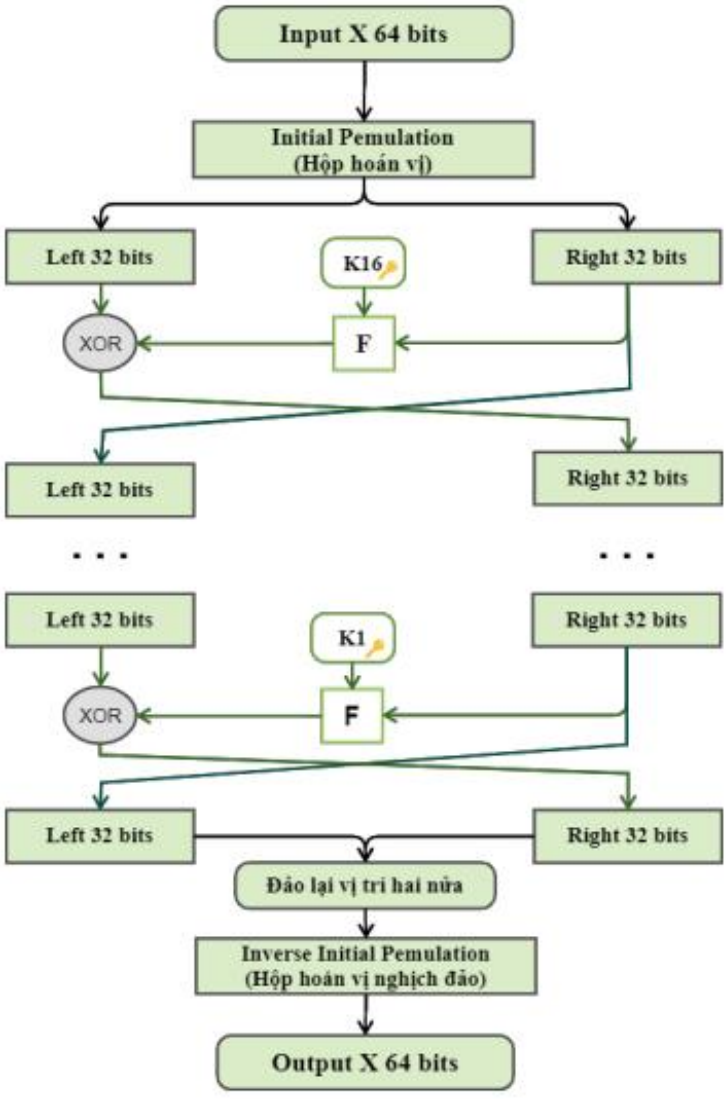
2.6 Thuật toán giải mã DES

Thuật toán giải mã DES là thuật toán sử dụng cùng khóa K với thuật toán mã hóa, biến bản mã hóa y thành bản rõ x.

Thuật toán giải mã DES tương tự như thuật toán mã hóa DES, tuy nhiên bộ khóa giải mã DES sử dụng bộ khóa ngược lại. Đầu vào của thuật toán là chuỗi 64 bit bản mã y, khóa K; kết quả đầu ra là bản rõ 64 bit x. Cách thức hoạt động của thuật toán giải mã DES được minh họa bằng sơ đồ sau, với:

F là hàm Feistel.

Và 16 khóa con K_i được sinh ra từ khóa K đầu vào nhờ hàm sinh khóa.



Hình 9 Sơ đồ thuật toán giải mã DES

2.6 Các vấn đề của mã hóa DES

2.6.1 Tính bảo mật của DES.

Về tính an toàn của DES, có hai vấn đề nổi bật là : Khóa và cách thiết kế hộp S-box.

Thứ nhất về khóa: khóa được sử dụng trong DES là 64 bit (trong đó có 8 bit kiểm tra chẵn lẻ). Hệ thống mật mã LUCIFER của IBM, tiền thân của DES trước đõ sử dụng khóa 128 bit, sau đó DES được đề xuất với 64 bit khóa tuy nhiên IBM đã giảm xuống 56 bit bởi 8 bit trong 64 bit dùng để kiểm tra tính chẵn lẻ. Do khả năng của máy tính điện tử ngày càng nâng cao, với khóa 56 bit, độ an toàn của DES đã bị tấn công bạo lực (Bruce force attack) đe dọa. Điều đó dẫn đến sự ra đời của các hệ thống phá mã DES như: năm 1998, hệ thống phá mã DES của Hiệp hội EFF (Electronic Frontier Foundation) có thể phá khóa DES trong vòng vài ngày. Và theo định luật Moore, các hệ thống phá mã DES giá rẻ được ra mắt.

Thứ hai về S-box: 8 hộp S-box sử dụng trong mã hóa DES là trái tim của mã hóa bởi thông qua tám bảng thay thế này, tính tuyến tính của quá trình mã hóa bị phá bỏ, đảm bảo tính chất xáo trộn và khuếch tán (confusion and diffusion), khiến việc thám mã trở nên khó khăn. Tuy nhiên tiêu chí thiết kế, cách thức thiết kế của S-box lại không được công khai. Nó được nghi ngờ tồn tại các điểm yếu có thể khai thác, và NSA bị nghi ngờ sử dụng những yếu điểm này để phá mã, khai thác thông tin. Tuy nhiên hiện nay, chưa có ai thành công trong việc phát hiện những điểm yếu này của S-box.

2.6.2 Khóa yếu

K được gọi là khóa yếu nếu:

$$E_k(E_k(x)) = x;\ \forall x \in P$$

có nghĩa là mã hóa và giải mã sẽ cho cùng kết quả khi dùng khóa yếu.

Trong DES, có 4 khóa yếu sau:

0101 0101 0101 0101

FEFE FEFE FEFE FEFE

1F1F 1F1F 0E0E 0E0E

E0E0 E0E0 F1F1 F1F1

Bên cạnh đó, trong DES còn có các cặp khóa nửa yếu *(semi-weak keys)*. Hai khóa *K1, K2* là nửa yếu nếu:

$E_{K1}(E_{K2}(x)) = x, \forall x \in P.$

Trong DES, có 6 cặp khóa nửa yếu sau:

Cặp 1: K1: 01FE 01FE 01FE O1FE K2: FE01 FE01 FE01 FE01

Cặp 2: K1: 1FE0 0EF1 0EF1 0EF1 K2: E01F E01F F10E F10E

Cặp 3: K1: 01E0 01E0 01F1 01F1 K2: E001 E001 F101 F101

Cặp 4: K1: 1FFE 1FFE 0EFE 0EFE K2: FE1F FE1F FE0E FE0E

Cặp 5: K1: 011F 011F 010E 010E K2: 1F01 1E01 0E01 0E01

Cặp 6: K1: E0FE E0FE F1FE F1FE K2: FEE0 FEE0 FEF1 FEF1

CHƯƠNG 3:
THUẬT TOÁN CHIA SẺ BÍ MẬT SHAMIR

Nội dung chương 3 sẽ giải thích khái niệm; các bước thực hiện chia bí mật và khôi phục bí mật gắn với khoá DES; trình bày ưu điểm, nhược điểm, ứng dụng của thuật toán chia sẻ bí mật Shamir.

3.1 Khái niệm

Thuật toán chia sẻ bí mật Shamir *(Shamir's Secret Sharing) (SSS)* là một thuật toán trong mật mã được tạo bởi Adi Shamir. Mục đích chính của thuật toán này là chia bí mật cần được mã hóa thành nhiều phần riêng biệt khác nhau.

SSS được sử dụng để bảo mật bí mật ở dạng phân tán, thường là để bảo mật khóa mã hóa. Bí mật được chia thành nhiều phần chia sẻ, riêng lẻ chúng không cung cấp bất kỳ thông tin nào về bí mật.

Để xây dựng lại bí mật được bảo mật bởi SSS, cần có một số lượng chia sẻ, được gọi là "ngưỡng". Không thể lấy được thông tin nào về bí mật từ bất kỳ số lượng chia sẻ nào dưới ngưỡng (thuộc tính được gọi là bí mật hoàn hảo).

Ví dụ: Một công ty cần bảo mật kho tiền của mình. Nếu một một người biết mã của kho tiền thì mã đó có thể bị mất hoặc không khả dụng khi cần mở kho tiền. Nếu có một số người biết mật mã, họ có thể không tin tưởng lẫn nhau để luôn hành động trung thực.

SSS có thể được sử dụng trong trường hợp này để tạo ra các phần mã kho tiền được phân phối cho các cá nhân được ủy quyền trong công ty. Ngưỡng tối thiểu và số lượng cổ phần được trao cho mỗi cá nhân có thể được chọn sao cho chỉ (nhóm) cá nhân được ủy quyền mới có thể truy cập vào kho tiền. Nếu có ít lượt chia sẻ hơn ngưỡng thì kho tiền sẽ không thể mở được.

Do vô tình hoặc do hành động phản đối, một số cá nhân có thể đưa ra thông tin không chính xác về cổ phiếu của họ. Nếu tổng số cổ phiếu hợp lệ không đáp ứng được ngưỡng tối thiểu thì kho tiền vẫn bị khóa.

3.2 Ý tưởng và thuật toán

3.2.1 Ý tưởng

Ý tưởng chính đằng sau Thuật toán chia sẻ bí mật của Shamir là sử dụng Đa thức nội suy Lagrange

Đa thức nội suy Lagrange đề cập đến với K điểm cho trước, chúng ta có thể tìm ra phương trình đa thức có bậc (K – 1).

Ví dụ:

- Với hai điểm cho trước *(x1, y1)* và *(x2, y2)* ta tìm được đa thức tuyến tính $ax + by = c$.
- Tương tự, với ba điểm cho trước, ta tìm được đa thức bậc hai $ax^2 + bx + cy = d$.

Vì vậy, ý tưởng là xây dựng một đa thức có bậc (K – 1) sao cho số hạng hằng số là mã bí mật và các số còn lại là ngẫu nhiên và số hạng không đổi này có thể được tìm thấy bằng cách sử dụng bất kỳ K điểm nào trong số N điểm được tạo từ đa thức này bằng cách sử dụng Đa thức cơ sở Lagrange.

Ví dụ: Đặt mã bí mật S = 65, N = 4, K = 2.

1. Ban đầu, để thực hiện mã hóa mã bí mật, chúng ta sẽ xây dựng đa thức bậc (K – 1).
2. Do đó, gọi đa thức là $y = a + bx$. Ở đây, phần hằng số 'a' là mã bí mật.
3. Cho b là số ngẫu nhiên bất kỳ, giả sử b = 15.
4. Do đó, đối với đa thức $y = 65 + 15x$ này, tạo ra N = 4 điểm từ nó.
5. Gọi 4 điểm đó là *(1, 80), (2, 95), (3, 110), (4, 125)*. Rõ ràng, chúng ta có thể tạo đa thức ban đầu từ hai điểm bất kỳ trong số 4 điểm này và trong đa thức thu được, số hạng không đổi a là mã bí mật bắt buộc.

Để xây dựng lại đa thức đã cho, đa thức cơ sở Lagrange được sử dụng.

Khái niệm chính đằng sau đa thức Lagrange là trước tiên hình thành các đẳng thức Lagrange và tổng của các đẳng thức này cho chúng ta hàm số cần tìm mà chúng ta cần tìm từ các điểm đã cho. Các phương trình sau đây cho biết cách tính:

$$l_i = \frac{x - x_0}{x_i - x_0} \times \ldots \times \frac{x - x_{i-1}}{x_i - x_{i-1}} \times \frac{x - x_{i+1}}{x_i - x_{i+1}} \times \ldots \times \frac{x - x_{k-1}}{x_i - x_{k-1}}$$

$$f(x) = \sum_{i=0}^{K-1} y_i l_i(x)$$

Ví dụ: Giả sử tính phương trình đã cho bằng cách sử dụng K = 2 điểm từ bốn điểm đã chọn *(1, 80), (3, 110)*. Sau đây là minh hoạ các bước giải:

$$l_0 = \frac{x - x_1}{x_0 - x_1} = \frac{x - 3}{1 - 3}$$

$$l_1 = \frac{x - x_0}{x_1 - x_0} = \frac{x - 1}{3 - 1}$$

$$f(x) = y_0 l_0 + y_1 l_1$$

$$f(x) = 80\left(\frac{x - 3}{-2}\right) + 110\left(\frac{x - 1}{2}\right)$$

$$f(x) = -40x + 120 + 55x - 55$$

$$f(x) = 15x + 65$$

3.2.2 Các bước thực hiện thuật toán

Phần này sẽ trình bày các bước thực hiện thuật toán chia sẻ Shamir để chia sẻ khoá DES.

Các bước thực hiện thuật toán để chia sẻ:

Bước 1: Chuyển khoá DES từ hệ Hexa sang hệ cơ số 10. Khoá DES được biểu diễn ở hệ Hexa với 16 kí tự (có cả kí tự chữ để biểu diễn giá trị từ 10-15). Vì thuật toán chia sẻ Shamir thực hiện tính toán dựa trên số học nên chúng ta sẽ chuyển khoá DES từ hệ Hexa sang hệ cơ số 10 để thực hiện tính toán (việc chuyển đổi không tính bit dấu).

Bước 2: Tạo đa thức chia sẻ: Tạo đa thức bậc k-1 với các hệ số (a_0, a_1,…, a_{k-1}) trong đó có hệ số tự do a_0 là phần "bí mật", k-1 hệ số còn lại từ a_1 đến a_{k-1} được khởi tạo ngẫu nhiên.

Bước 3: Sinh các giá trị chia sẻ ngẫu nhiên: Các giá trị chia sẻ ngẫu nhiên có cấu trúc là cặp *(x, y)*. Trong đó x được khởi tạo ngẫu nhiên (x khác 0 và không trùng nhau), y là giá trị được tính từ hàm chia sẻ với giá trị và là x tương ứng.

Các bước thực hiện thuật toán để khôi phục:

Bước 1: Nhận các giá trị đầu vào: Lấy các giá trị đầu vào bao gồm danh sách các điểm và một ngưỡng k.

Bước 2: Thực hiện tính hệ số tự do: Hệ số tự do là bản mà chúng ta cần khôi phục. Hệ số tự do được tính theo công thức.

$$\sum_{i=0}^{k-1} y_i \times \frac{\prod_{j=0}^{k-1}(-x_i)}{\prod_{j=0}^{k-1}(x_i - x_j)}$$

3.3 Đặc điểm

3.3.1 Ưu điểm

Bảo mật: Đặc điểm bảo mật của thuật toán Shamir là do tính chất của đa thức bậc k - 1. Nếu ít hơn k - 1 điểm được cung cấp, thì có thể có nhiều đa thức bậc k - 1 khác nhau đi qua các điểm đó. Điều này có nghĩa là không thể xác định đa thức duy nhất đại diện cho bí mật.

Ví dụ: giả sử có ba người, A, B, và C, muốn chia sẻ một bí mật. Họ có thể sử dụng các điểm sau để đại diện cho bí mật: *(0, 123) ; (1, 135) ; (2, 147)*

Nếu chỉ có hai điểm, chẳng hạn như *(0, 123)* và *(1, 135)*, thì có thể có hai đa thức bậc hai khác nhau đi qua các điểm đó:

Đa thức 1: $y = x^2 + 12x + 147$ và đa thức 2: $y = x^2 + 13x + 159$

Tuy nhiên, nếu có ba điểm, thì chỉ có một đa thức bậc hai duy nhất có thể đi qua các điểm đó:

Đa thức: $y = x^2 + 12x + 123$

Do đó, nếu A và B có hai điểm, thì họ không thể khôi phục bí mật. Họ cần có sự tham gia của C để có đủ ba điểm.

Có khả năng mở rộng: Đối với bất kỳ ngưỡng nhất định nào, các lượt chia sẻ có thể được thêm hoặc xóa động mà không ảnh hưởng đến các lượt chia sẻ hiện có

Tính động: Bảo mật có thể được tăng cường mà không cần thay đổi bí mật, nhưng bằng cách thay đổi đa thức (giữ nguyên thuật ngữ miễn phí) và tạo phần chia sẻ mới cho mỗi người tham gia.

Tính linh hoạt: Trong các tổ chức có hệ thống phân cấp quan trọng, mỗi người tham gia có thể được phát hành số lượng cổ phiếu khác nhau tùy theo tầm quan trọng của họ trong tổ chức.

Ví dụ: Với ngưỡng là 3, chủ tịch có thể mở két một mình nếu được chia ba cổ phần, trong khi ba thư ký với một cổ phần, mỗi người phải gộp số cổ phần của họ lại để mở khóa két.

3.3.2 Nhược điểm

Không chia sẻ bí mật có thể xác minh: Trong quá trình tập hợp lại chia sẻ, SSS không cung cấp cách xác minh tính chính xác của từng chia sẻ đang được sử dụng. Việc chia sẻ bí mật có thể xác minh nhằm mục đích xác minh rằng các cổ đông trung thực và không gửi cổ phiếu giả mạo.

Điểm lỗi tập hợp duy nhất: Bí mật phải tồn tại ở một nơi khi nó được chia thành các phần chia sẻ và lại ở một nơi khi nó được tập hợp lại. Đây là những điểm tấn công và các kế hoạch khác bao gồm đa chữ ký loại bỏ ít nhất một trong số điểm lỗi duy nhất. Tức là bản mật có thể bị tấn công ở nơi nó hình thành khoá con và ở nơi tập hợp khoá con để khôi phục lại bản mật.

Truyền khoá trên kênh truyền: các khoá con sau khi được tạo vẫn sẽ được chia sẻ tới mọi người thông qua kênh truyền. Trong trường hợp xấu nhất, kẻ đánh cắp lấy được k khoá con trong n khoá con thì bí mật không còn được an toàn.

3.4 Ứng dụng

Thuật toán Shamir có thể được ứng dụng trong nhiều lĩnh vực, chẳng hạn như:

Chia sẻ khóa mật mã: Thuật toán Shamir có thể được sử dụng để chia sẻ một khóa mật mã giữa nhiều người. Trong trường hợp này, mỗi người sẽ giữ một phần của khóa. Để sử dụng khóa, cần có sự tham gia của ít nhất k người. Điều này làm cho việc tấn công khóa trở nên khó khăn hơn.

Ví dụ: một tổ chức có thể sử dụng thuật toán Shamir để chia sẻ khóa DES của mình giữa các thành viên trong hội đồng quản trị. Trong trường hợp này, nếu một thành viên trong hội đồng quản trị bị tấn công, thì kẻ tấn công vẫn cần có sự tham gia của ít nhất hai thành viên khác để có thể sử dụng khóa.

Bảo vệ dữ liệu nhạy cảm: Thuật toán Shamir có thể được sử dụng để bảo vệ dữ liệu nhạy cảm, chẳng hạn như thông tin tài chính hoặc thông tin cá nhân. Trong trường hợp này, dữ liệu sẽ được chia thành nhiều phần và mỗi phần sẽ được lưu trữ ở một vị trí khác nhau. Để truy cập dữ liệu, cần có sự tham gia của ít nhất k người.

Ví dụ: một công ty có thể sử dụng thuật toán Shamir để chia sẻ bản sao dữ liệu khách hàng của mình giữa các máy chủ khác nhau. Trong trường hợp này, nếu một máy chủ bị tấn công, thì kẻ tấn công vẫn cần có sự tham gia của ít nhất k máy chủ khác để có thể truy cập dữ liệu.

Kiểm soát truy cập: Thuật toán Shamir có thể được sử dụng để kiểm soát truy cập vào các tài nguyên, chẳng hạn như các hệ thống máy tính hoặc các phòng họp. Trong trường hợp này, mỗi người sẽ được cấp một phần của khóa truy cập. Để truy cập tài nguyên, cần có sự tham gia của ít nhất k người.

Ví dụ: một công ty có thể sử dụng thuật toán Shamir để cấp quyền truy cập vào phòng họp của mình. Trong trường hợp này, mỗi nhân viên sẽ được cấp một

phần của khóa truy cập. Để mở cửa phòng họp, cần có sự tham gia của ít nhất k nhân viên.

Bầu cử điện tử: Thuật toán Shamir có thể được sử dụng để bảo vệ tính an toàn của bầu cử điện tử. Trong trường hợp này, phiếu bầu sẽ được chia thành nhiều phần và mỗi phần sẽ được lưu trữ ở một vị trí khác nhau. Để tính kết quả bầu cử, cần có sự tham gia của ít nhất k người.

Ví dụ: một quốc gia có thể sử dụng thuật toán Shamir để tổ chức bầu cử tổng thống. Trong trường hợp này, mỗi lá phiếu sẽ được chia thành k phần và mỗi phần sẽ được gửi đến một khu vực bầu cử khác nhau. Để tính kết quả bầu cử, cần có sự tham gia của ít nhất k khu vực bầu cử.

Ngoài ra, thuật toán Shamir còn có thể được ứng dụng trong các lĩnh vực khác, chẳng hạn như: Quản lý dữ liệu, Tín dụng, Bảo hiểm, Y tế, Khoa học, Giáo dục.

CHƯƠNG 4:
CHƯƠNG TRÌNH VÀ KẾT QUẢ THỰC HIỆN

Nội dung chương 4 tập trung vào giới thiệu giao diện của chương trình và demo các chức năng và kết quả của chương trình.

4.1 Giao diện chương trình

Giao diện phân tách thành 2 phần tương ứng với 2 chức năng của thuật toán mã hoá DES và thuật toán chia sẻ bí mật Shamir.

Hình 10 Giao diện của chương trình

4.2 Mô tả chức năng

4.2.1 Chức năng mã hoá và giải mã DES

Hình 11 Giao diện mã hóa DES

Chức năng mã hoá DES: cho phép người dùng mã hoá văn bản. Yêu cầu người dùng nhập vào bản rõ (ô Input) và key (ô Key). Sản phẩm sẽ thực hiện mã hoá bản rõ và xuất kết quả là bản mã (ô Output).

Chức năng giải mã DES: cho phép người dùng giải mã văn bản. Yêu cầu người dùng nhập vào bản mã (ô Input) và key (ô Key). Sản phẩm sẽ thực hiện giải mã bản rõ và xuất kết quả là bản rõ (ô Output).

+ *Nút File Input:* cho phép người dùng lựa chọn File Word trong bộ nhớ làm đầu vào cho quá trình mã hoá hoặc quá trình giải mã. Sau khi lựa chọn, nội dung của File Word sẽ hiện thị lên ô Input.

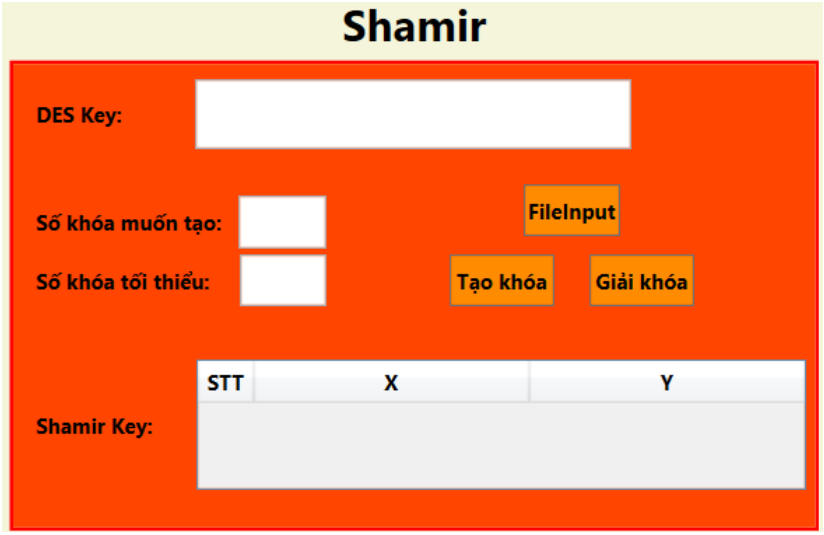
+ *Nút Key File:* cho phép người dùng nhập Key từ một file có đuôi .txt. Sau khi lựa chọn, nội dung của Key sẽ hiển thị lên ô Key.

+ *Nút Mã hoá:* cho phép người dùng thực hiện quá trình mã hoá sau khi đã nhập bản rõ và Key.

+ *Nút Giải mã:* cho phép người dùng thực hiện quá trình giải mã sau khi đã nhập bản mã và Key.

+ *Nút Xoá:* thực bắt đầu quá trình mã hoá hoặc giải mã mới.

4.2.2 Chức năng chia sẻ khoá DES bằng thuật toán SSS



Hình 12 Giao diện chia sẻ khóa Shamir

Chức năng tạo khoá: cho phép người dùng chia khoá DES. Yêu cầu người dùng nhập vào khoá DES muốn chia (khoá DES biểu diễn dưới dạng hệ Hexa và có 16 kí tự) (ô Khoá DES), số khoá mà người dùng muốn chia sẻ (ô Số khoá muốn tạo), sô khoá tối thiểu để khôi phục lại khoá DES hay còn gọi là ngưỡng (ô Số khoá tối thiểu). Sản phẩm sẽ thực hiện việc tạo ra số lượng khoá

bằng với số đã nhập trong ô (Số khoá muốn tạo) và hiển thị lên danh sách khoá Shamir.

Chức năng giải khoá: cho phép người dùng khôi phục lại khoá DES (với số khoá tối thiểu bằng ngưỡng). Yêu cầu người dùng nhập vào ngưỡng khôi phục (ô Số khoá tối thiểu), danh sách khoá Shamir (bảng khoá Shamir). Sản phẩm sẽ thực hiện khôi phục khoá DES ban đầu (khoá DES được biểu diễn dưới dạng hệ Hexa và có 16 kí tự).

+ *Nút DES Key File:* cho phép người dùng nhập Key từ một file có đuôi .txt. Sau khi lựa chọn, nội dung của Key sẽ hiển thị lên ô Key.

+ *Nút Shamir Key File:* cho phép người dùng nhập khoá Shamir từ một file có đuôi .txt. Sau khi lựa chọn, nội dung của khoá Shamir sẽ hiển thị lên bảng Shamir Key.

+ *Nút Tạo khoá:* cho phép người dùng bắt đầu quá trình tạo khoá Shamir sau khi đã nhập DES, số khoá muốn tạo, số khoá tối thiểu.

+ *Nút Khôi phục khoá:* cho phép người dùng bắt đầu quá trình khôi phục khoá DES sau khi đã nhập số khoá tối thiểu, danh sách khoá Shamir.

4.3 Kết quả thực hiện

4.3.1 Kết quả mã hoá và giải mã DES

*** Kết quả mã hoá DES:**

Người dùng nhập bản rõ và key (key được biểu diễn dưới hệ Hexa và có 16 kí tự). Ở đây, chúng ta sẽ ví dụ với những kí tự khó nhất (những kí tự có 2 dấu ví dụ như: ế, ể, ễ).

+ Input: **ế ể ệ ễ **** Tôi yêu đất nước Việt Nam**** ố ộ ợ ổ**

+ Key: **234ABCD4EFFFFFEE**

Sản phẩm thực hiện quá trình mã hoá và cho ra kết quả:

+ Output:

弊⁖□맡뻼씇녤넵寧?혔ɗ⊣櫄?廑색ꉊ忙□汅籛?蔯?飲□三⊙□輐갹

?佳弟ỗ樧嗎?ത?败ំ偾瑻숼12月□敢鏇

Hình 13 Demo mã hóa DES

*** Kết quả giải mã DES:**

Người dùng nhập vào bản mã và key (key được biểu diễn dưới hệ Hexa và có 16 kí tự). Ở đây, chúng ta sẽ thử ngược lại với trường hợp đã mã hoá ở trên để kiểm chứng và một trường hợp khoá bi thay đổi đi một kí tự:

+ Input:

弊⁏□맡삠໕녤녭寧⍰皛↤櫄⍰廑□섁⍢忙□汅㼿⍰蕨⍰飢□☰⍟□輐갹⍰佳弟ỗ榭嗎⍰ൌ⍰败̊偾瑣䒢12ꬰ□敢鏇

+ Key: *Trường hợp 1 là với key giống phần mã hoá ở trên:*

234ABCD4EFFFFFEE

=> Output: **ế ể ệ ễ **** Tôi yêu đất nước Việt Nam**** ố ộ ợ ổ**

Như vậy với Key đúng, chúng ta được bản rõ ban đầu.

Hình 14 Demo giải mã DES

Trường hợp 2 là với Key bị thay đổi đi một kí tự ở vị trí thứ 16 (từ kí tự ‘E’ thành kí tự ‘A’)

234ABCD4EFFFFFEA

=> Output:

› □嫘싓湘□⍰詟밮㊮蟦怾ǯ謖쑤犎踁룊遣좄轙□豣뾃◌⍰효詠輛觐穫솑⍰쉬 麛颇貙ȣ垺갦൭⍰꺄□ᕂ肮ꓒ녫ᄡ諿闧

Như vậy với key sai, chúng ta không thu được bản rõ ban đầu.

4.3.2 Kết quả chia sẻ bí mật Shamir

Hình 15 Demo trường hợp sai khóa

*** Kết quả chia sẻ khoá DES thành khoá con:**

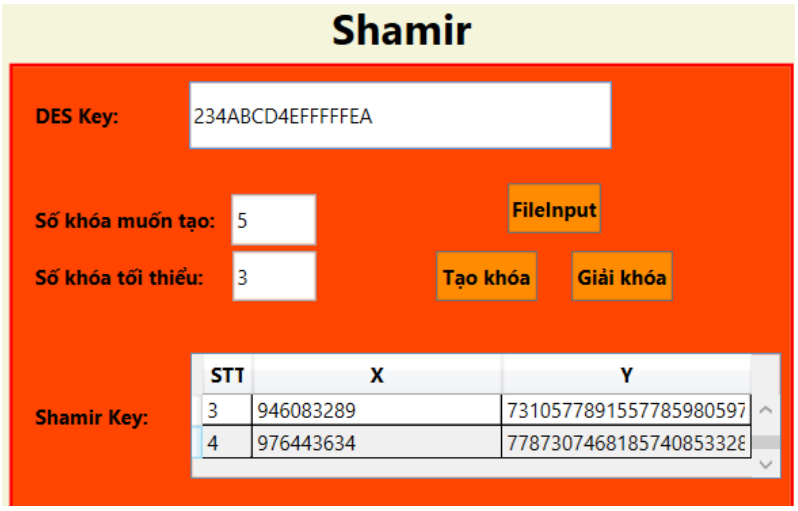
Người dùng nhập vào key DES (key được biểu diễn dưới hệ Hexa và có 16 kí tự), số khoá muốn tạo và số khoá tối thiểu.

+ Khoá DES: **234ABCD4EFFFFFEA**. Số khoá muốn tạo: **5** khoá. Số khoá tối thiểu: **3** khoá.

=> Output: là 5 cặp khoá con (x, y) để người dùng chia sẻ.

Bảng 16 Demo tạo khóa Shamir

STT	Khoá X	Khoá Y
1	1494401833	18240122379123516532935754 24
2	473652203	18323654490620196120146595 4
3	2126624692	36938109846393203100478096 30
4	946083289	73105778915577859805974484 8
5	976443634	77873074681857408533280392 8



Hình 16 Demo chia sẻ khóa Shamir

*** Kết quả khôi phục khoá DES:**

Người dùng nhập vào Số lượng khoá tối thiểu và danh sách cặp (x, y) khoá con. Ở đây, chúng ta sẽ nhập Số lượng khoá tối thiểu và số cặp (x, y) khoá con bằng số lượng khoá tối thiểu ở phần chia khoá để kiểm chứng và trường hợp nhập vào một khoá sai.

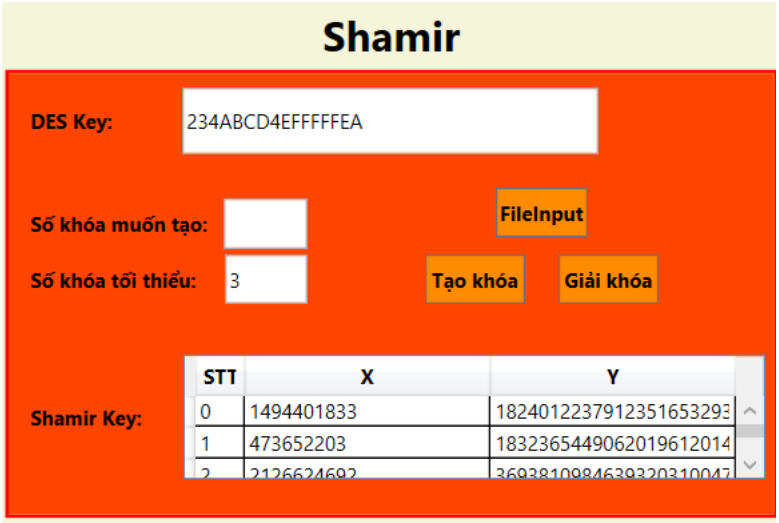
+ Số lượng khoá tối thiểu: **3**

+ Danh sách khoá con:

Trường hợp 1 là một danh sách các cặp khoá (x, y) như phần tạo ở trên:

Bảng 17 Demo khôi phục khóa

STT	Khoá X	Khoá Y
1	1494401833	18240122379123516532935754 24
2	473652203	18323654490620196120146595 4
3	2126624692	36938109846393203100478096 30



Hình 17 Demo khôi phục khóa

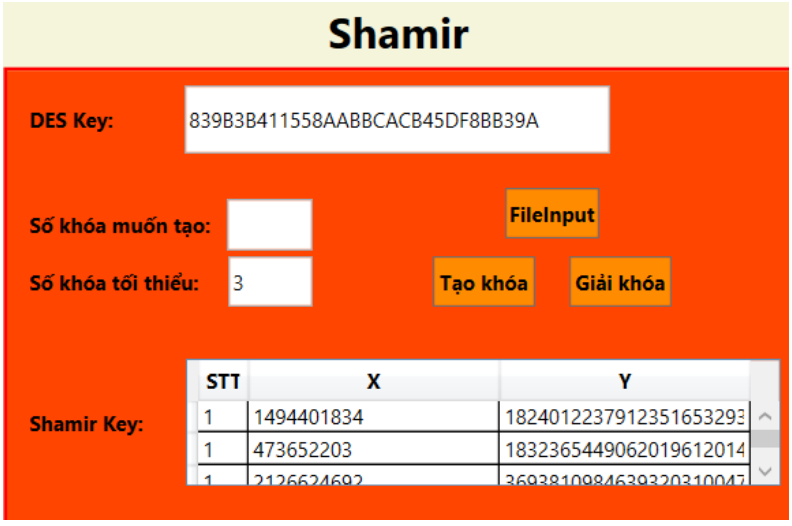
=> Output: **234ABCD4EFFFFFEA**. Là khoá DES ban đầu.

Trường hợp 2 là một danh sách các cặp khoá (x, y) ở trên nhưng sẽ đổi khoá thứ nhất đi một chữ số (từ "1494401833" thành "1494401834"):

Bảng 18 Demo trường hợp sai của Shamir

STT	Khoá X	Khoá Y
1	1494401834	1824012237912351653293575424
2	473652203	183236544906201961201465954
3	2126624692	369381098463932031004780963 0

=> Output: **839B3B411558AABBCACB45DF8BB39A**. Không phải khoá DES ban đầu.



Hình 18 Demo trường hợp nhập sai của Shamir

KẾT LUẬN

Sau khi tìm hiểu về mã hóa DES *(Data Encryption Standard)*, chúng ta có thể kết luận rằng DES đã đóng vai trò quan trọng trong lịch sử an ninh thông tin. Được giới thiệu vào những năm 1970, DES đã trở thành một tiêu chuẩn quốc tế và được sử dụng rộng rãi trong nhiều năm. Tuy nhiên, do sự phát triển của công nghệ và sự tiến bộ của các phương pháp tấn công, DES đã trở nên không an toàn đối với các môi trường yêu cầu độ bảo mật cao hơn.

Mặc dù DES đã lỗi thời, nhưng sự nghiên cứu về nó vẫn đóng vai trò quan trọng trong việc hiểu về lịch sử và phát triển của an ninh thông tin. Các bài học từ DES đã hỗ trợ chúng ta xây dựng các tiêu chuẩn an toàn thông tin ngày nay và thách thức chúng ta liên tục cải tiến để đảm bảo rằng dữ liệu của chúng ta được bảo vệ một cách hiệu quả trước những mối đe dọa ngày càng phức tạp.

Về đề tài nghiên cứu:

Ưu điểm: mã hoá DES là mã hoá đối xứng, tức là người nhận và người gửi phải thống nhất chung một khoá, việc truyền khoá này là không an toàn. Vì thế mà đề tài đã bổ sung thêm thuật toán chia sẻ bí mật Shamir để chia sẻ khoá DES, phần nào khắc phục được nhược điểm của mã hoá DES. Dựa trên thuật toán chia sẻ bí mật Shamir, chúng ta có thể chia sẻ khoá cho nhiều người, ứng dụng trong việc một bí mật cần có tối thiểu "ngưỡng" người mới có thể khôi phục bí mật.

Nhược điểm: nhược điểm của đề tài nằm trong nhược điểm của mã hoá DES bao gồm nhược điểm của khoá chỉ có 64 bits (trong đó có 8 bits dùng để kiểm tra tính chẵn lẻ) không còn an toàn với các máy tính hiện đại, tính bù, tính khoá yếu. Tiếp theo là nhược điểm của thuật toán chia sẻ bí mật Shamir, nếu "ngưỡng" quá nhỏ thì bí mật không còn được an toàn; thuật toán chia sẻ bí mật Shamir chia sẻ n khoá con và tối thiểu k khoá con thì có thể khôi phục lại, các khoá con này vẫn được chia sẻ trên đường truyền và trong trường hợp xấu nhất, kẻ đánh cắp có thể thu thập được k khoá con này thì bí mật không còn được an toàn.

Hướng phát triển tương lai: nhóm sẽ phát triển mã hoá 3DES dựa trên mã hoá DES có sẵn nhằm khắc phục nhược điểm về khoá. Thuật toán chia sẻ bí mật Shamir sẽ được nhóm hướng tới việc áp dụng thêm số học Modulo gây khó khăn cho việc thám mã.

TÀI LIỆU THAM KHẢO

[1]. TS. Lê Thị Anh, *Slide bài giảng An toàn và bảo mật thông tin*, khoa Công nghệ thông tin, Trường Đại học Công nghiệp Hà Nội.

[2]. Nhóm tác giả, *Giáo trình An toàn và bảo mật thông tin*, khoa Công nghệ thông tin, Trường Đại học Hàng hải.

[3]. *https://www.geeksforgeeks.org/shamirs-secret-sharing-algorithm-cryptography/*

[4]. *https://evervault.com/blog/shamir-secret-sharing*

[5]. *https://en.wikipedia.org/wiki/Shamir%27s_secret_sharing*