LỜI NÓI ĐẦU

Ngày nay, với sự phát triển mạnh mẽ của chuyển đổi số đã làm thay đổi cuộc sống của chúng ta. Cách chúng ta sống, làm việc, vui chơi và học tập đều đã thay đổi. Tuy nhiên, một trong những thách thức đặt ra là vấn đề an ninh và an toàn dữ liệu, mọi tổ chức khi cung cấp các dịch vụ đều phải bảo vệ an toàn dữ liệu và an ninh hệ thống mạng của mình. An toàn bảo mật thông tin cũng giúp tổ chức bảo vệ thông tin độc quyền khỏi bị tấn công, đồng thời góp phần bảo vệ danh tiếng của tổ chức. Nhận thức được vai trò to lớn của an toàn và bảo mật thông tin, Khoa Công nghệ thông tin của Trường Đại học Công nghiệp Hà Nội đã sớm đưa học phần An toàn và bảo mật thông tin vào các chương trình đào tạo của mình. Để tạo điều kiện thuận lợi cho việc học tập và nghiên cứu của các sinh viên khoa Công nghệ thông tin, nhóm giảng viên giảng dạy học phần An toàn và bảo mật thông tin đã tổ chức biên soạn giáo trình này. Nội dung của giáo trình gồm 6 chương:

Chương 1: Tổng quan về bảo mật, nội dung của chương chủ yếu tập trung trình bày khái niệm về an toàn và bảo mật thông tin, khái niệm về bảo mật dữ liệu, mục tiêu của an toàn và bảo mật thông tin, sự tấn công và mục đích tấn công, các loại điểm yếu và loại tội phạm, các biện pháp nhằm đảm bảo an toàn và bảo mật thông tin. Chương này được biên soạn bởi Thạc sĩ Trần Phương Nhung.

Chương 2: Cơ sở toán học, nội dung của chương này tập trung trình bày về cơ sở toán học của lý thuyết mật mã là số nguyên tố và số học đồng dư (modulo) như khái niệm về đồng dư, quan hệ tương đương, phép toán số học trên modulo, ước số, ước số chung lớn nhất, thuật toán Ơclit để tìm ước số chung lớn nhất, khái niệm số nguyên tố, định lý Fermat và Ơle, kiểm tra tính nguyên tố và định lý phần dư Trung Hoa. Chương này được biên soạn bởi chủ biên Nguyễn Bá Nghiễn.

Chương 3: Mã cổ điển, nội dung của chương này tập trung trình bày các phương pháp mã hóa cổ điển như Caesar, Vigenere, Affine, Hill.... Cài đặt và thử nghiệm các giải thuật mã hóa và giải mã. Chương này được biên soạn bởi Thạc sỹ Trần Phương Nhung.

Chương 4: Chuẩn mã dữ liệu DES và chuẩn mã nâng cao AES, chương này tác giả tập trung trình bày nguyên lý của mã hóa đối xứng hiện đại. Chương này được biên soạn bởi chủ biên Nguyễn Bá Nghiễn.

Chương 5: Mã hóa công khai và quản lý khóa, nội dung của chương này tác giả tập trung trình bày về nguyên lý của mã hóa bất đối xứng hay còn gọi là mã hóa khóa công khai, trong đó trình bày chi tiết một số mật mã bất đối xứng được sử dụng rộng rãi đó là RSA và trao đổi khóa Diffie-Hellman. Nội dung của chương được biên soạn bởi Tiến sỹ Đặng Trọng Hợp.

Chương 6: An toàn IP và web, nội dung của chương này tác giả tập trung trình bày về bảo mật hệ thống mạng ở cấp độ IP, bảo mật website và an toàn thư điện tử, trong đó trình bày chi tiết các giao thức IPSec, giao thức SSL, giao thức TLS, dịch vụ PGP, dịch vụ S/MIME. Nội dung của chương được biên soạn bởi Thạc sỹ Phạm Văn Hiệp.

Mặc dù nhóm tác giả đã cố gắng, tuy nhiên cuốn giáo trình không thể tránh khỏi thiếu sót. Nhóm tác giả rất mong nhận được các ý kiến đóng góp từ các đồng nghiệp và bạn đọc để cuốn giáo trình ngày càng hoàn thiện hơn. Một lần nữa nhóm tác giả xin chân thành cảm ơn đến gia đình, bạn bè và đồng nghiệp đã động viên, đóng góp các ý kiến quý báu để nhóm tác giả hoàn thành cuốn giáo trình này.

NHÓM TÁC GIẢ

MỤC LỤC

DANH MỤC HÌNH ẢNH

DANH MỤC BẢNG BIỂU

CHƯƠNG 1. TỔNG QUAN VỀ BẢO MẬT

Chương này tập trung trình bày các khái niệm về an toàn thông tin và bảo mật thông tin, khái niệm về bảo mật dữ liệu, mục tiêu của an toàn thông tin và bảo mật thông tin, sự tấn công và mục đích tấn công, các loại điểm yếu và loại tội phạm, các biện pháp nhằm đảm bảo an toàn thông tin và bảo mật thông tin

1.1 Giới thiệu chung về bảo mật thông tin

1.1.1 Mở đầu về bảo mật thông tin

Ngày nay với sự phát triển bùng nổ của công nghệ thông tin, hầu hết các thông tin của doanh nghiệp như chiến lược kinh doanh, các thông tin về khách hàng, nhà cung cấp, tài chính, mức lương nhân viên,…đều được lưu trữ trên hệ thống máy tính. Cùng với sự phát triển của doanh nghiệp là những đòi hỏi ngày càng cao của môi trường kinh doanh yêu cầu doanh nghiệp cần phải chia sẻ thông tin của mình cho nhiều đối tượng khác nhau qua Internet hay Intranet. Việc mất mát, rò rỉ thông tin có thể ảnh hưởng nghiêm trọng đến tài chính, danh tiếng của công ty và quan hệ với khách hàng.

Các phương thức tấn công thông qua mạng ngày càng tinh vi, phức tạp có thể dẫn đến mất mát thông tin, thậm chí có thể làm sụp đổ hoàn toàn hệ thống thông tin của doanh nghiệp. Vì vậy an toàn và bảo mật thông tin là nhiệm vụ rất nặng nề và khó đoán trước được, nhưng tựu trung lại gồm ba hướng chính sau:

- Bảo đảm an toàn thông tin tại máy chủ
- Bảo đảm an toàn cho phía máy trạm
- Bảo mật thông tin trên đường truyền

Đứng trước yêu cầu bảo mật thông tin, ngoài việc xây dựng các phương thức bảo mật thông tin thì người ta đã đưa ra các nguyên tắc về bảo vệ dữ liệu như sau:

- Nguyên tắc hợp pháp trong lúc thu thập và xử lý dữ liệu.
- Nguyên tắc đúng đắn.
- Nguyên tắc phù hợp với mục đích.
- Nguyên tắc cân xứng.
- Nguyên tắc minh bạch.
- Nguyên tắc được cùng quyết định cho từng cá nhân và bảo đảm quyền truy cập cho người có liên quan.
- Nguyên tắc không phân biệt đối xử.
- Nguyên tắc an toàn.

- Nguyên tắc có trách nhiệm trước pháp luật.

- Nguyên tắc giám sát độc lập và hình phạt theo pháp luật.

- Nguyên tắc mức bảo vệ tương ứng trong vận chuyển dữ liệu xuyên biên giới.

Ở đây chúng ta sẽ tập trung xem xét các nhu cầu an ninh và đề ra các biện pháp an toàn cũng như vận hành các cơ chế để đạt được các mục tiêu đó. An toàn thông tin bao gồm các nội dung sau:

- Tính bí mật: tính kín đáo riêng tư của thông tin

- Tính xác thực của thông tin, bao gồm xác thực đối tác(bài toán nhận danh), xác thực thông tin trao đổi.

- Tính trách nhiệm: đảm bảo người gửi thông tin không thể thoái thác trách nhiệm về thông tin mà mình đã gửi.

1.1.2 Nguy cơ và hiểm họa đối với hệ thống thông tin

Các hiểm họa đối với hệ thống có thể được phân loại thành hiểm họa vô tình hay cố ý, chủ động hay thụ động.

- Hiểm họa vô tình: khi người dùng khởi động lại hệ thống ở chế độ đặc quyền, họ có thể tùy ý chỉnh sửa hệ thống. Nhưng sau khi hoàn thành công việc họ không chuyển hệ thống sang chế độ thông thường, vô tình để kẻ xấu lợi dụng.

- Hiểm họa cố ý: như cố tình truy nhập hệ thống trái phép.

- Hiểm họa thụ động: là hiểm họa nhưng chưa hoặc không tác động trực tiếp lên hệ thống, như nghe trộm các gói tin trên đường truyền.

- Hiểm họa chủ động: là việc sửa đổi thông tin, thay đổi tình trạng hoặc hoạt động của hệ thống.

Đối với mỗi hệ thống thông tin mối đe dọa và hậu quả tiềm ẩn là rất lớn, nó có thể xuất phát từ những nguyên nhân như sau:

- Từ phía người sử dụng: xâm nhập bất hợp pháp, ăn cắp tài sản có giá trị

- Trong kiến trúc hệ thống thông tin: tổ chức hệ thống kỹ thuật không có cấu trúc hoặc không đủ mạnh để bảo vệ thông tin.

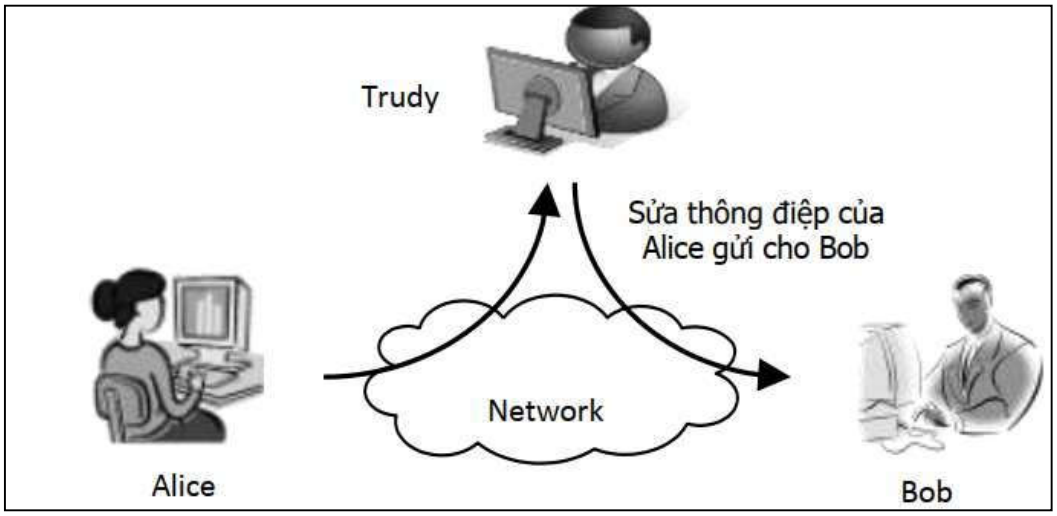
- Ngay trong chính sách bảo mật an toàn thông tin: không chấp hành các chuẩn an toàn, không xác định rõ các quyền trong vận hành hệ thống.

- Thông tin trong hệ thống máy tính cũng sẽ dễ bị xâm nhập nếu không có công cụ quản lý, kiểm tra và điều khiển hệ thống.

- Nguy cơ nằm ngay trong cấu trúc phần cứng của các thiết bị tin học và trong phần mềm hệ thống và ứng dụng do hãng sản xuất cài sẵn các loại 'rệp' điện tử theo ý đồ định trước, gọi là 'bom điện tử'.

- Nguy hiểm nhất đối với mạng máy tính mở là tin tặc, từ phía bọn tội phạm.

1.1.3 Phân loại tấn công phá hoại an toàn



Hình 1. 1 Mô hình truyền tin bị tấn công

Các hệ thống trên mạng có thể là đối tượng của nhiều kiểu tấn công:

- Tấn công giả mạo là một thực thể tấn công giả danh một thực thể khác. Tấn công giả mạo thường được kết hợp với các dạng tấn công khác như tấn công chuyển tiếp và tấn công sửa đổi thông báo.

- Tấn công chuyển tiếp xảy ra khi một thông báo, hoặc một phần thông báo được gửi nhiều lần, gây ra các tác động tiêu cực.

- Tấn công sửa đổi thông báo xảy ra khi nội dung của một thông báo bị sửa đổi nhưng không bị phát hiện.

- Tấn công từ chối dịch vụ là kiểu tấn công ngăn không cho những người dùng khác truy cập vào hệ thống, làm cho hệ thống bị quá tải và không thể hoạt động. DoS là tấn công "one-to-one" và DDoS(distributed denial of service) là sử dụng các Zombie host tấn công "many-to-one". Hiện nay, các hình thức tấn công từ chối dịch vụ DoS (Denial of Service) và DDoS được đánh giá là các nguy cơ lớn nhất đối với sự an toàn của các hệ thống thông tin, gây ra những thiệt hại lớn và đặc biệt là chưa có giải pháp ngăn chặn hữu hiệu. Các hình thức tấn công này đều nhắm vào tính khả dụng của hệ thống.

- Tấn công từ bên trong hệ thống xảy ra khi người dùng hợp pháp cố tình hoặc vô ý can thiệp hệ thống trái phép. Còn tấn công từ bên ngoài là nghe trộm, thu chặn, giả mạo người dùng hợp pháp và vượt quyền hoặc lách qua các cơ chế kiểm soát truy nhập.

- Tấn công bị động là do thám, theo dõi đường truyền để: nhận được nội dung bản tin hoặc theo dõi luồng truyền tin.

- Tấn công chủ động là thay đổi luồng dữ liệu để: giả mạo một người nào đó, lặp lại bản tin trước, thay đổi ban tin khi truyền, từ chối dịch vụ.

- Tấn công bằng mã nguy hiểm là dùng một đoạn mã không mong muốn được nhúng trong một chương trình nhằm thực hiện các truy cập trái phép vào hệ thống máy tính để thu thập các thông tin nhạy cảm, làm gián đoạn hoạt động hoặc gây hại cho hệ thống máy tính, bao gồm: virus, worm, trojan horses, spyware, adware, backdoor,…

1.2 Dịch vụ cơ chế tấn công

Nhu cầu thực tiễn dẫn đến sự cần thiết có một phương pháp hệ thống xác định các yêu cầu an ninh của tổ chức. Trong đó cần có tiếp cận tổng thể xét cả ba khía cạnh của an toàn thông tin: bảo vệ tấn công, cơ chế an toàn và dịch vụ an toàn.

1.2.1 Các dịch vụ an toàn

Đây là công cụ đảm bảo an toàn của hệ thống xử lý thông tin và truyền thông tin trong tổ chức. Chúng được thiết lập để chống lại các tấn công phá hoại. Có thể dùng một hay nhiều cơ chế an toàn để cung cấp dịch vụ.

Thông thường người ta cần phải tạo ra các liên kết với các tài liệu vật lý: như có chữ ký, ngày tháng, bảo vệ cần thiết chống khám phá, sửa bậy, phá hoại, được công chứng, chứng kiến, được ghi nhận hoặc có bản quyền.

1.2.2 Các cơ chế an toàn

Từ các công việc thực tế để chống lại các phá hoại an ninh, người ta đã hệ thống và sắp xếp lại tạo thành các cơ chế an ninh khác nhau. Đây là cơ chế được thiết kế để phát hiện, bảo vệ hoặc khôi phục do tấn công phá hoại.

Không có cơ chế đơn lẻ nào đáp ứng được mọi chức năng yêu cầu của công tác an ninh. Tuy nhiên có một thành phần đặc biệt nằm trong mọi cơ chế an toàn đó là: kỹ thuật mã hoá. Do đó chúng ta sẽ dành một thời lượng nhất định tập trung vào lý thuyết mã.

1.2.3 Tấn công phá hoại an ninh

Ta xác định rõ thế nào là các hành động tấn công phá hoại an ninh. Đó là mọi hành động chống lại sự an toàn thông tin của các tổ chức.

An toàn thông tin là bàn về bằng cách nào chống lại tấn công vào hệ thống thông tin hoặc phát hiện ra chúng. Trên thực tế có rất nhiều cách và nhiều kiểu tấn công khác nhau. Thường thuật ngữ đe dọa và tấn công được dùng như nhau. Cần tập trung chống một số kiểu tấn công chính: thụ động và chủ động.

1.3 Mô hình an toàn mạng

1.3.1 Kiến trúc an toàn của hệ thống truyền thông mở OSI.

Để giúp cho việc hoạch định chính sách và xây dựng hệ thống an ninh tốt. Bộ phận chuẩn hóa tiêu chuẩn của tổ chức truyền thông quốc tế (International Telecommunication Union) đã nghiên cứu và đề ra Kiến trúc an ninh X800 dành cho hệ thống trao đổi thông tin mở OSI. Trong đó định nghĩa một cách hệ thống phương pháp xác định và cung cấp các yêu cầu an toàn.Nó cung cấp cho chúng ta một cách nhìn tổng quát, hữu ích về các khái niệm mà chúng ta nghiên cứu.

Trước hết nói về dịch vụ an toàn, X800 định nghĩa đây là dịch vụ cung cấp cho tầng giao thức của các hệ thống mở trao đổi thông tin, mà đảm bảo an toàn thông tin cần thiết cho hệ thống và cho việc truyền dữ liệu.

Trong tài liệu các thuật ngữ chuẩn trên Internet RFC 2828 đã nêu định nghĩa cụ thể hơn dịch vụ an toàn là dịch vụ trao đổi và xử lý cung cấp cho hệ thống việc bảo vệ đặc biệt cho các thông tin nguồn.Tài liệu X800 đưa ra định nghĩa dịch vụ theo 5 loại chính:

- Xác thực: tin tưởng là thực thể trao đổi đúng là cái đã tuyên bố. Người đang trao đổi xưng tên với mình đúng là anh ta, không cho phép người khác mạo danh.

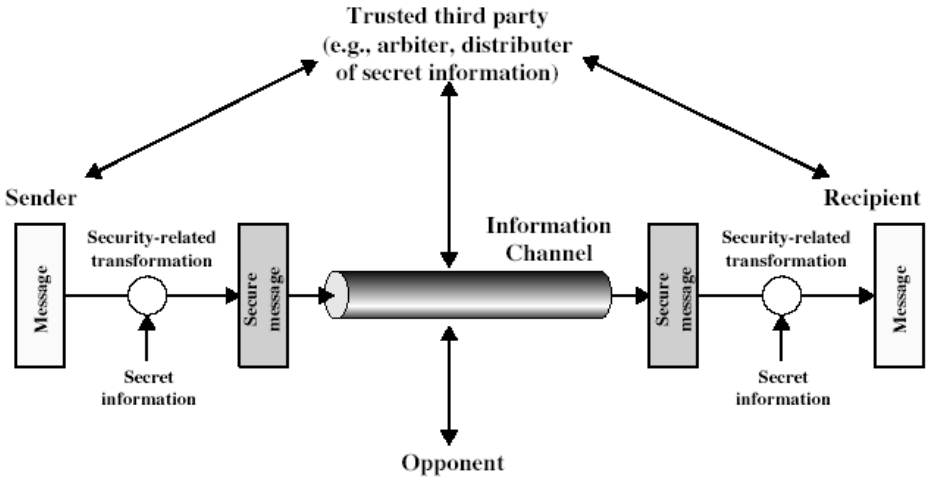
- Quyền truy cập: ngăn cấm việc sử dụng nguồn thông tin không đúng vai trò. Mỗi đối tượng trong hệ thống được cung cấp các quyền hạn nhất định và chỉ được hành động trong khuôn khổ các quyền hạn đó.

- Bảo mật dữ liệu: bảo vệ dữ liệu không bị khám phá bởi người không có quyền. Chẳng hạn như dùng các ký hiệu khác để thay thế các ký hiệu trong bản tin, mà chỉ người có bản quyền mới có thể khôi phục nguyên bản của nó.

- Toàn vẹn dữ liệu: tin tưởng là dữ liệu được gửi từ người có quyền. Nếu có thay đổi như làm trì hoãn về mặt thời gian hay sửa đổi thông tin, thì xác thực sẽ cho cách kiểm tra nhận biết là có các hiện tượng đó đã xảy ra.

- Không từ chối: chống lại việc chối bỏ của một trong các bên tham gia trao đổi. Người gửi cũng không chối bỏ là mình đã gửi thông tin với nội dung như vậy và người nhận không thể nói dối là tôi chưa nhận được thông tin đó. Điều này là rất cần thiết trong việc trao đổi, thỏa thuận thông tin hàng ngày.

1.3.2 Mô hình an toàn mạng tổng quát



Hình 1. 2 Mô hình truy cập mạng an toàn

Sử dụng mô hình trên đòi hỏi chúng ta phải thiết kế:

- Thuật toán phù hợp cho việc truyền an toàn.

- Phát sinh các thông tin mật (khoá) được sử dụng bởi các thuật toán.

- Phát triển các phương pháp phân phối và chia sẻ các thông tin mật.

- Đặc tả giao thức cho các bên để sử dụng việc truyền và thông tin mật cho các dịch vụ an toàn.

- Lựa chọn hàm canh cổng phù hợp cho người sử dụng có danh tính.

- Cài đặt kiểm soát quyền truy cập để tin tưởng rằng chỉ có người có quyền mới truy cập được thông tin đích hoặc nguồn.

- Các hệ thống máy tính tin cậy có thể dùng mô hình này.

1.4 Bảo mật thông tin trong hệ cơ sở dữ liệu

1.4.1 Giới thiệu chung

Các hệ cơ sở dữ liệu (CSDL) ngày nay như Oracle, SQL/Server, DB2/Informix đều có sẵn các công cụ bảo vệ tiêu chuẩn như hệ thống định danh và kiểm soát truy xuất. Tuy nhiên, các biện pháp bảo vệ này hầu như không có tác dụng trước các tấn công từ bên trong. Để bảo vệ thông tin khỏi mối đe dọa này, người ta đưa ra hai giải pháp.

Giải pháp đơn giản nhất bảo vệ dữ liệu trong CSDL ở mức độ tập tin, chống lại sự truy cập trái phép vào các tập tin CSDL bằng hình thức mã hóa. Tuy nhiên, giải pháp này không cung cấp mức độ bảo mật truy cập đến CSDL ở mức độ bảng, cột và dòng. Một điểm yếu nữa của giải pháp này là bất cứ ai với quyền truy xuất CSDL đều có thể

truy cập vào tất cả dữ liệu trong CSDL cũng có nghĩa là cho phép các đối tượng với quyền quản trị truy cập tất cả các dữ liệu nhạy cảm.

Giải pháp thứ hai, giải quyết vấn đề mã hóa ở mức ứng dụng. Giải pháp này xử lý mã hóa dữ liệu trước khi truyền dữ liệu vào CSDL. Những vấn đề về quản lý khóa và quyền truy cập được hỗ trợ bởi ứng dụng. Truy vấn dữ liệu đến CSDL sẽ trả kết quả dữ liệu ở dạng mã hóa và dữ liệu này sẽ được giải mã bởi ứng dụng. Giải pháp này giải quyết được vấn đề phân tách quyền an toàn và hỗ trợ các chính sách an toàn dựa trên vai trò.

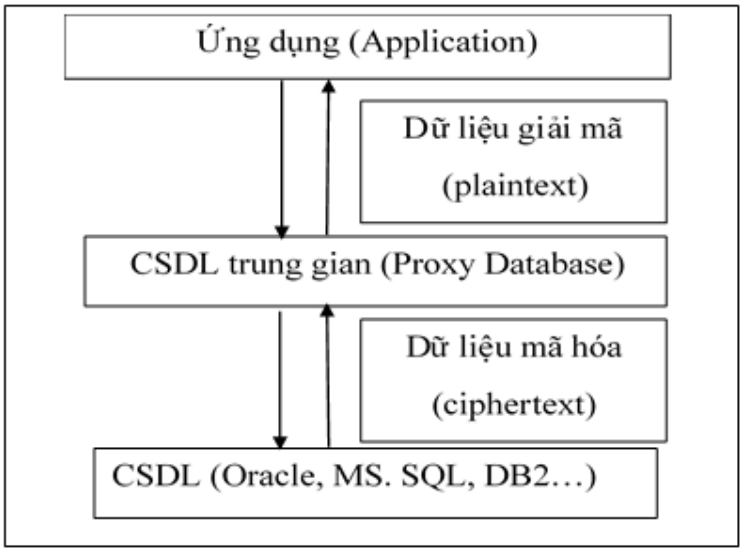
1.4.2 Một số mô hình bảo mật cơ sở dữ liệu

Để đáp ứng những yêu cầu về bảo mật cho các hệ thống CSDL hiện tại và sau này người ta đưa ra 2 mô hình bảo mật CSDL thông thường sau đây.

Xây dựng tầng CSDL trung gian

Một CSDL trung gian được xây dựng giữa ứng dụng và CSDL gốc. CSDL trung gian này có vai trò mã hóa dữ liệu trước khi cập nhật vào CSDL gốc, đồng thời giải mã dữ liệu trước khi cung cấp cho ứng dụng. CSDL trung gian đồng thời cung cấp thêm các chức năng quản lý khóa, xác thực người dùng và cấp phép truy cập.

Giải pháp này cho phép tạo thêm nhiều chức năng về bảo mật cho CSDL. Tuy nhiên, mô hình CSDL trung gian đòi hỏi xây dựng một ứng dụng CSDL tái tạo tất cả các chức năng của CSDL gốc.



Hình 1. 3 Mô hình Proxy

Sử dụng cơ chế sẵn có trong CSDL

Mô hình này giải quyết các vấn đề mã hóa cột dựa trên các cơ chế sau:

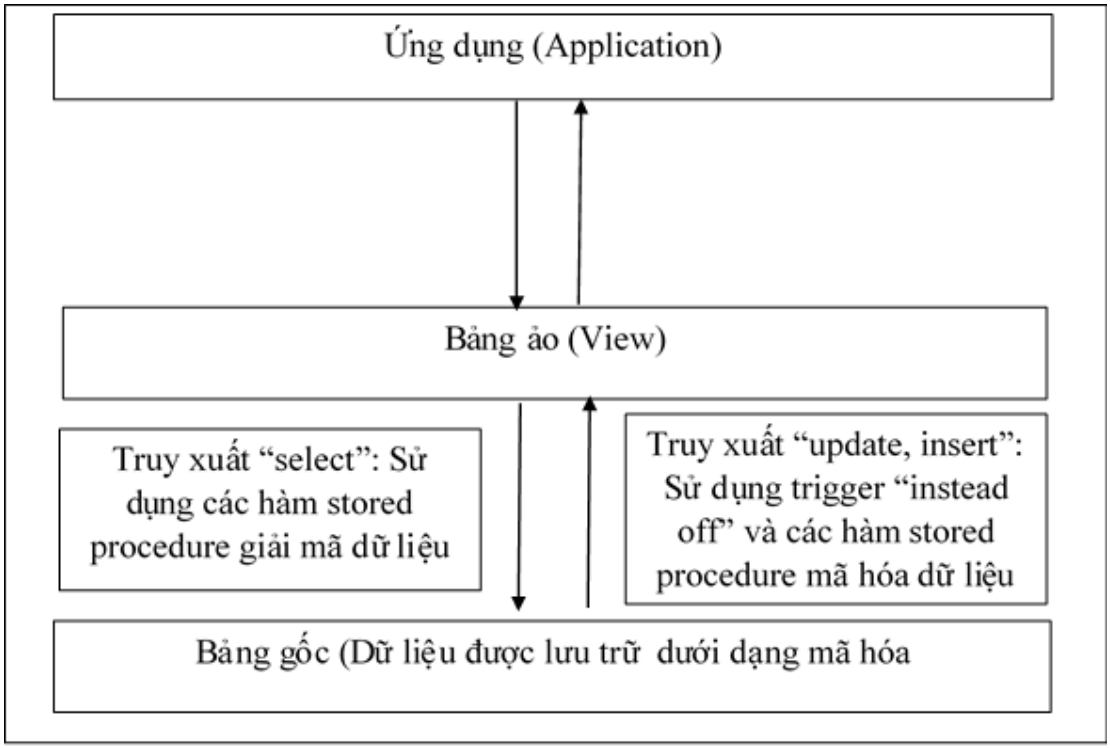
a. Các hàm Stored Procedure trong CSDL cho chức năng mã hóa và giải mã

b. Sử dụng cơ chế View trong CSDL tạo các bảng ảo, thay thế các bảng thật đã được mã hóa.

c. Cơ chế "instead of" trigger được sử dụng nhằm tự động hóa quá trình mã hóa từ View đến bảng gốc.

Trong mô hình này, dữ liệu trong các bảng gốc sẽ được mã hóa, tên của bảng gốc được thay đổi. Một bảng ảo được tạo ra mang tên của bảng gốc, ứng dụng sẽ truy cập đến bảng ảo này.

Truy xuất dữ liệu trong mô hình này có thể được tóm tắt như sau:



Hình 1. 4 Mô hình bảng ảo

Các truy xuất dữ liệu đến bảng gốc sẽ được thay thế bằng truy xuất đến bảng ảo.

Bảng ảo được tạo ra để mô phỏng dữ liệu trong bảng gốc. Khi thực thi lệnh "select", dữ liệu sẽ được giải mã cho bảng ảo từ bảng gốc (đã được mã hóa). Khi thực thi lệnh "Insert, Update", "instead of" trigger sẽ được thi hành và mã hóa dữ liệu xuống bảng gốc.

Quản lý phân quyền truy cập đến các cột sẽ được quản lý ở các bảng ảo. Ngoài các quyền cơ bản do CSDL cung cấp, hai quyền truy cập mới được định nghĩa:

- Người sử dụng chỉ được quyền đọc dữ liệu ở dạng mã hóa. Quyền này phù hợp với những đối tượng cần quản lý CSDL mà không cần đọc nội dung dữ liệu.
- Người sử dụng được quyền đọc dữ liệu ở dạng giải mã.

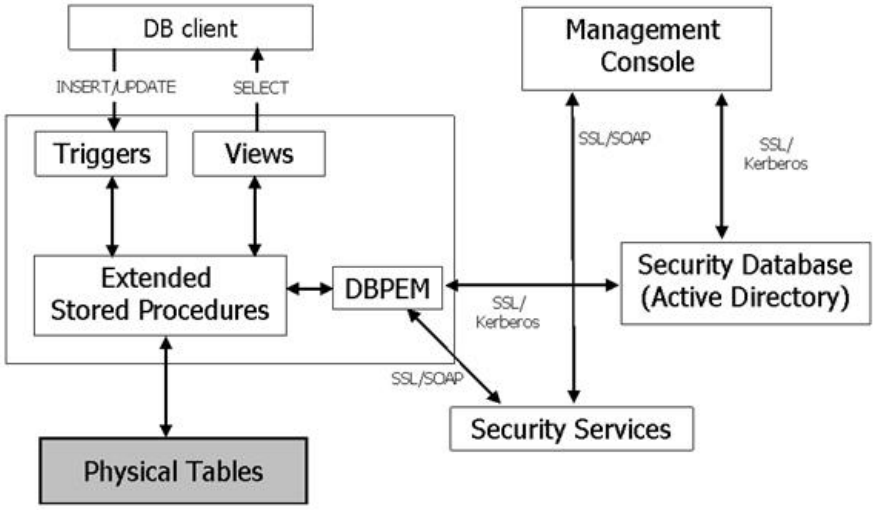
1.4.3 Sơ lược kiến trúc của 1 hệ bảo mật CSDL

Triggers: các trigger được sử dụng để lấy dữ liệu đến từ các câu lệnh INSERT, UPDATE (để mã hóa).

Views: các view được sử dụng để lấy dữ liệu đến từ các câu lệnh SELECT (để giải mã).

Extended Stored Procedures: được gọi từ các Trigger hoặc View dùng để kích hoạt các dịch vụ được cung cấp bởi Modulo DBPEM từ trong môi trường của hệ quản tri CSDL.

DBPEM (Database Policy Enforcing Modulo): cung cấp các dịch vụ mã hóa/giải mã dữ liệu gửi đến từ các Extended Stored Procedures và thực hiện việc kiểm tra quyền truy xuất của người dùng (dựa trên các chính sách bảo mật được lưu trữ trong CSDL về quyền bảo mật).



Hình 1. 5. Kiến trúc một hệ bảo mật CSDL

Security Database: lưu trữ các chính sách bảo mật và các khóa giải mã. Xu hướng ngày nay thường là lưu trữ CSDL về bảo mật này trong Active Directory (một CSDL dạng thư mục để lưu trữ tất cả thông tin về hệ thống mạng).

Security Services: chủ yếu thực hiện việc bảo vệ các khóa giải mã được lưu trong CSDL bảo mật.

Management Console: dùng để cập nhật thông tin lưu trong CSDL bảo mật (chủ yếu là soạn thảo các chính sách bảo mật) và thực hiện thao tác bảo vệ một trường nào đó trong CSDL để đảm bảo tối đa tính bảo mật, thông tin được trao đổi.

Trong chương này giáo trình đã trình bày các kiến thức căn bản về an toàn và bảo mật thông tin. Chương tiếp theo giáo trình sẽ trình bày về cơ sở toán học như số đồng dư, thuật toán Ơclit và Ơclit mở rộng, số học trên vành Z_n …

Câu hỏi và bài tập

Câu 1. Trình bày các khái niệm về an toàn thông tin và bảo mật thông tin?

Câu 2. Vai trò của an toàn thông tin và bảo mật thông tin là gì?

Câu 3. Các nguy cơ tấn công vào hệ thống thông tin?

Câu 4. Các yêu cầu cũng như mục tiêu của việc đảm bảo an toàn và bảo mật thông tin?

Câu 5. Quy trình và mô hình đảm bảo an toàn thông tin và bảo mật thông tin?

Câu 6. Định hướng để tăng cường an toàn thông tin và bảo mật thông tin?

Câu 7. Trình bày mô hình mạng an toàn?

Câu 8. Trình bày kiến trúc bảo mật CSDL?

CHƯƠNG 2. CƠ SỞ TOÁN HỌC

Chương này tập trung trình bày cơ sở toán học của lý thuyết mật mã là số nguyên tố và số học đồng dư (modulo) như khái niệm về đồng dư, quan hệ tương đương, phép toán số học trên modulo, ước số, ước số chung lớn nhất, thuật toán Ơclit để tìm ước số chung lớn nhất, khái niệm số nguyên tố, định lý Fermat và Ơle, kiểm tra tính nguyên tố và định lý phần dư Trung Hoa.

2.1 Số học đồng dư (modulo)

2.1.1 Định nghĩa

Cho một số nguyên *a* và số nguyên dương *n* bất kỳ, thực hiện phép chia *a* cho *n* thì thu được thương số *q* và phần dư *r* thỏa mãn mối quan hệ sau:

$a = q \times +r \qquad 0 \leq r < n$

Ví dụ minh họa trên bảng 2.1 sau:

Bảng 2. 1 Minh họa thương số và phần dư khi thực hiện phép chia a cho n

a = 13	*n = 4*	*13 = 3 x 4 + 1*	*q = 3*	*r = 1*
a = -13	*n = 4*	*-13 = (-4) x 4 + 3*	*q = -4*	*r = 3*

Tóm lại, cho một số nguyên *a* và số nguyên dương *n* thì ta định nghĩa *a mod n* là phần dư của phép chia *a* cho *n*.

Ví dụ: *13 mod 4 = 1* và *-13 mod 4 = 3*

Hai số nguyên a và b được gọi là đồng dư modulo với n nếu *(a mod n) = (b mod n)* và được ký hiệu như sau: $a \equiv b \ (mod\ n)$.

Ví dụ: $13 \equiv 5 \ (mod\ 4) \quad vì\ 13\ mod\ 4 = 1\ và\ 5\ mod\ 4 = 1$

$7 \equiv -13 \ (mod\ 4) \quad vì\ 7\ mod\ 4 = 3\ và - 13\ mod\ 4 = 3$

2.1.2 Ước số

Ta nói rằng *a* chia hết cho số khác không *b* nếu tồn tại số *m* nào đó để *a = m.b*, trong đó *m, a, b* là các số nguyên. *a* chia hết cho *b* được kí hiệu là *b|a* và *b* được gọi là ước số của *a*.

Ví dụ các ước số dương của 15 là 1, 3, 5 và 15.

2.1.3 Các tính chất của đồng dư

1. $a \equiv b(mod\ n) \quad nếu\ \ n|(a - b)$

Chứng minh: Nếu $n|(a-b)$ thì tồn tại số m sao cho $(a - b) = m.n \rightarrow a = b + m.n$. Do đó, a mod n chính là phần dư của phép chia $(b + m.n)$ cho n bằng phần dữ của phép chia b cho n và chính là $b \ mod \ n$.

Ví dụ: $23 \equiv 8 \ (mod \ 5) \ \ bởi \ vì \ 23 - 8 = 15 = 3 \times 5,$

$$11 \equiv -5 \ (mod \ 8) \ \ bởi \ vì \ 11 - (-5) = 11 + 5 = 16 = 2 \times 8$$

2. $a \equiv b(mod \ n) \quad kéo \ theo \ b \equiv a(mod \ n)$
3. $a \equiv b(mod \ n) \ \ và \ b \equiv c(mod \ n) \ \ kéo \ theo \ a \equiv c(mod \ n)$

2.1.4 Các phép toán trên modulo

Ta có thể áp dụng phép toán cộng, trừ, nhân trên modulo như sau:

1. $[(a \ mod \ n) + (b \ mod \ n)] \ mod \ n = (a + b) \ mod \ n$

   Chứng minh: Theo định nghĩa $a \ mod \ n = r1 \rightarrow a = i \times n + r1$ và $b \ mod \ n = r2 \rightarrow b = j \times n + r2$. Do đó, $(a + b) \ mod \ n = (i \times n + r1 + j \times n + r2) \ mod \ n = (r1 + r2 + (i + j) \times n) \ mod \ n = (r1 + r2) \ mod \ n = [(a \ mod \ n) + (b \ mod \ n)] \ mod \ n.$

   Ví dụ:

   11 mod 8 = 3 và 15 mod 8 = 7. Do đó *[(11 mod 8) + (15 mod 8)] mod 8 = (3 + 7) mod 8 = 10 mod 8 = 2*. Ta lại có, *(11+15) mod 8 = 26 mod 8 = 2*.

2. $[(a \ mod \ n) - (b \ mod \ n)] \ mod \ n = (a - b) \ mod \ n$

   Ví dụ:

   [(11 mod 8) – (15 mod 8)] mod 8 = (3 – 7) mod 8 = -4 mod 8 = 4. Ta lại có, *(11 – 15) mod 8 = -4 mod 8 = 4*.

3. $[(a \ mod \ n) \times (b \ mod \ n)] \ mod \ n = (a \times b) \ mod \ n$

   Vi dụ:

   [(11 mod 8)×(15 mod 8)] mod 8 = (3×7) mod 8 = 21 mod 8 = 5. Ta lại có, *(11 x 15) mod 8 = 165 mod 8 = 5*.

Bảng 2.2 sau minh họa phép toán số học cộng, nhân, số đối và số nghịch đảo trên modulo 8. Số đối của số nguyên x là số nguyên y sao cho *(x + y) mod 8 = 0*, số nghịch đảo của số nguyên x là số nguyên y sao cho *(x.y) mod 8 = 1*. Như vậy, để tìm các số đối của các số nguyên ở cột bên trái bảng cộng bằng cách quét lần lượt các phần từ trong dòng để tìm phần tử 0 và khi đó phần tử ở dòng đầu tiên ứng với cột này là số đối. Tương tự, để tìm giá trị nghịch đảo của số bên trái của bảng nhân bằng cách quét lần lượt các phần tử trong dòng để tìm phần tử 1 và khi đó phần tử đầu tiên ứng với cột này là số nghịch đảo.

Bảng 2. 2 Minh họa các phép toán số học trên modulo 8

+	0	1	2	3	4	5	6	7
0	0	1	2	3	4	5	6	7
1	1	2	3	4	5	6	7	0
2	2	3	4	5	6	7	0	1
3	3	4	5	6	7	0	1	2
4	4	5	6	7	0	1	2	3
5	5	6	7	0	1	2	3	4
6	6	7	0	1	2	3	4	5
7	7	0	1	2	3	4	5	6

x	0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7
2	0	2	4	6	0	2	4	6
3	0	3	6	1	4	7	2	5
4	0	4	0	4	0	4	0	4
5	0	5	2	7	4	1	6	3
6	0	6	4	2	0	6	4	2
7	0	7	6	5	4	3	2	1

x	$-x$	x^{-1}
0	0	-
1	7	1
2	6	-
3	5	3
4	4	-
5	3	5
6	2	-
7	1	7

2.1.5 Thuộc tính của số học modulo

Ta định nghĩa Z_n là tập các số nguyên không âm nhỏ hơn n. Như vậy, $Z_n = \{0, 1, 2, \dots, n-1\}$ được gọi là tập các số đồng dư hoặc lớp đồng dư modulo n. Mỗi một số nguyên trong tập Z_n đại diện cho một lớp đồng dư. Ta có thể đặt tên cho các lớp đồng dư modulo n là *[0], [1], [2], …, [n-1]*, trong đó:

$$[r] = \{a: a \text{ là số nguyên sao cho } a \equiv r \ (mod\ n)\}$$

Ví dụ các lớp đồng dư của modulo 3 như sau:

[0] = {…, -12, -9, -3, **0**, 3, 9, 12, …}

[1] = {…, -11, -8, -2, **1**, 4, 7, 10, …}

[2] = {…, -10, -7, **2**,5, 8, 11, 14, …}

Ứng với mỗi lớp đồng dư ta tìm số nguyên không âm nhỏ nhất chính là đại diện của lớp. Như vậy, Z_n chính là tập hợp đại diện của các lớp đồng dư modulo n. Ví dụ $Z_3 = \{0, 1, 2\}$, các phần tử này là đại diện của lớp đồng dư modulo 3: [0], [1] và [2].

Nếu ta thực hiện phép toán số học modulo trên Z_n thì các tính chất trong bảng 2.3 được thỏa mãn với các số nguyên trong Z_n.

Bảng 2. 3 Tính chất của các phép toán số học modulo trên Zn

Tính chất	***Biểu thức***
Giao hoán	$(x + y)\ mod\ n = (y + x)\ mod\ n$ $(x \times y)\ mod\ n = (y \times x)\ mod\ n$
Kết hợp	$[(x + y) + z]\ mod\ n = [x + (y + z)]\ mod\ n$ $[(x \times y) \times z]\ mod\ n = [x \times (y \times z)]\ mod\ n$
Phân phối	$[x \times (y + z)]\ mod\ n = [(x \times y) + (x \times z)]\ mod\ n$
Số đối (-x)	Với mỗi số nguyên $x \in Z_n$ tồn tại số y sao cho $x + y \equiv 0\ (mod\ n)$
Identities	$(0 + x)\ mod\ n = x\ mod\ n$ $(1 \times x)\ mod\ n = x\ mod\ n$

Ngoài ra, một số tính chất của số học đồng dự được dựa vào các tính chất của số học thông thường như sau:

- Nếu $(a + b) \equiv (a + c)\ (mod\ n)\ thì\ b \equiv c\ (mod\ n)$
  Ví dụ, $(5 + 23) \equiv (5 + 7)\ (mod\ 8) \rightarrow 23 \equiv 7\ (mod\ 8)$
- Nếu $(a \times b) \equiv (a \times c)\ (mod\ n)$ thì $b \equiv c\ (mod\ n)$ khi và chỉ khi a và n là 2 số nguyên tố cùng nhau. Hai số là nguyên tố cùng nhau nếu chúng chỉ có một ước số chung dương duy nhất là 1. Ví dụ 5 và 8 là 2 số nguyên tố cùng nhau vì chúng có ước số chung dương duy nhất là 1.

2.1.6 Ước số chung lớn nhất

Ước số chung lớn nhất của 2 số nguyên *a* và *b* là số nguyên dương lớn nhất vừa là ước của a và của b, được ký hiệu là *gcd(a,b)*. Biểu diễn tương đương của ước số chung lớn nhất như sau: $\gcd(a, b) = max[k, sao\ cho\ k|a\ và\ k|b]$.

Ví dụ *gcd(24,18) = 6* bởi vì:

Các ước số của 24 là: 1, 2, 3, 4, 6, 8, 12, 24

Các ước số của 18 là: 1, 2, 3, 6, 18.

Như vậy, ước số chung của 24 và 18 là: 1, 2, 3, 6. Do đó, *gcd(24,18) = max{1, 2, 3, 6} = 6*.

Bởi vì 0 chia hết cho bất kỳ số nguyên khác 0. Do đó *gcd(a,0) = |a|*.

Hai số a và b được gọi là nguyên tố cùng nhau nếu *gcd(a,b) = 1*. Ví dụ 8 và 15 là 2 số nguyên tố cùng nhau bởi vì các ước số dương của 8 là: 1, 2, 4, 8 và các ước số dương của 15 là: 1, 3, 5, 15. Như vậy, ước số chung lớn nhất của 8 và 15 là 1.

Thuật toán Ơclit tìm ước số chung lớn nhất dựa vào định lý sau: Ứng với số nguyên không âm *a* và số nguyên dương *b* bất kỳ thì:

$$\gcd(a,b) = \gcd(b, a \bmod b) \tag{2.1}$$

Ví dụ: *gcd(55,22) = gcd(22,11) = gcd(11,0) = 11*

Để chứng minh định lý trên, ta đặt *c = gcd(a,b)*. Theo định nghĩa của ước số chung lớn nhất thì *a* và *b* chia hết cho *c*. Với số nguyên dương a và b bất kỳ ta có thể biểu diễn dưới dạng $a = k \times b + r \equiv r \ (mod\ b)$ và $a \bmod b = r$ với *k, r* là các số nguyên. Do đó, $a \bmod b = a - k \times b$. Bởi vì, b chia hết cho *c* nên *k.b* cũng chia hết cho *c*. Ta lại có, *a* chia hết cho *c*. Do đó, *a mod b* cũng chia hết cho *c*. Như vậy, *c* là ước số chung của *b* và *a mod b*. Ngược lại, nếu *c* là ước số chung của *b* và *a mod b* thì *k.b* chia hết cho *c* và do đó (*k.b + a mod b*) cũng chia hết cho *c*, điều này tương đương *a* chia hết cho *c*. Do đó, tập các ước số chung của *a* và *b* bằng với tập các ước số chung của *b* và *a mod b*. Nên, *gcd(a,b) = gcd(b,a mod b)*.

Bằng cách lặp lại việc sử dụng phương trình (2.1) ta có thể tìm được ước số chung lớn nhất.

Ví dụ: Tìm ước số chung lớn nhất của 18 và 12.

gcd(18,12) = gcd(12, 18 mod 12) = gcd(12,6) = gcd(6,12 mod 6) = gcd(6,0) = 6.

Thuật toán Ơclit sử dụng lặp phương trình (2.1) để tìm ước số chung lớn nhất. Thuật toán giả sử *a > b > 0*. Có thể chấp nhận thuật toán chỉ áp dụng đối với số nguyên dương bởi vì *gcd(a,b) = gcd(|a|,|b|)*. Thuật toán Ơclid được thực hiện như sau:

Đầu vào: 2 số nguyên dương *a, b* và *a > b*.

Đầu ra: Ước số chung lớn nhất của *a* và *b*.

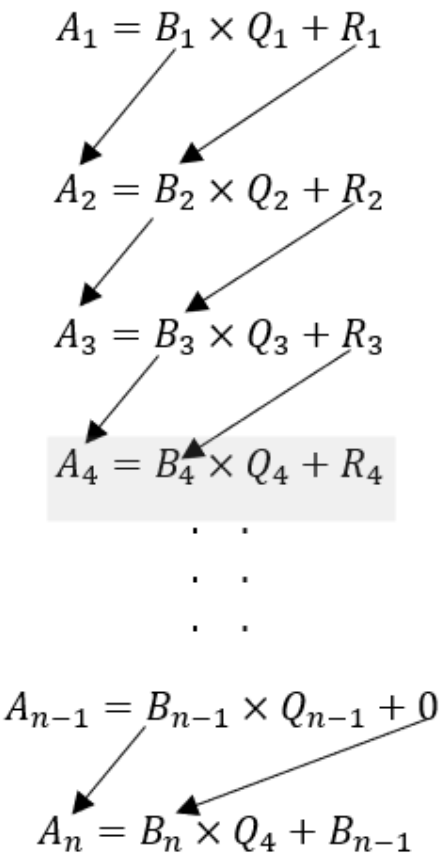
```
EUCLID(a,b)
1. A ← a; B ← b
2. IF B = 0 THEN RETURN A = gcd(a,b)
3. R = A mod B
```

4. A ← b

5. B ← R

6. Quay lại bước 2

Minh họa tiến trình thực hiện của thuật toán như sau:



Ví dụ minh họa thuật toán Ơclit để tìm ước số chung lớn nhất của 1970 và 1066.

Bảng 2. 4 Minh họa các bước tìm gcd(1970,1066)

Tìm ước số chung lớn nhất của 1970 và 1066		
1970	*1066 x 1 + 904*	*gcd(1066,904)*
1066	*904 x 1 + 162*	*gcd(904,162)*
904	*162 x 5 + 94*	*gcd(162,94)*
162	*94 x 1 + 68*	*gcd(94,68)*
94	*68 x 1 + 26*	*gcd(68,26)*
68	*26 x 2 + 16*	*gcd(26,16)*
26	*16 x 1 + 10*	*gcd(16,10)*

16	*10 x 1 + 6*	*gcd(10,6)*
10	*6 x 1 + 4*	*gcd(6,4)*
6	*4 x 1 + 2*	*gcd(4,2)*
4	*2 x 2 + 0*	*gcd(2,0)*
Do đó, *gcd(1970,1066) = 2*		

Đoạn chương trình sau minh họa cài đặt thuật toán Ơclit để tìm ước số chung lớn nhất bằng ngôn ngữ lập trình Java.

```
int euclid(int a, int b){
 int r;
 while(true){
      if(b==0) return a;
      r = a%b;
      a = b;
      b = r;
 }
}
```

2.2 Một số thuật toán trên Z_n

2.2.1 Tìm phần tử nghịch đảo

Phần tử nghịch đảo của số nguyên $a \in Z_n$ là số nguyên $x \in Z_n$ sao cho:

$a \times x \equiv 1(mod\ n)$. Nếu tồn tại *x* thì nó là duy nhất và *a* được gọi là khả nghịch. Phần tử nghịch đảo của *a* ký hiệu là a^{-1}. Để tìm phần tử nghịch đảo của *a* với *n* nhỏ thì ta có thể sử dụng bảng nhân để tìm trực tiếp. Tuy nhiên, với *n* lớn thì phương pháp này không khả thi.

Nếu *gcd(a,n) = 1* thì *a* là khả nghịch modulo *n* có nghĩa là $a \times a^{-1} \equiv 1(mod\ n)$. Như vậy, ta có thể mở rộng thuật toán Ơclit để tìm ước số chung lớn nhất của *a* và *n*. Nếu *gcd(a,n) = 1* thì thuật toán sẽ trả về phần tử nghịch đảo của *a*. Thuật toán Ơclit mở rộng được thực hiện như sau:

```
EXTENDED EUCLID(n,a)
```

1. $(A_1,\ A_2,\ A_3) \leftarrow (1,0,n); (B_1, B_2, B_3) \leftarrow (0,1,a)$
2. IF B_3 = 0 THEN RETURN A_3 = gcd(n,a); không có phần tử nghịch đảo

3. IF B_3 = 1 THEN RETURN B_3 = gcd(n,a); $B_2 = a^{-1} \bmod n$

4. Q = Phần nguyên của phép chia A_3 cho B_3

5. $(R_1, R_2, R_3) \leftarrow (A_1 - Q \times B_1, A_2 - Q \times B_2, A_3 - Q \times B_3)$

6. $(A_1, A_2, A_3) \leftarrow (B_1, B_2, B_3)$

7. $(B_1, B_2, B_3) \leftarrow (R_1, R_2, R_3)$

8. Quay lại bước 2

Từ thuật toán Ơclit và Ơclit mở rộng ta thấy nếu thay thế *A* và *B* trong thuật toán Ơclit bằng A_3 và B_3 trong thuật toán Ơclit mở rộng thì cả 2 thuật toán xử lý 2 biến này giống nhau. Tại mỗi lần lặp của thuật toán Ơclit thì *A* được thay thế bằng giá trị cũ của *B* và *B* được thay bằng phần dữ của phép chia giá trị cũ của *A* cho *B (A mod B)*. Tương tự, tại mỗi lần lặp của thuật toán Ơclit mở rộng A_3 được thay bằng giá trị cũ của B_3 và B_3 được thay thế bằng giá trị cũ của A_3 trừ đi phần nguyên của phép chia A_3 cho B_3 nhân với B_3, đấy chính là phần dư của phép chia A_3 cho B_3 (A_3 *mod* B_3).

Từ các bước của thuật toán Ơclit mở rộng ta thấy nếu *gcd(n,a) = 1* thì tại bước cuối cùng $B_3 = 0$ và $A_3 = 1$. Do đó, bước ngay trước khi kết thúc thì $B_3 = 1$. Vì $n.B_1 + a.B_2 = B_3$ nên $n.B_1 + a.B_2 = 1$ → $a.B_2 = 1 - n.B_1$ → $a.B_2 \equiv 1 (mod\ n)$ → B_2 là nghịch đảo của *a modulo n*.

Bảng 2.5 minh họa các bước thực hiện của thuật toán. Từ bảng ta thấy *gcd(1759,550) = 1* và nghịch đảo của 550 là 355, tức là $550 \times 335 \equiv 1\ (mod\ 1759)$.

Bảng 2. 5 Minh họa tìm nghịch đảo của a = 550 với n = 1759

Q	A_1	A_2	A_3	B_1	B_2	B_3
	1	0	1759	0	1	550
3	0	1	550	1	-3	109
5	1	-3	109	-5	16	5
21	-5	16	5	106	-339	4
1	106	-339	4	-111	355	**1**

Đoạn chương trình sau minh họa cài đặt thuật toán Ơclit mở rộng để tìm phần tử nghịch đảo bằng ngôn ngữ lập trình Java.

```
int extendEuclid(int n, int a){
/* input: n và a
   output: 0 nếu không có phần tử nghịch đảo
           Trái lại, trả về giá trị phần tử nghịch đảo
*/
 int a1 = 1;
 int a2 = 0;
 int a3 = n;
 int b1 = 0;
 int b2 = 1;
 int b3 = a;
 int r1,r2,r3;
 int q;
 while(true){
      if(b3==0) return 0;
      if(b3==1) return b2;
      q = a3/b3;
      r1 = a1-q*b1;
      r2 = a2-q*b2;
      r3 = a3-q*b3;
      a1 = b1;
      a2 = b2;
      a3 = b3;
      b1 = r1;
      b2 = r2;
      b3 = r3;
 }
}
```

2.2.2 Tính a^b mod n

Để tính a^b với *a* và *b* là các số nguyên dương. Nếu ta biểu diễn b thành số nhị phân $b_k b_{k-1} \ldots b_0$ thì $b = \sum_{bi \neq 0} 2^i$ nên ta có:

$$a^b = a^{\sum_{b_i \neq 0} 2^i} = \prod_{b_i \neq 0} a^{2^i}$$

$$a^b mod\ n = \left[\prod_{b_i \neq 0} a^{2^i}\right] mod\ n = \left(\left[\prod_{b_i \neq 0} a^{2^i}\ mod\ n\right]\right) mod\ n$$

Do đó, ta có thể phát triển thuật toán bình phương và nhân để tính giá trị của a^b *mod n* như đoạn mã giả sau đây:

```
MODULE calcExponent(a,b,n)
 Biểu diễn b dưới dạng nhị phân: bkbk-1…b0
 f = 1
 FOR i = k DOWNTO 0 DO
      f  = (f*f) mod n
      IF bi = 1 THEN
           f = (f*a) mod n
      END_IF
 END_FOR
 RETURN f
END MODULE
```

Bảng 2.6 minh họa các bước trong thuật toán bình phương và nhân để tính a^b với *a = 7, b = 560 = 1000110000, n = 561*. Từ bảng ta thu được kết quả 7^{560} *mod 561 = 1*.

Bảng 2. 6 Minh họa các bước thuật toán để tính 7560 mod 561

i	*9*	*8*	*7*	*6*	*5*	*4*	*3*	*2*	*1*	*0*
b_i	*1*	*0*	*0*	*0*	*1*	*1*	*0*	*0*	*0*	*0*
f	*7*	*49*	*157*	*526*	*160*	*241*	*298*	*166*	*67*	***1***

Đoạn chương trình sau cài đặt thuật toán tính a^b mod *n* bằng ngôn ngữ lập trình Java.

```
int calcExponent(int a,int b,int n){
        String bBin = Integer.toBinaryString(b);
        int f = 1;
        for(int i = 0;i<bBin.length();i++){
            f = (f*f)%n;
            if(bBin.charAt(i)=='1'){
                f = (f*a)%n;
            }
        }
        return f;
 }
```

2.3 Giới thiệu về lý thuyết số

2.3.1 Số nguyên tố

Số nguyên $p > 1$ được gọi là số nguyên tố nếu nó chỉ có ước số là ± 1 và $\pm p$. Ví dụ 2 là số nguyên tố vì nó chỉ có các ước số là ± 1 và ± 2. Bảng 2.7 liệt kê các số nguyên tố nhỏ hơn 500.

Bảng 2. 7 Các số nguyên tố nhỏ hơn 500

2	101	211	307	401
3	103	223	311	409
5	107	227	313	419
7	109	229	317	421
11	113	233	331	431
13	127	239	337	433
17	131	241	347	439
19	137	251	349	443
23	149	257	353	449
29	151	263	359	457
31	157	269	367	461
37	163	271	373	463
41	167	277	379	467
43	173	281	383	479
47	179	283	389	487
53	181	293	397	491
59	191			499
61	193			
67	197			
71	199			

73				
79				
83				
89				
97				

2.3.2 Phân tích một số ra thừa số nguyên tố

Phân tích một số ra thừa số nguyên tố tức là viết nó dưới dạng tích lũy thừa của các số nguyên tố. Với mọi số nguyên *a > 1*, ta có thể phân tích nó thành tích sau:

$a = {p_1}^{a_1} \times {p_2}^{a_2} \times \ldots \times {p_k}^{a_k}$

Trong đó, $p_1 < p_2 < \ldots < p_k$ là các số nguyên tố dương.

Ví dụ: $91 = 3 \times 17; \quad 3600 = 2^4 \times 3^2 \times 5^2$

Một cách tổng quát với mọi số nguyên dương *a* đều có thể phân tích duy nhất thành tích sau:

$$a = \prod_{p \in P} p^{a_p}$$

Trong đó, P là tập tất cả các số nguyên tố, $a_p \geq 0$.

Ứng với một số nguyên dương a bất kỳ thì hầu hết các a_p là 0.

Ví dụ: Số nguyên 12 được biểu diễn bởi $\{a_2 = 2, a_3 = 1\}$, tức là $12 = 2^2 \times 3^1$. Số nguyên 91 được biểu diễn bởi $\{a_7 = 2, a_{13} = 1\}$, tức là $91 = 7^2 \times 13^1$.

Ta có thể dễ dàng tìm được ước số chung lớn nhất của 2 số nguyên dương bằng cách phân tích chúng ra thừa số nguyên tố, bởi vì nếu $k = gcd(a,b)$ thì $k_p = min(a_p, b_p)$ với mọi giá trị của *p*.

Ví dụ: Tìm *gcd(300,18)*. Ta có thể phân tích $300 = 2^2 \times 3^1 \times 5^2, 18 = 2^1 \times 3^2$. Do đó, $\gcd(300,18) = 2^1 \times 3^1 \times 5^0 = 6$.

2.3.3 Định lý Fermat

Định lý Fermat phát biểu như sau: Nếu p là số nguyên tố và a là số nguyên dương không chia hết cho p thì:

$$a^{p-1} \equiv 1 (mod\ p)$$

Ví dụ p = 3 là số nguyên tố, a = 5 không chia hết cho 3. Ta có $a^{p-1} mod\ p = 5^2 mod\ 3 = 25\ mod\ 3 = 1$.

Định lý Fermat có thể phát biểu cách khác như sau: Nếu p là số nguyên tố và a là số nguyên dương thì:

$$a^p \equiv a\ (mod\ p)$$

Chú ý, với cách phát biểu thứ nhất thì a và p phải là hai số nguyên tố cùng nhau, còn với cách phát biểu thứ hai thì a và p không cần là hai số nguyên tố cùng nhau.

Ví dụ: p = 5, a = 3 khi đó: $a^p mod\ p = 3^5 mod\ 5 = 243\ mod\ 5 = 3\ mod\ 5 = a\ mod\ p$.

2.3.4 Hàm Ơle

Trước khi trình bày định lý Ơle, ta xét hàm quan trọng của lý thuyết số là hàm Ơle được ký hiệu là $\varphi(n)$ và được định nghĩa là số lượng các số nguyên dương nhỏ hơn *n* và là số nguyên tố cùng nhau với *n*. Theo quy ước thì $\varphi(1) = 1$.

Ví dụ: Xác định $\varphi(37)$ và $\varphi(35)$

Vì 37 là số nguyên tố nên tất cả các số từ 1 đến 36 là số nguyên tố cùng nhau với 37. Do đó, $\varphi(37) = 36$.

Để tìm $\varphi(35)$, ta liệt kê tất cả các số nguyên dương nhỏ hơn 35 và là nguyên tố cùng nhau với 35. Danh sách các số này bao gồm: 1, 2, 3, 4, 6, 8, 9, 11, 12, 13, 16, 17, 18, 19, 22, 23, 24, 26, 27, 29, 31, 32, 33, 34. Danh sách này có 24 số. Do đó, $\varphi(35) = 24$.

Nếu *p* là số nguyên tố thì $\varphi(p) = p - 1$. Ngoài ra, nếu ta có hai số nguyên tố p, q và $p \neq q$ khi đó với *n = p.q* thì $\varphi(n) = \varphi(p.q) = \varphi(p).\varphi(q) = (p-1).(q-1)$.

Ví dụ tìm $\varphi(21) = \varphi(3.7) = \varphi(3).\varphi(7) = (3-1).(7-1) = 12$. Trong đó 12 số nguyên tố cùng 21 là {*1, 2, 4, 5, 8, 10, 11, 13, 16, 17, 19, 20*}.

2.3.5 Định lý Ơle

Định lý Ơle phát biểu như sau: Với bất kỳ cặp số *a* và *n* nguyên tố cùng nhau nào thì $a^{\varphi(n)} \equiv 1(mod\ n)$.

Ví dụ với *a = 3; n = 10*, khi đó $\varphi(n) = \varphi(10) = \varphi(2 \times 5) = \varphi(2) \times \varphi(5) = 1 \times 4 = 4$. Nên $a^{\varphi(n)} = 3^4 = 81 \equiv 1\ (mod\ 10) = 1\ (mod\ n)$.

Cách phát biểu khác của định lý Ơle là $a^{\varphi(n)+1} \equiv a(mod\ n)$

Tương tự như định lý Fermat, dạng thứ nhất của định lý Ơle yêu cầu *a* và *n* phải là số nguyên tố cùng nhau, còn dạng thứ 2 của định lý Ơle thì không cần yêu cầu này.

2.3.6 Kiểm tra tính nguyên tố

Trong một số thuật toán mã hóa ta cần chọn một hoặc một số nguyên tố rất lớn một cách ngẫu nhiên. Như vậy, ta cần phải giải quyết bài toán xác định liệu xem một số lớn có phải là số nguyên tố hay không. Như vậy, việc áp dụng thuật toán kiểm tra một số nhỏ là nguyên tố hay không để kiểm tra một số rất lớn là không phù hợp.

Để kiểm tra một số lớn có phải là nguyên tố hay không ta dựa vào hai tính chất của số nguyên tố.

- Tính chất thứ nhất: Nếu *p* là số nguyên tố và *a* là một số nguyên dương nhỏ hơn *p* thì $a^2 \bmod p = 1$ khi và chỉ khi $a \bmod p = 1$ hoặc $a \bmod p = p - 1$. Thật vậy, vì theo tính chất số học của modulo $(a \bmod p) \times (a \bmod p) = a^2 \bmod p$. Do đó, nêu $a \bmod p = 1$ thì $a^2 \bmod p = 1$ và $a \bmod p = p - 1$ thì $a^2 \bmod p = (p-1)^2 \bmod p = (p^2 - 2.p + 1) \bmod p = 1$.
- Tính chất thứ hai: Nếu p là số nguyên tố lớn hơn 2 thì ta có thể phân tích $p - 1 = 2^k \times q$, với k > 0 và q là số lẻ. Gọi a là một số nguyên bất kỳ trong phạm vi 1 < a < p – 1. Khi đó, một trong hai điều kiện sau đây được thỏa mãn:
    1. $a^q \equiv 1 (mod\ p)$
    2. Một trong các số $a^q, a^{2.q}, a^{4.q}, \dots, a^{2^{(k-1)}.q}$ đồng dư với *1 modulo p*

Thuật toán Miller-Rabin thường được sử dụng để kiểm tra một số lớn có phải là số nguyên tố hay không. Thuật toán dựa vào kết luận rằng nếu *n* là số nguyên tố thì hoặc là phần dư đầu tiên trong danh sách $\left(a^q, a^{2.q}, \dots, a^{2^{k-1}.q}, a^{2^k.q}\right) mod\ n = 1$ hoặc một số phần tử trong danh sách bằng n – 1; nếu không thi n không phải là số nguyên tố (hợp số). Nói cách khác, nếu điều kiện được thỏa mãn thì không chắc chắn n là số nguyên tố. Thật vậy, nếu $n = 2047 = 23 \times 89$, nên $n - 1 = 2046 = 2 \times 1023$. Do đó, $k = 1$, $q = 1023$ và với $a = 2$ thì $2^{1023} \bmod 2047 = 1$ thỏa mãn điều kiện nhưng 2047 không phải là số nguyên tố.

Thuật toán Miller-Rabin nhận tham số đầu vào là một số nguyên và trả về kết quả số cần kiểm tra là hợp số (không phải số nguyên tố) hoặc có thể là số nguyên tố.

```
TEST (n)
Tìm số nguyên k, q với k > 0 và q là số lẻ sao cho n − 1 = 2^k × q;
Chọn một số nguyên ngẫu nhiên a, 1 < a < n − 1;
if a^q mod n = 1 then return "n có thể là số nguyên tố";
for j = 0 to k − 1 do
if  a^(2^j.q) mod n = n − 1 then return "n có thể là số nguyên tố";
return "n không phải là số nguyên tố";
```

Ví dụ áp dụng thuật toán để kiểm tra số nguyên tố $n = 29$. Ta có, $n - 1 = 28 = 2^2 \times 7 = 2^k \times q$. Đầu tiên ta hãy thử với $a = 10$. Ta tính $a^q \bmod n = 10^7 mod\ 29 =$

17. Giá trị này không trùng với *1* và *28* nên ta tiếp tục tính $(10^7)^2 mod\ 29 = 28$. Như vậy, thuật toán sẽ trả về kết quả 29 có thể là số nguyên tố.

2.3.7 Định lý phần dư trung hoa

Trong nhiều trường hợp ta muốn tìm cách để tăng tốc độ tính toán Modulo. Các phép toán trên modulo các số nhỏ tính nhanh nhiều so với các số lớn. Chính vì vậy nếu số lớn phân tích được thành tích của các số nhỏ, từng cặp là nguyên tố cùng nhau. Giả sử ta cần tính *A mod M*, trong đó *M* là một số lớn và có thể phân tích thành tích các số nhỏ như công thức sau:

$$M = \prod_{i=1}^{k} m_i$$

Trong đó m_i là cặp các số nguyên tố cùng nhau, tức là $gcd(m_i,m_j) = 1$, với mọi $i \neq j$ và $1 \leq i,j \leq k$. Ta có thể biểu diễn số nguyên *A* bất kỳ trong Z_M bởi một bộ *k* thành phần và các thành phần nằm trong Z_{m_i} như sau:

$$A \leftrightarrow (a_1, a_2, a_3, \dots, a_k)$$

Trong đó, $A \in Z_M, a_i \in Z_{m_i}$ và $a_i = A\ mod\ m_i$ với $1 \leq i \leq k$.

Đặt $M_i = \frac{M}{m_i}$ và $c_i = M_i \times (M_i^{-1}\ mod\ m_i)$ với $1 \leq i \leq k$.

Định lý phần dư trung hoa xác định $A\ mod\ M = \left(\sum_{i=1}^{k} a_i \times c_i\right)\ mod\ M$.

Định lý phần dư trung hoa áp dụng cho các phép toán số học. Nếu ta có:

$$A \leftrightarrow (a_1, a_2, a_3, \dots, a_k)$$

$$B \leftrightarrow (b_1, b_2, b_3, \dots, b_k)$$

$$(A + B)\ mod\ M \leftrightarrow \left((a_1 + b_1)\ mod\ m_1, (a_2 + b_2)\ mod\ m_2, \dots, (a_k + b_k)\ mod\ m_k\right)$$

$$(A - B)\ mod\ M \leftrightarrow \left((a_1 - b_1)\ mod\ m_1, (a_2 - b_2)\ mod\ m_2, \dots, (a_k - b_k)\ mod\ m_k\right)$$

$$(A \times B)\ mod\ M \leftrightarrow \left((a_1 \times b_1)\ mod\ m_1, (a_2 \times b_2)\ mod\ m_2, \dots, (a_k \times b_k)\ mod\ m_k\right)$$

Ví dụ áp dụng định lý phần dư trung hoa để tính 973 mod 1813. Như vậy, A = 973, $M = 1813 = 37 \times 49 = m_1 \times m_2$. Nên $m_1 = 37$ và $m_2 = 49$.

Ta tính $M_1 = \frac{M}{m_1} = \frac{1813}{37} = m_2 = 49$, $M_2 = \frac{M}{m_2} = \frac{1813}{49} = 37$.

Sử dụng thuật toán Ơclit mở rộng, ta tính được $M_1^{-1} = 34\ mod\ m_1$ và $M_2^{-1} = 4\ mod\ m_2$. Ta cũng tính được, $a_1 = A\ mod\ m_1 = 973\ mod\ 37 = 11$ và $a_2 = A\ mod\ m_2 = 973\ mod\ 49 = 42$, $c_1 = 49 \times 34, c_2 = 37 \times 4$. Như vậy, *A* được biểu diễn bởi cặp (11, 42) và $A\ mod\ M = [a_1 \times c_1 + a_2 \times c_2]\ mod\ M = (11 \times 49 \times 34 + 42 \times 37 \times 4) mod\ M = 24542 \bmod 1813 = 973$.

Giả sử ta cần tính *(A+B) mod M* với *B* = 678. Khi đó *B* được biểu diễn bởi cặp *(b_1, b_2) = (678 mod 37, 678 mod 49) = (12, 41)*. Do đó, *(A+B) mod M* được biểu diễn bởi cặp *(($a_1 + b_1$) mod m_1, ($a_2 + b_2$) mod m_2) = ((11+12) mod 37, (42+41) mod 49) = (23, 34)*. Nên $(\mathrm{A} + \mathrm{B}) \bmod \mathrm{M} = [23 \times 49 \times 34 + 34 \times 37 \times 4] \, mod \, 1813 = 43350 \, mod \, 1813 = 1651.$

Áp dụng định lý phần dư trung hoa để giải hệ phương trình modulo. Cho $a_i = x \, mod \, m_i$ với $\forall \, i, j, \gcd(m_i, m_j) = 1$.

Ví du giải hệ phương trình modulo sau: $x \equiv 3 \, mod \, 7$ và $x \equiv 6 \, mod \, 11$. Từ hệ phương trình trên ta có $a_1 = x \, mod \, 7 = 3, a_2 = x \, mod \, 11 = 6, M = m_1 \times m_2 = 77, m_1 = 7, m_2 = 11, M_1 = \frac{M}{m_1} = \frac{77}{7} = 11, M_2 = \frac{M}{m_2} = \frac{77}{11} = 7, M_1^{-1} \, mod \, m_1 = 11^{-1} mod \, 7 = 2$ và $M_2^{-1} mod \, m_2 = 7^{-1} \, mod \, 11 = 8, c_1 = M_1 \times M_1^{-1} mod \, m_1 = 11 \times 2 = 22, c_2 = M_2 \times M_2^{-1} mod \, m_2 = 7 \times 8 = 56.$ Áp dụng định lý phần dư trung hoa ta có: $x = (a_1 \times c_1 + a_2 \times c_2) \, mod \, M = (3 \times 22 + 6 \times 56) \, mod \, 77 = 402 \, mod \, 77 = 17.$

Trong chương này giáo trình đã trình bày các kiến thức căn bản của toán học được sử dụng trong mã hóa dữ liệu như số học đồng dư và lý thuyết số. Chương tiếp theo giáo trình sẽ trình bày một số loại mã cổ điển thông dụng như mã Ceasar, mã Playfair, mã Vigenere, …

Câu hỏi và bài tập

Bài 1. Hãy thực hiện các phép toán số học modulo sau:

a) 5 mod 12 + 9 mod 12

b) 5 mod 12 – 9 mod 12

c) $5 \, mod \, 12 \times 9 \, mod \, 12$

d) $5^{17} mod \, 7$

Bài 2. Hãy minh họa các bước tìm ước số chung lớn nhất theo thuật toán Ơclit

a) gcd(24140,16762)

b) gcd(4655,12075)

Bài 3. Tìm giá trị của x sao cho

a) $5x \equiv 4 \, (mod \, 3)$

b) $7x \equiv 6 \, (mod \, 5)$

c) $9x \equiv 8 \, (mod \, 7)$

Bài 4. Áp dụng thuật toán Ơclit mở rộng để tìm số nghịch đảo sau:

a) 1234 mod 4321

b) 24140 mod 40902

c) 550 mod 1769

Bài 5. Tìm giá trị của hàm Ơle của các số nguyên sau:

a) $\varphi(19)$

b) $\varphi(27)$

c) $\varphi(231)$

d) $\varphi(440)$

Bài 6. Áp dụng định lý phần dư trung hoa để tính các biểu thức sau:

a) 120 mod (11*23)

b) 15^{30} mod (7*9)

Bài 7. Áp dụng định lý phần dư trung hoa để tìm giá trị x

a) x mod 13 = 5 và x mod 7 = 4

b) $x \equiv 2\ (mod\ 3); x \equiv 3\ (mod\ 5); x \equiv 2\ (mod\ 7)$

Bài 8. Viết chương trình để cài đặt thuật toán tính nhanh số mũ.

Bài 9. Viết chương trình để kiểm tra một số n có phải là nguyên tố hay không áp dụng thuật toán Miller-Rabin.

CHƯƠNG 3. MÃ CỔ ĐIỂN

Chương này tập trung trình bày các khái niệm về mã hóa, giải mã và các hệ mã cổ điển. Để đáp ứng mục đích trên nội dung của chương tập chung trình bày các phương pháp mã hóa cổ điển như Ceasar, Vigenere, Affine, Hill.... Cài đặt và thử nghiệm các giải thuật mã hóa và giải mã đó.

3.1 Mã đối xứng

3.1.1 Các khái niệm cơ bản

Mật mã đối xứng sử dụng cùng một khóa cho việc mã hóa và giải mã. Ở đây người gửi và người nhận chia sẻ khoá chung K, mà họ có thể trao đổi bí mật với nhau. Ta xét hai hàm ngược nhau: E là hàm biến đổi bản rõ thành bản mã và D là hàm biến đổi bản mã trở về bản rõ. Giả sử X là văn bản cần mã hóa và Y là dạng văn bản đã được thay đổi qua việc mã hóa. Khi đó ta ký hiệu:

$Y = E_K(X)$

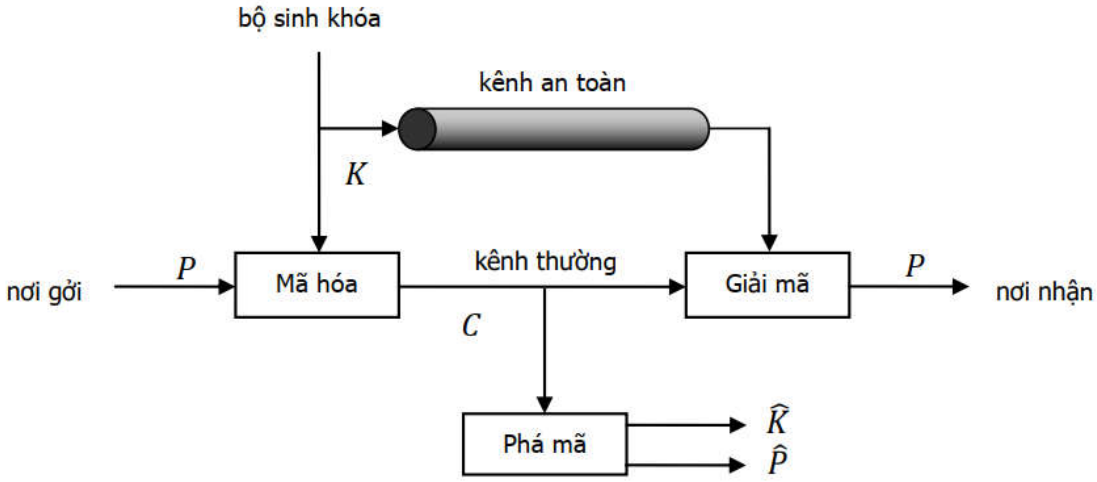
$X = D_K(Y)$

Các thuật toán mã cổ điển đều là mã đối xứng. Mã đối xứng là kiểu duy nhất trước khi phát minh ra khoá mã công khai (còn được gọi là mã không đối xứng) vào những năm 1970. Hiện nay các mã đối xứng và công khai tiếp tục phát triển và hoàn thiện. Mã công khai ra đời hỗ trợ mã đối xứng chứ không thay thế nó, do đó mã đối xứng đến nay vẫn được sử dụng rộng rãi.

Sau đây ta đưa ra định nghĩa một số khái niệm cơ bản về mã hóa.

- **Bản rõ X:** được gọi là là bản tin gốc. Bản rõ có thể được chia nhỏ có kích thước phù hợp.
- **Bản mã Y:** là bản tin gốc đã được mã hoá. Ở đây ta thường xét phương pháp mã hóa mà không làm thay đổi kích thước của bản rõ, tức là chúng có cùng độ dài.
- **Mã:** là thuật toán E chuyển bản rõ thành bản mã. Thông thường chúng ta cần thuật toán mã hóa mạnh, cho dù kẻ thù biết được thuật toán, nhưng không biết thông tin về khóa cũng không tìm được bản rõ.
- **Khoá K**: là thông tin tham số dùng để mã hoá, chỉ có người gửi và người nhận biết. Khóa là độc lập với bản rõ và có độ dài phù hợp với yêu cầu bảo mật.
- **Mã hoá**: là quá trình chuyển bản rõ thành bản mã, thông thường bao gồm việc áp dụng thuật toán mã hóa và một số quá trình xử lý thông tin kèm theo.
- **Giải mã:** là quá trình chuyển bản mã thành bản rõ, đây là quá trình ngược lại của mã hóa.

- **Mật mã**: là chuyên ngành khoa học của Khoa học máy tính nghiên cứu về các nguyên lý và phương pháp mã hoá. Hiện nay người ta đưa ra nhiều chuẩn an toàn cho các lĩnh vực khác nhau của công nghệ thông tin.
- **Thám mã**: nghiên cứu các nguyên lý và phương pháp giải mã mà không biết khoá. Thông thường khi đưa các mã mạnh ra làm chuẩn dùng chung giữa các người sử dụng, các mã đó được các kẻ thám mã cũng như những người phát triển mã tìm hiểu nghiên cứu các phương pháp giải một phần bản mã với các thông tin không đầy đủ.
- **Lý thuyết mã**: bao gồm cả mật mã và thám mã. Nó là một thể thống nhất, để đánh giá một mã mạnh hay không, đều phải xét từ cả hai khía cạnh đó. Các nhà khoa học mong muốn tìm ra các mô hình mã hóa khái quát cao đáp ứng nhiều chính sách an toàn khác nhau.



Hình 3. 1 Mô hình mã hóa đối xứng

Đặc tính quan trọng của mã hóa đối xứng là khóa phải được giữ bí mật giữa người gửi và người nhận, hay nói cách khác khóa phải được chuyển một cách an toàn từ người gửi đến người nhận. Có thể đặt ra câu hỏi là nếu đã có một kênh an toàn để chuyển khóa như vậy thì tại sao không dùng kênh đó để chuyển bản tin, tại sao cần đến chuyện mã hóa? Câu trả lời là nội dung bản tin thì có thể rất dài, còn khóa thì thường là ngắn. Ngoài ra một khóa còn có thể áp dụng để truyền tin nhiều lần. Do đó nếu chỉ chuyển khóa trên kênh an toàn thì đỡ tốn kém chi phí.

3.1.2 Các yêu cầu đối với hệ mã đối xứng

Một mã đối xứng có các đặc trưng là cách xử lý thông tin của thuật toán mã, giải mã, tác động của khóa vào bản mã, độ dài của khóa. Mối liên hệ giữa bản rõ, khóa và bản mã càng phức tạp càng tốt, nếu tốc độ tính toán là chấp nhận được. Cụ thể hai yêu cầu để sử dụng an toàn mã khoá đối xứng là

- Thuật toán mã hoá mạnh. Có cơ sở toán học vững chắc đảm bảo rằng mặc dù công khai thuật toán, mọi người đều biết, nhưng việc thám mã là rất khó khăn và phức tạp nếu không biết khóa.
- Khoá mật chỉ có người gửi và người nhận biết. Có kênh an toàn để phân phối khoá giữa các người sử dụng chia sẻ khóa. Mối liên hệ giữa khóa và bản mã là không nhận biết được.

3.2 Các hệ mã thay thế

Mã thay thế là phương pháp mà từng kí tự (nhóm kí tự) trong bản rõ được thay thế bằng một kí tự (một nhóm kí tự) khác để tạo ra bản mã. Bên nhận chỉ cần thay thế ngược lại trên bản mã để có được bản rõ ban đầu.

3.2.1 Mã Ceasar

Đây là mã thế được biết sớm nhất, được sáng tạo bởi Julius Ceasar. Lần đầu tiên được sử dụng trong quân sự. Việc mã hoá được thực hiện đơn giản là thay mỗi chữ trong bản rõ bằng chữ thứ ba tiếp theo trong bảng chữ cái.

Mã dịch vòng (shift cipher): thông điệp được mã hóa bằng cách dịch chuyển xoay vòng từng ký tự đi k vị trí trong bảng chữ cái, mã dịch vòng được xác định trên Z_{26} (do có 26 chữ cái trên bảng chữ cái tiếng Anh) mặc dù có thể xác định nó trên Z_m với m tuỳ ý. Mã dịch vòng sẽ tạo nên một hệ mật như đã xác định ở trên, tức là $d_K(e_K(x)) = x$ với mọi $x \in Z_{26}$. Ta có sơ đồ mã như sau:

$P = C = K = Z_{26}$ với $0 \le k \le 25$

Mã hóa: $e_k(x) = x+k \bmod 26$

Giải mã: $d_k(x) = y-k \bmod 26$ (với $x,y \in Z_{26}$)

Trong trường hợp K = 3, hệ mật thường được gọi là mã Ceasar. Thế kỷ thứ 3 trước công nguyên, nhà quân sự người La Mã Julius Ceasar đã nghĩ ra phương pháp mã hóa một bản tin như sau: thay thế mỗi chữ trong bản tin bằng chữ đứng sau nó k vị trí trong bảng chữ cái. Giả sử chọn k = 3, ta có bảng chuyển đổi như sau:

Chữ ban đầu: a b c d e f g h i j k l m n o p q r s t u v w x y z
Chữ thay thế: D E F G H I J K L M N O P Q R S T U V W X Y Z A B C
(sau Z sẽ vòng lại là A, do đó x $\rightarrow$ A, y $\rightarrow$ B và z $\rightarrow$ C)

Giả sử có bản tin gốc (bản rõ): meet me after the toga party

Như vậy bản tin mã hóa (bản mã) sẽ là: PHHW PH DIWHU WKH WRJD SDUWB

Thay vì gửi trực tiếp bản rõ cho các cấp dưới, Ceasar gửi bản mã. Khi cấp dưới nhận được bản mã, tiến hành giải mã theo quy trình ngược lại để có được bản rõ. Như vậy nếu đối thủ của Ceasar có lấy được bản mã, thì cũng không hiểu được ý nghĩa của bản mã. Chúng ta hãy gán cho mỗi chữ cái một con số nguyên từ 0 đến 25:

Bảng 3. 1 Bảng chữ cái tiếng Anh

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

Mã hóa Ceasar, từ một bản mã có thể dễ dàng suy ra được bản rõ ban đầu mà không cần biết khóa bí mật. Hành động đi tìm bản rõ từ bản mã mà không cần khóa như vậy được gọi là hành động phá mã (cryptanalysis). Do đó một hệ mã hóa đối xứng được gọi là an toàn khi và chỉ khi nó không thể bị phá mã (điều kiện lý tưởng) hoặc thời gian phá mã là bất khả thi.

Trong phương pháp Ceasar, lý do mà phương pháp này kém an toàn là ở chỗ khóa k chỉ có 25 giá trị, do đó kẻ phá mã có thể thử được hết tất cả các trường hợp của khóa rất nhanh chóng. Phương pháp tấn công này được gọi là phương pháp vét cạn khóa (bruteforce attack). Chỉ cần nới rộng miền giá trị của khóa thì có thể tăng thời gian phá mã đến một mức độ được coi là bất khả thi. Bảng dưới đây liệt kê một số ví dụ về thời gian phá mã trung bình tương ứng với kích thước của khóa.

3.2.2 Phương pháp mã hóa VIIGNERE

Mã thay thế đa bảng đơn giản nhất là mã Vigenere. Thực chất quá trình mã hoá Vigenere là việc tiến hành đồng thời dùng nhiều mã Ceasar cùng một lúc trên bản rõ với nhiều khoá khác nhau. Khoá cho mỗi chữ dùng để mã phụ thuộc vào vị trí của chữ đó trong bản rõ và được lấy trong từ khoá theo thứ tự tương ứng.

Giả sử khoá là một chữ có độ dài d được viết dạng $K = k_1k_2\ldots k_m$, trong đó k_i nhận giá trị nguyên từ 0 đến 25. Khi đó ta chia bản rõ thành các khối gồm d chữ. Mỗi chữ thứ i trong khối chỉ định dùng bảng chữ thứ i với tịnh tiến là k_i giống như trong mã Ceasar. Trên thực tế khi mã ta có thể sử dụng lần lượt các bảng chữ và lặp lại từ đầu sau m chữ của bản rõ. Vì có nhiều bảng chữ khác nhau, nên cùng một chữ ở các vị trí khác nhau sẽ có các bước nhảy khác nhau, làm cho tần suất các chữ trong bản mã dãn tương đối đều.

Phương pháp Vigenere dựa trên sơ đồ mã như sau:

Cho m là một số nguyên dương xác định. Định nghĩa $P = C = K = (Z_{26})^m$. Với

khoá $K = (k_1, k_2, \ldots, k_m)$ ta xác định :

Mã hóa: $e_K(x_1, x_2, \ldots, x_m) = (x_1+k_1, x_2+k_2, \ldots, x_m+k_m)$

Giải mã: $d_K(y_1, y_2, \ldots, y_m) = (y_1-k_1, y_2-k_2, \ldots, y_m-k_m)$

trong đó tất cả các phép toán được thực hiện trong Z_{26}

Ví dụ: Giả sử m =6 và từ khoá là CIPHER. Từ khoá này tương ứng với dãy số K =(2,8,15,4,17). Giả sử bản rõ là xâu “Thiscryptosystemisnotsecure”. Ta sẽ biến đổi các

phần tử của bản rõ thành các thặng dư theo modulo 26, viết chúng thành các nhóm 6 rồi cộng với từ khoá theo modulo 26 như sau:

19	7	8	18	2	17
2	8	15	7	4	17
21	**15**	**23**	**25**	**6**	**8**

24	15	19	14	18	24
2	8	15	7	4	17
0	**23**	**8**	**21**	**22**	**15**

18	19	4	12	8	18
2	8	15	7	4	17
20	**1**	**19**	**19**	**12**	**9**

13	14	19	18	4	2
2	8	15	7	4	17
15	**22**	**8**	**25**	**8**	**19**

20	17	4
2	8	15
22	**25**	**19**

Vậy bản mã sẽ là: "vpxzgiaxivwoubttmjpwizitwzt". Để giải mã ta có thể dùng cùng từ khoá nhưng thay cho cộng, ta trừ cho nó theo modulo 26.

Ta thấy rằng với các từ khoá có độ dài m trong mật mã Vigenère là 26^m, bằng phương pháp tìm kiếm vét cạn yêu cầu thời gian khá lớn cho việc tìm kiếm thủ công bằng tay. Ví dụ, nếu m = 5 thì không gian khoá cũng có kích thước lớn hơn $1{,}1 \times 10^7$. Lượng khoá này đã đủ lớn để ngăn ngừa việc tìm khoá bằng tay nhưng ngày nay với thiết bị máy tính thì điều đó là dễ dàng.

3.2.3 Mã Affine

Một trường hợp khác của mã thay thế là mã Affine. Mã Affine được định nghĩa như sau:

P = C = Z_{26}, K = {(a,b) ∈ $Z_{26} \times Z_{26}$, ước chung lớn nhất của a và 26 bằng 1}.

Với mỗi k ∈ K ta có:

Hàm mã hóa $e_k(x)= ax + b \pmod{26}$

Hàm giải mã $d_k(y)= a^{-1} (y-b) \pmod{26}$.

Để việc giải mã có thể thực hiện được, yêu cầu cần thiết là hàm Affine phải là đơn ánh. Nói cách khác, với bất kỳ y ∈ Z_{26}, ta muốn có đồng nhất thức sau $ax + b \equiv y \pmod{26}$ phải có nghiệm x duy nhất. Đồng dư thức này tương đương với $ax \equiv y-b \pmod{26}$. Vì y thay đổi trên Z_{26} nên y-b cũng thay đổi trên Z_{26} . Bởi vậy, ta chỉ cần nghiên cứu

phương trình đồng dư $ax \equiv y \pmod{26}$ ($y \in Z_{26}$). Ta biết rằng, phương trình này có một nghiệm duy nhất đối với mỗi y khi và chỉ khi ước chung lớn nhất của a và 26 bằng 1.

Ví dụ: Giả sử k = (7,3). Ta có $7^{-1} \bmod 26 = 15$.

Hàm mã hoá là: $e_k(x) = 7x+3 \pmod{26}$

Và hàm giải mã tương ứng là: $d_k(x) = 15(y-3) \pmod{26} = 15y - 19 \pmod{26}$

Ở đây, tất cả các phép toán đều thực hiện trên Z_{26}. Ta sẽ kiểm tra liệu $d_k(e_k(x)) = x$ với mọi $x \in Z_{26}$ hay không. Ta thực hiện các tính toán trên Z_{26} , ta có $d_k(e_k(x)) = d_k(7x+3) = 15(7x+3)-19 = x +45 -19 = x$.

Để minh hoạ, ta hãy mã hoá bản rõ "HOT". Trước tiên biến đổi các chữ H, O, T thành các thặng dư theo modulo 26. Ta được các số tương ứng là 7, 14 và 19. Bây giờ sẽ mã hoá:

$7 \times 7 + 3 \bmod 26 = 52 \bmod 26 = 0$

$7 \times 14 + 3 \bmod 26 = 101 \bmod 26 = 23$

$7 \times 19 + 3 \bmod 26 = 136 \bmod 26 = 6$

Bởi vậy 3 ký hiệu của bản mã là 0, 23 và 6 tương ứng với xâu ký tự AXG.

Thám hệ mã Affine: Mật mã Affine là một ví dụ đơn giản cho ta thấy cách thám hệ mã nhờ dùng các số liệu thống kê. Giả sử ta đã thu trộm được bản mã sau:

Bảng 3. 2 Tần suất xuất hiện của 26 chữ cái của bản mã

Kí tự	Tần suất	Kí tự	Tần suất	Kí tự	Tần suất	Kí tự	Tần suất
A	2	H	5	O	1	U	2
B	1	I	0	P	3	V	4
C	0	J	0	Q	0	W	0
D	6	K	5	R	8	X	2
E	5	L	2	S	3	Y	1
F	4	M	2	T	0	Z	0
G	0	N	1				

Bản mã nhận được từ mã Affine:

FMXVEDRAPHFERBNDKRXRSREFMORUDSDKDVSHVUFEDKPK

DLYEVLRHHRH

Phân tích tần suất của bản mã này được cho ở bảng trên. Bản mã chỉ có 57 ký tự. Tuy nhiên độ dài này cũng đủ phân tích thám mã đối với hệ Affine. Các ký tự có tần suất cao nhất trong bản mã là: R (8 lần xuất hiện), D (6 lần xuất hiện), E, H, K (mỗi ký

tự 5 lần) và F, S, V (mỗi ký tự 4 lần). Trong phỏng đoán ban đầu, ta giả thiết rằng R là ký tự mã của chữ e và D là kí tự mã của t, vì e và t tương ứng là 2 chữ cái thông dụng nhất. Biểu thị bằng số ta có: $e_k(4) = 17$ và $e_k(19) = 3$. Nhớ lại rằng $e_k(x) = ax + b$ trong đó a và b là các số chưa biết. Bởi vậy ta có hai phương trình tuyến tính hai ẩn:

$4a + b = 17$

$19a + b = 3$

Hệ này có duy nhất nghiệm a = 6 và b = 19 (trong Z_{26}). Tuy nhiên đây là một khoá không hợp lệ do UCLN(a,26) = 2. Bởi vậy giả thiết của ta là không đúng. Phỏng đoán tiếp theo của ta là: R là ký tự mã của e và E là mã của t. Thực hiện như trên, ta thu được a =13 và đây cũng là một khoá không hợp lệ. Bởi vậy ta phải thử một lần nữa: ta coi rằng R là mã hoá của e và H là mã hoá của t. Điều này dẫn tới a = 8 và đây cũng là một khoá không hợp lệ. Tiếp tục, giả sử rằng R là mã hoá của e và K là mã hoá của t. Theo giả thiết này ta thu được a = 3 và b =5 là khóa hợp lệ.

Ta sẽ tính toán hàm giải mã ứng với k = (3,5) và giải mã bản mã để xem liệu có nhận được xâu tiếng Anh có nghĩa hay không. Điều này sẽ khẳng định tính hợp lệ của khoá (3,5). Khi thực hiện các phép toán này, ta có $d_k(y) = 9y - 19$ và giải mã bản mã đã cho, ta được: algorithmsarequitegeneraldefinitionsof

arithmeticprocesses

Như vậy khoá xác định trên là khoá đúng.

3.2.3 Mật mã Hill

Trong phần này sẽ mô tả một hệ mật thay thế khác được gọi là mật mã Hill. Mật mã Hill do Lester S.Hill đưa ra năm 1929. Giả sử m là một số nguyên dương, đặt $P = C = (Z_{26})^m$. Ý tưởng ở đây là lấy m tổ hợp tuyến tính của m ký tự trong một phần tử của bản rõ để tạo ra m ký tự ở một phần tử của bản mã.

Ví dụ nếu m = 2 ta có thể viết một phần tử của bản rõ là $x = (x_1, x_2)$ và một phần tử của bản mã là $y = (y_1, y_2)$, ở đây, y_1 cũng như y_2 đều là một tổ hợp tuyến tính của x_1 và x_2. Chẳng hạn, có thể lấy

$y_1 = 11x_1 + 3x_2$

$y_2 = 8x_1 + 7x_2$

Tất nhiên có thể viết gọn hơn theo ký hiệu ma trận như sau:

$$(y_1\ y_2) = (x_1\ x_2) \begin{pmatrix} 11 & 8 \\ 3 & 7 \end{pmatrix}$$

Nói chung, có thể lấy một ma trận k kích thước m × m làm khoá. Nếu một phần tử ở hàng i và cột j của k là $k_{i,j}$ thì có thể viết $k = (k_{i,j})$, với $x = (x_1, x_2, \ldots, x_m) \in P$ và $k \in K$, ta tính $y = e_k(x) = (y_1, y_2, \ldots, y_m)$ như sau:

$(y_1,\ldots,y_m)=(x_1,\ldots,x_m)\begin{pmatrix} k1,1 & k1,2\ldots & k1,m \\ \ldots & \ldots\ \ldots & \ldots \\ km,1 & km,2\ldots & km,m \end{pmatrix}$

Nói một cách khác y = xk. Chúng ta nói rằng bản mã nhận được từ bản rõ nhờ phép biến đổi tuyến tính. Ta sẽ xét xem phải thực hiện giải mã như thế nào, tức là làm thế nào để tính x từ y. Bạn đọc đã làm quen với đại số tuyến tính sẽ thấy rằng phải dùng ma trận nghịch đảo K^{-1} để giả mã. Bản mã được giải mã bằng công thức $y K^{-1}$.

Như vậy mật mã Hill được định nghĩa như sau:

Cho m là một số nguyên dương có định. Cho $P = C = (Z_{26})^m$ và cho K = { các ma trận khả nghịch cấp m × m trên Z_{26}}. Với một khoá k ∈K ta xác định

$e_k(x) = xk$

và $d_k(y) = yk^{-1}$

Tất cả các phép toán được thực hiện trong Z_{26}

Ví dụ: Có khóa k=$\begin{pmatrix} 11 & 8 \\ 3 & 7 \end{pmatrix}$ hãy mã hoá bản rõ "JULY" với hệ mã Hill.

Ta có hai phần tử của bản rõ để mã hoá là (9,20) (ứng với "JU") và (11,24) (ứng với "LY"). Ta tính như sau:

Với bản rõ "JU"=(9,20) ta có bản mã là (9,20)$\begin{pmatrix} 11 & 8 \\ 3 & 7 \end{pmatrix}$=(99+60, 72+140) = (3,4)= "DE". Tương tự với bản rõ "LY" ta có bản mã là "LW". Bởi vậy bản mã của "JULY" là "DELW".

Để giải mã ta sẽ tính $k^{-1}= (\det K)^{-1}K^*$. Trong đó K*=$\begin{pmatrix} 7 & -8 \\ -3 & 11 \end{pmatrix} = \begin{pmatrix} 7 & 18 \\ 23 & 11 \end{pmatrix}$ và det K=det$\begin{pmatrix} 11 & 8 \\ 3 & 7 \end{pmatrix}$=11×7-8×3 mod 26=77 - 24 mod 26 = 53 mod 26=1. Vì 1^{-1} mod 26 = 1 nên ma trận nghịch đảo là k^{-1}=$\begin{pmatrix} 7 & 18 \\ 23 & 11 \end{pmatrix}$.

Vậy với bản mã "DE"=(3,4) ta có bản rõ là (3,4) $\begin{pmatrix} 7 & 18 \\ 23 & 11 \end{pmatrix}$=(9,20)= "JU". Với bản mã "LW"=(11,22) ta có bản rõ là (11,22) $\begin{pmatrix} 7 & 18 \\ 23 & 11 \end{pmatrix}$=(11,24)= "LY"

Hệ mã Hill là một hệ mật khó phá hơn nếu tấn công chỉ với bản mã. Tuy nhiên hệ mật này dễ bị phá nếu tấn công bằng bản rõ đã biết. Trước tiên, giả sử rằng, thám mã đã biết được giá trị m đang sử dụng. Giả sử thám mã có ít nhất m cặp véc tơ khác nhau $x_j = (x_{1,j}, x_{2,j}, , \ldots, x_{m,j})$ và $y_j = (y_{1,j}, y_{2,j},\ldots,y_{m,j})$ $(1 \le j \le m)$ sao cho $y_j = e_k(x_j)$, $1 \le j \le m$. Nếu xác định hai ma trận: $x = (x_{i,j})$ $y = (y_{i,j})$ cấp m×m thì ta có phương trình ma trận y = xk, trong đó ma trận k cấp m×m là khoá chưa biết. Với điều kiện ma trận y là khả nghịch. ta có thể tính $k = x^{-1}y$ và nhờ vậy phá được hệ mật. Nếu y không khả nghịch thì cần phải thử các tập khác gồm m cặp bản rõ và bản mã.

3.3 Các hệ mã hoán vị

Trong phương pháp mã hoán vị, các kí tự trong bản rõ vẫn được giữ nguyên, chúng chỉ được sắp xếp lại vị trí để tạo ra bản mã. Tức là các kí tự trong bản rõ hoàn toàn không bị thay đổi bằng kí tự khác mà chỉ đảo chỗ của chúng để tạo thành bản mã.

3.3.1 Mã Rail Fence

Đây là mã hoán vị đơn giản. Viết các chữ của bản rõ theo đường chéo trên một số dòng. Sau đó đọc các chữ theo theo từng dòng sẽ nhận được bản mã. Số dòng chính là khoá của mã. Vì khi biết số dòng ta sẽ tính được số chữ trên mỗi dòng và lại viết bản mã theo các dòng sau đó lấy bản rõ bằng cách viết lại theo các cột.

Ví dụ: Viết bản tin "meet me after the toga party" lần lượt trên hai dòng như sau

```
m e m a t r h t g p r y
e t e f e t e o a a t
```

Sau đó ghép các chữ ở dòng thứ nhất với các chữ ở dòng thứ hai cho bản mã:

MEMATRHTGPRYETEFETEOAAT

3.3.2 Mã dịch chuyển dòng

Mã có sơ đồ phức tạp hơn. Viết các chữ của bản tin theo các dòng với số cột xác định. Sau đó thay đổi thứ tự các cột theo một dãy số khóa cho trước, rồi đọc lại chúng theo các cột để nhận được bản mã. Quá trình giải mã được thực hiện ngược lại.

Ví dụ:

```
Key:       4 3 1 2 5 6 7
Plaintext: a t t a c k p
           o s t p o n e
           d u n t i l t
           w o a m x y z
```

Ta đọc theo thứ tự các cột từ 1 đến 7 để nhận được bản mã:

Ciphertext: TTNAAPTMTSUOAODWCOIXKNLYPETZ

3.3.3 Mã tích

Mã dùng hoán vị hoặc dịch chuyển không an toàn vì các đặc trưng tần suất của ngôn ngữ không thay đổi. Có thể sử dụng một số mã liên tiếp nhau sẽ làm cho mã khó hơn. Mã cổ điển chỉ sử dụng một trong hai phương pháp thay thế hoặc hoán vị. Người ta nghĩ đến việc kết hợp cả hai phương pháp này trong cùng một mã và có thể sử dụng đan xen hoặc lặp nhiều vòng. Đôi khi ta tưởng lặp nhiều lần cùng một loại mã sẽ tạo nên mã phức tạp hơn, nhưng trên thực tế trong một số trường hợp về bản chất chúng cũng tương đương với một lần mã cùng loại nào đó như: tích của hai phép thế sẽ là một phép thế; tích của hai phép hoán vị sẽ là một phép hoán vị. Nhưng nếu hai loại mã đó khác nhau thì sẽ tạo nên mã mới phức tạp hơn, chính vì vậy phép thế được nối tiếp bằng

phép dịch chuyển sẽ tạo nên mã mới khó hơn rất nhiều. Đây chính là chiếc cầu nối từ mã cổ điển sang mã hiện đại.

Trong chương này giáo trình đã trình bày một số loại mã cổ điển thông dụng như mã Ceasar, mã Afine, mã Playfair, mã Vigenere, mã Hill. Chương tiếp theo giáo trình sẽ trình bày về chuẩn mã dữ liệu DES và chuẩn mã nâng cao AES.

Câu hỏi và bài tập

Bài 1. Cho đoạn bản mã "GCUA VQ DTGCM". Hãy dùng mã Ceasar suy luận để tìm bản rõ (sử dụng bảng chữ cái tiếng Anh).

Bài 2. Mã hóa bản rõ “Chung toi se la nhung ky su cong nghe thong tin gioi trong mot vai nam nua” sử dụng từ khóa 631425 bằng phương pháp Vigenere.

Bài 3. Cho hệ mã Vigenere có M = 6, K = “CIPHER”.

a) Hãy thực hiện mã hóa xâu P = “THIS IS MY TEST“.

b) Hãy thực hiện giải mã xâu M = “EICJIC RTPUEI GBGLEK CBDUGV”.

Bài 4. Cho hệ mã Vigenere có M = 6. Mã hóa xâu P = “THIS IS MY TEST“ người ta thu được bản mã là “LLKJML ECVVWM”.

a) Hãy tìm khóa mã hóa đã dùng của hệ mã trên.

b) Dùng khóa tìm được ở phần trên hãy giải mã bản mã C = “KLGZWT OMBRVW”.

Bài 5. Cho hệ mã Affine được cài đặt trên Z_{99}. Khi đó khóa là các cặp (a, b) trong đó a, b $\in Z_{99}$ với ước số chung lớn nhất của (a, 99) = 1. Hàm mã hóa $e_k(x) = (a * x + b) \bmod 99$ và hàm giải mã $d_k(y) = a^{-1} * (y - b) \bmod 99$.

a) Hãy xác định số khóa có thể được sử dụng cho hệ mã này.

b) Nếu như khóa giải mã là $k^{-1} = (16, 7)$, hãy thực hiện mã hóa xâu m = “DANGER”.

Bài 6. Giả sử hệ mã Affine được cài đặt trên Z_{126}.

a) Hãy xác định só khóa có thể có của hệ mã.

b) Giả sử khóa mã hóa là (23, 7), hãy xác định khóa giải mã.

Bài 7. Cho hệ mã Hill có m = 2

a) Ma trận A = $\begin{pmatrix} 5 & 3 \\ 13 & 7 \end{pmatrix}$ có thể được sử dụng làm khóa cho hệ mã trên không giải thích.

b) Cho A = $\begin{pmatrix} 12 & 5 \\ 3 & 7 \end{pmatrix}$ hãy thực hiện mã hóa và giải mã với xâu S = “HARD”.

Bài 8. Viết chương trình cài đặt thuật toán mã hóa và giải mã của hệ mã Ceasar

Bài 9. Viết chương trình cài đặt thuật toán mã hóa và giải mã của hệ mã Affine

Bài 10. Viết chương trình cài đặt thuật toán mã hóa và giải mã của hệ mã Hill

Bài 11. Viết chương trình cài đặt thuật toán mã hóa và giải mã của hệ mã Vigenere

CHƯƠNG 4. CHUẨN MÃ DỮ LIỆU DES VÀ CHUẨN MÃ NÂNG CAO AES

Chương này tập trung trình bày nguyên lý của mã hóa đối xứng hiện đại. Để đáp ứng được mục đích này nội dung của chương tập trung vào mật mã đối xứng được sử dụng rộng rãi nhất đó là chuẩn mã hóa dữ liệu DES (Data Encryption Standard). Mặc dầu, đã có nhiều mật mã đối xứng đã được phát triển kể từ khi DES ra đời và có mục đích nhằm thay thế nó như chuẩn mã hóa dữ liệu nâng cao AES (Advanced Encryption Standard), DES vẫn là chuẩn mã hóa dữ liệu quan trọng nhất. Hơn nữa, việc nghiên cứu chi tiết thuật toán mã hóa DES sẽ cung cấp kiến thức cần thiết để hiểu nguyên lý được sử dụng trong các thuật toán mã hóa đối xứng khác.

4.1 Chuẩn mã hóa dữ liệu DES

4.1.1 Giới thiệu

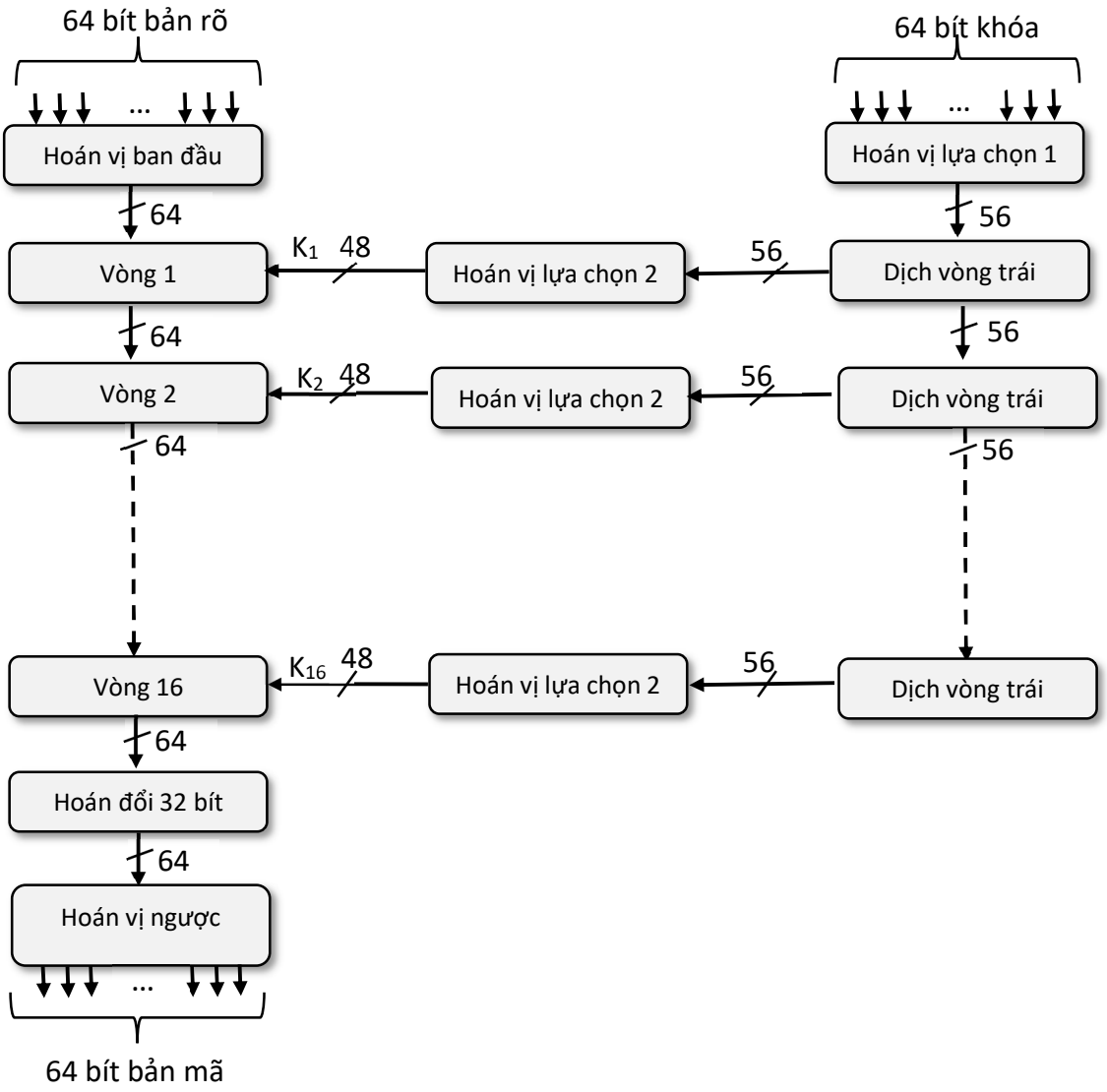
Vào cuối những năm 1960, IBM đã thiết lập một dự án nghiên cứu về mật mã máy tính do Horst Feistel đứng đầu. Dự án này kết thúc vào năm 1971 với sự phát triển của một thuật toán với ký hiệu LUCIFER và được bán cho công ty Lloyd ở London để sử dụng trong hệ thống phân phối tiền mặt, cũng do IBM phát triển. LUCIFER là một mật mã khối Feistel hoạt động trên các khối 64 bit, sử dụng kích thước khóa là 128 bit. Bởi vì những kết quả đầy hứa hẹn do dự án LUCIFER tạo ra, IBM đã bắt tay vào nỗ lực phát triển sản phẩm mã hóa thương mại có thể bán trên thị trường mà lý tưởng có thể được thực hiện trên một con chíp duy nhất. Nỗ lực do Walter Tuchman và Carl Meyer đứng đầu, không chỉ có sự tham gia của các nhà nghiên cứu IBM mà còn tư vấn bên ngoài và tư vấn kỹ thuật từ NSA. Kết quả của nỗ lực này là một phiên bản tinh chỉnh của LUCIFER không những có khả năng bảo mật tốt hơn mà còn giảm kích thước khóa xuống còn 56 bit, để có thể thực hiện trên một chíp duy nhất.

Năm 1973, Văn phòng tiêu chuẩn quốc gia Hoa Kỳ (NBS – National Bureau of Standards) đã đưa ra yêu cầu đề xuất về chuẩn mật mã quốc gia. IBM đã đệ trình các kết quả của dự án Tuchman-Meyer của mình. Đây là thuật toán tốt nhất cho đến thời điểm đó được đề xuất và được thông qua vào năm 1977 như là chuẩn mã hóa dữ liệu gọi tắt là DES. DES là thuật toán mã khối với kích thước khối là 64 bít sử dụng khóa độ dài 56 bít và là loại mã hóa khối được sử dụng rộng rãi nhất cho đến tận thời điểm hiện tại.

4.1.2 Thuật toán mã hóa

a) Mô tả

Lược đồ tổng thể của thuật toán mã hóa DES được minh họa trên hình 4.1. Cũng giống như các lược đồ mã hóa khác, có hai thông tin đầu vào cho chức năng mã hóa là bản rõ cần mã hóa và khóa. Trong trường hợp này, bản rõ phải có độ dài 64 bít và khóa có độ dài 56 bít.



Hình 4. 1 Lược đồ mã hóa tổng thể của thuật toán mã hóa DES

Nửa bên trái của lược đồ mã hóa là quá trình xử lý bản rõ gồm 3 pha. Pha đầu tiên 64 bít của bản rõ được đưa qua bộ hoán vị ban đầu (IP – Initial Permutation) để xáo trộn các bít đầu vào. Pha tiếp theo bao gồm 16 vòng giống nhau để thực hiện 2 chức năng hoán vị và thay thế. Kết quả đầu ra của vòng cuối cùng (vòng 16) bao gồm 64 bít, đây chính là đầu ra của hàm với 2 đầu vào là bản rõ và khóa. Nửa trái và phải kết quả của vòng 16 được hoán đổi cho nhau để tạo ra tiền kết quả cuối cùng. Pha cuối cùng, đưa tiền kết quả cuối cùng qua bộ hoán vị ngược (IP^{-1}) để tạo ra 64 bít bản mã.

b) Hoán vị ban đầu

Hoán vị ban đầu và hoán vị ngược được xác định thông qua bảng 4.1 và 4.2 bên dưới.

Bảng 4. 1 hoán vị ban đầu (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

Bảng 4. 2 hoán vị ngược (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

Để minh họa hoán vị ban đầu và hoán vị ngược của 64 bít đầu vào B như sau:

$$B_1 \quad B_2 \quad B_3 \quad B_4 \quad B_5 \quad B_6 \quad B_7 \quad B_8$$

$$B_9 \quad B_{10} \quad B_{11} \quad B_{12} \quad B_{13} \quad B_{14} \quad B_{15} \quad B_{16}$$

$$B_{17} \quad B_{18} \quad B_{19} \quad B_{20} \quad B_{21} \quad B_{22} \quad B_{23} \quad B_{24}$$

$$B_{25} \quad B_{26} \quad B_{27} \quad B_{28} \quad B_{29} \quad B_{30} \quad B_{31} \quad B_{32}$$

B_{33} B_{34} B_{35} B_{36} B_{37} B_{38} B_{39} B_{40}

B_{41} B_{42} B_{43} B_{44} B_{45} B_{46} B_{47} B_{48}

B_{49} B_{50} B_{51} B_{52} B_{53} B_{54} B_{55} B_{56}

B_{57} B_{58} B_{59} B_{60} B_{61} B_{62} B_{63} B_{64}

Trong đó, B_i là các bít nhị phân, khi đó X = IP(B) sẽ có kết quả như sau:

B_{58} B_{50} B_{42} B_{34} B_{26} B_{18} B_{10} B_2

B_{60} B_{52} B_{44} B_{36} B_{28} B_{20} B_{12} B_4

B_{62} B_{54} B_{46} B_{38} B_{30} B_{22} B_{14} B_6

B_{64} B_{56} B_{48} B_{40} B_{32} B_{24} B_{16} B_8

B_{57} B_{49} B_{41} B_{33} B_{25} B_{17} B_9 B_1

B_{59} B_{51} B_{43} B_{35} B_{27} B_{19} B_{11} B_3

B_{61} B_{53} B_{45} B_{37} B_{29} B_{21} B_{13} B_5

B_{63} B_{55} B_{47} B_{39} B_{31} B_{23} B_{15} B_7

Nếu ta thực hiện hoán vị ngược kết quả này sẽ thu được kết quả ban đầu $Y = IP^{-1}(X) = IP^{-1}(IP(B))$.

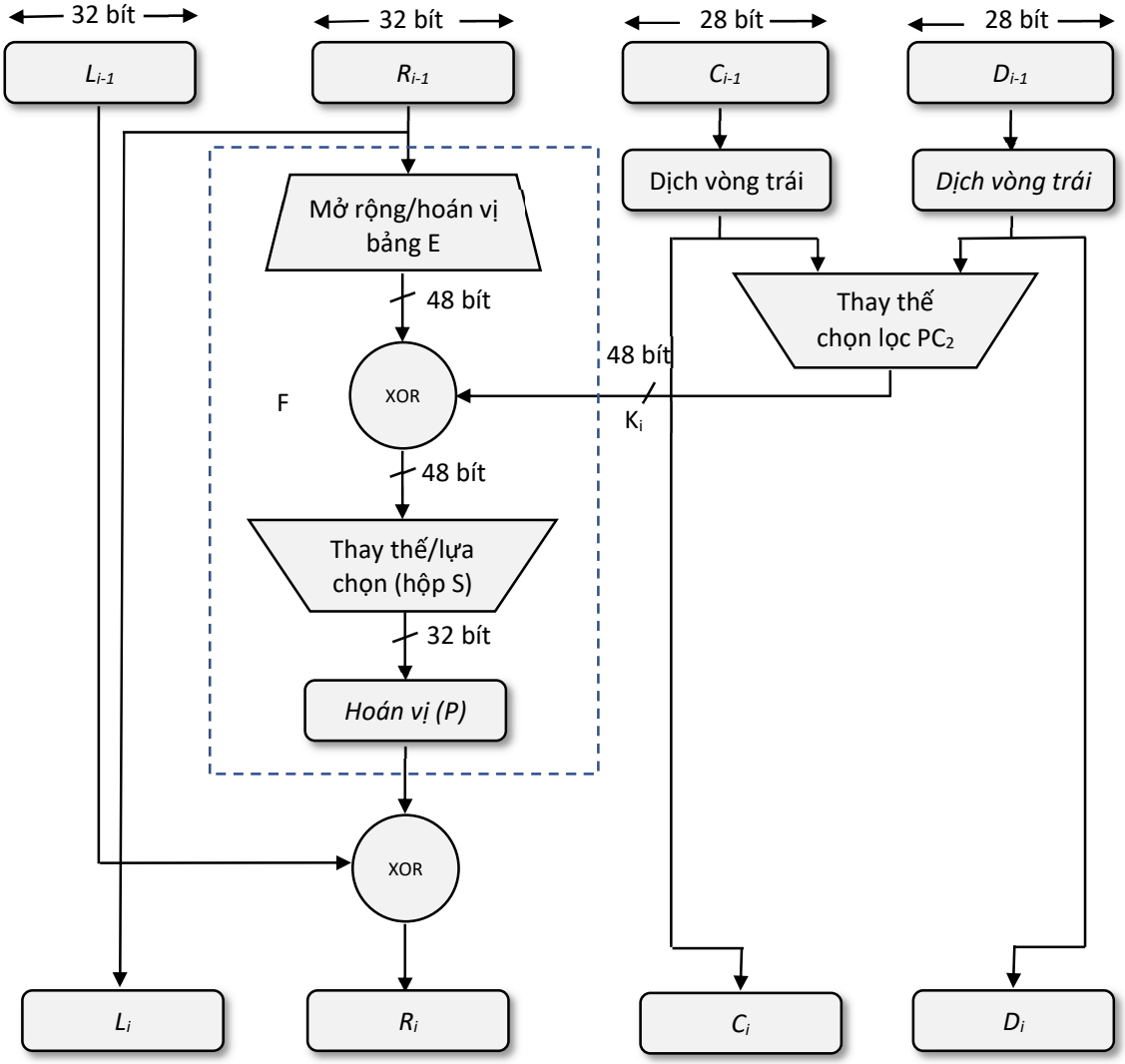
c) Chi tiết một vòng

Chi tiết một vòng mã hóa được minh họa trên hình 4.2. Nửa bên trái của hình, 64 bít đầu ra của vòng trước được tách làm 2 nửa trái, phải ký hiệu là L, R tương ứng và là đầu vào của vòng hiện tại. Toàn bộ quá trình xử lý tại mỗi vòng có thể tóm tắt bởi các công thức sau:

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

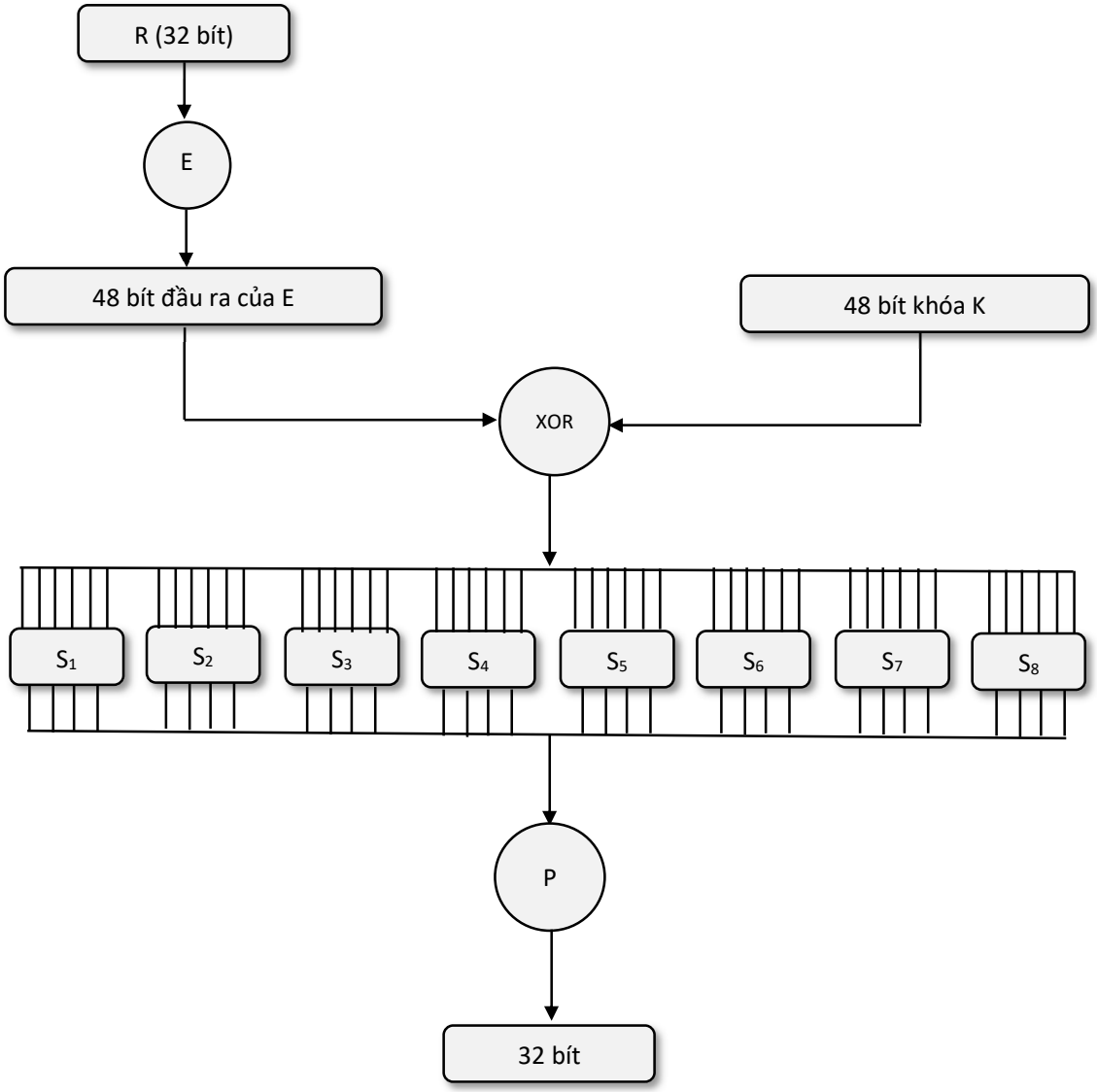
$$R_i = L_{i-1}\ XOR\ F(R_{i-1}, K_i)$$

Nửa phải của vòng liền trước được đưa qua bộ mở rộng và hoán vị theo bảng E để tạo thành 48 bít để đưa vào thực hiện phép toán XOR bít với 48 bít khóa của vòng hiện tại. Kết quả đầu ra được đưa qua bộ thay thế/lựa chọn sử dụng các hộp S để tạo ra 32 bít. Cuối cùng 32 bít này được đưa qua bộ hoán vị P để tạo ra kết quả 32 bít đầu ra của hàm F. Đầu ra của hàm F được thực hiện phép XOR bít với nửa trái của vòng liền trước để tạo ra nửa phải của vòng hiện tại.



Hình 4. 2 Minh họa chi tiết một vòng mã hóa DES

Vai trò của các hộp S trong hàm F được minh họa trên hình 4.3. Bộ phận thay thế gồm một tập 8 hộp S. Mỗi hộp S nhận 6 bít đầu vào được tạo bằng cách chia 48 bít đầu ra của mạch XOR thành 8 nhóm và sinh ra 4 bít đầu ra. Quá trình chuyển đổi này được diễn tả như sau: Bít đầu tiên và cuối cùng của 6 bít đầu vào hộp S_i tạo nên 2 bít nhị phân dùng để chọn 1 trong 4 dòng (từ 0 đến 3), bốn bít ở giữa dùng để chọn 1 trong 16 cột (từ 0 đến 15) của bảng cho hộp S_i. Giá trị thập phân của ô xác định bởi dòng và cột được biểu diễn sang 4 bít nhị phân chính là đầu ra của hộp S_i. Ví dụ đầu vào của hộp S_1 là 011001, 2 bít chọn dòng là 01 (dòng 1), 4 bít chọn cột là 1100 (cột 12). Tra bảng cho hộp S1, giá trị của ô dòng 1, cột 12 là 9, do đó đầu ra sẽ là 1001.



Hình 4. 3 Minh họa cách xác định đầu ra của hàm F

Bảng 4. 3 Bảng hoán vị 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*	*1*

Bảng 4. 4 Hoán vị 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*

Định nghĩa của các hộp S được liệt kê ở bảng 4.5 sau:

Bảng 4. 5 Bảng thay thế của các hộp S

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

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

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	15	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

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
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

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

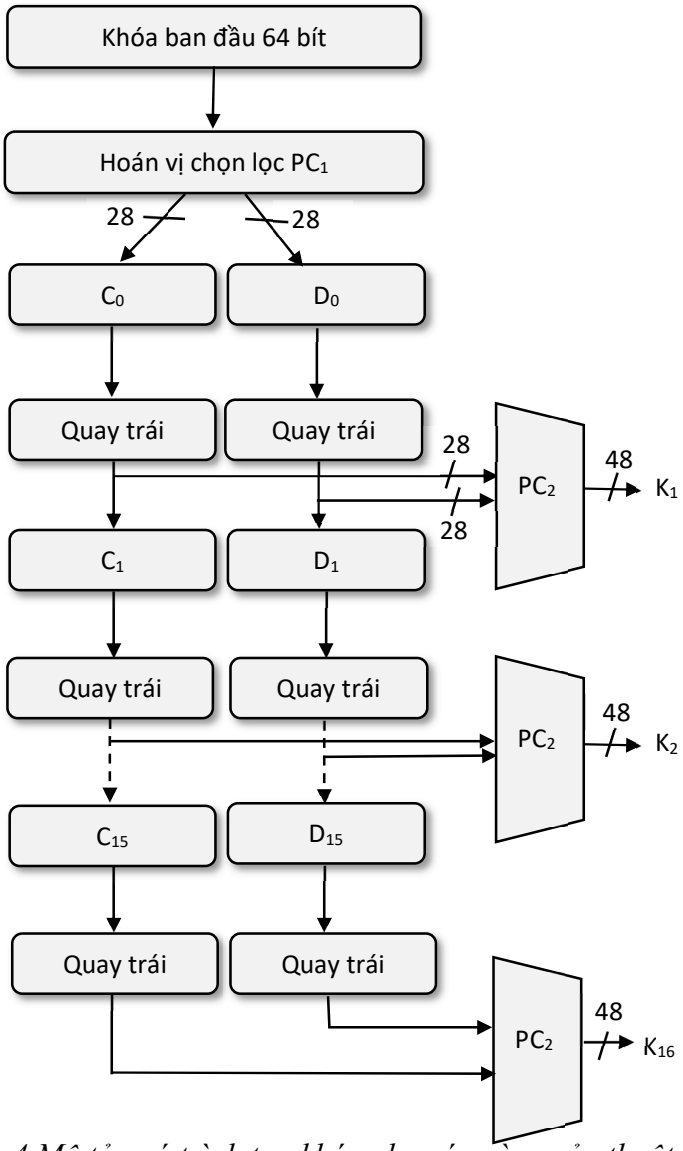
S_7	4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
	13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
	1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
	6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12

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

d) Tạo khóa

Chi tiết quá trình tạo khóa cho các vòng từ khóa ban đầu được minh họa trên hình 4.4. Khóa ban đầu có kích thước 64 bít được đánh số từ 1 đến 64, được đưa qua bộ hoán vị lựa chọn PC_1 như bảng 4.6 để tạo ra chuỗi 56 bít ở đầu ra. Chuỗi bít này được tách làm 2 nửa trái, phải được ký hiệu lần lượt C_0 và D_0, mỗi nửa có chiều dài 28 bít. Tại mỗi vòng, C_{i-1}, D_{i-1} được đưa vào bộ quay trái 1 hoặc 2 bít tùy thuộc vào vòng như bảng 4.7. Giá trị sau khi quay này đóng vai trò là đầu vào của vòng tiếp theo. Chúng cũng là đầu

vào của bộ thay thế lựa chọn PC_2 (bảng 4.8) để tạo ra 48 bít đầu ra. Các bít này đóng vai trò là khóa của vòng hiện tại (K_i) và là đầu vào của hàm $F(R_{i-1}, K_i)$.



Hình 4. 4 Mô tả quá trình tạo khóa cho các vòng của thuật toán DES

Bảng 4. 6 Bảng thay thế lựa chọn PC1

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

Bảng 4. 7 Bảng xác định số bít quay trong quá trình tạo khóa ở mỗi vòng

Vòng	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Số bít quay	1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

Bảng 4. 8 Bảng thay thế lựa chọn PC2

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

e) Ví dụ minh họa

Xác định kết quả của vòng 1 khi mã hóa bản rõ là chuỗi ‘ABCDEFGH’ với khóa là ‘abcdefgh’.

Chuỗi bản rõ ‘ABCDEFGH’ được mã hóa theo bảng mã ASCII ở dạng hệ hex và nhị phân như sau: Hệ hex: 41 42 43 44 45 46 47 48. Hệ nhị phân:

0100000101000001001000011010001000100010101000110010001110100100 0

Bản rõ được đưa qua bộ hoán vị ban đầu IP ta thu được 64 bít sau:

1111111100000000011110000101010100000001111111111000000001100110

58 (1)	50 (1)	42 (1)	34 (1)	26 (1)	18 (1)	10 (1)	2 (1)
60 (0)	52 (0)	44 (0)	36 (0)	28 (0)	20 (0)	12 (0)	4 (0)
62 (0)	54 (1)	46 (1)	38 (1)	30 (1)	22 (0)	14 (0)	6 (0)

64 (0)	56 (1)	48 (0)	40 (1)	32 (0)	24 (1)	16 (0)	8 (1)
57 (0)	49 (0)	41 (0)	33 (0)	25 (0)	17 (0)	9 (0)	1 (0)
59 (1)	51 (1)	43 (1)	35 (1)	27 (1)	19 (1)	11 (1)	3 (1)
61 (1)	53 (0)	45 (0)	37 (0)	29 (0)	21 (0)	13 (0)	5 (0)
63 (0)	55 (1)	47 (1)	39 (0)	31 (0)	23 (1)	15 (1)	7 (0)

Bản rõ sau khi đi qua bộ hoán vị ban đầu được phân làm nửa trái và phải như sau:

L_0 = 11111111000000000111100001010101

R_0 = 00000001111111111000000001100110

Ta có: ***$L_1 = R_0$ = 00000001111111111000000001100110***

$R_1 = L_0$ XOR $F(R_0, K_1)$

Để tính *$F(R_0,K_1)$*, đầu tiến ta đưa *R_0* qua phần mở rộng *E* (dựa vào bảng 4.3).

32 (0)	*1 (0)*	*2 (0)*	*3 (0)*	*4 (0)*	*5 (0)*
4 (0)	*5 (0)*	*6 (0)*	*7 (0)*	*8 (1)*	*9 (1)*
8 (1)	*9 (1)*	*10 (1)*	*11 (1)*	*12 (1)*	*13 (1)*
12 (1)	*13 (1)*	*14 (1)*	*15 (1)*	*16 (1)*	*17 (1)*
16 (1)	*17 (1)*	*18 (0)*	*19 (0)*	*20 (0)*	*21 (0)*
20 (0)	*21 (0)*	*22 (0)*	*23 (0)*	*24 (0)*	*25 (0)*
24 (0)	*25 (0)*	*26 (1)*	*27 (1)*	*28 (0)*	*29 (0)*
28 (0)	*29 (0)*	*30 (1)*	*31 (1)*	*32 (0)*	*1 (0)*

Do đó: ***$E(R_0)$ = 000000 000011 111111 111111 110000 000000 001100 001100***

Tiếp theo, ta cần xác định khóa *K_1*. Chuỗi khóa ban đầu ‘abcdefgh’ được mã hóa theo bảng mã ASCII ở dạng hệ hex và nhị phân tương ứng như sau: *61 62 63 64 65 66 67 68* và *01100001 01100010 01100011 01100100 01100101 01100110 01100111 01101000*.

Khóa ban đầu được đưa qua bảng hoán vị chọn lọc *PC_1* để thu được đầu ra 56 bít dựa vào bảng 4.6.

57 (0)	49 (0)	41 (0)	33 (0)	25 (0)	17 (0)	9 (0)
1 (0)	58 (1)	50 (1)	42 (1)	34 (1)	26 (1)	18 (1)
10 (1)	2 (1)	59 (1)	51 (0)	43 (1)	35 (1)	27 (1)
19 (1)	11 (1)	3 (1)	60 (1)	52 (0)	44 (0)	36 (0)

63 (0)	55 (1)	47 (1)	39 (0)	31 (0)	23 (1)	15 (1)
7 (0)	62 (0)	54 (1)	46 (1)	38 (1)	30 (1)	22 (0)
14 (0)	6 (0)	61 (1)	53 (0)	45 (0)	37 (0)	29 (0)
21 (0)	13 (0)	5 (0)	28 (0)	20 (0)	12 (0)	4 (0)

Ta có:

$PC_1(K)$ = 00000000 11111111 10111111 10000110 01100111 10001000 00000000.

Như vậy:

C_0 = 00000000 11111111 10111111 1000

D_0 = 01100110 01111000 10000000 0000

Tại vòng 1 thì ta cần quay trái C_0 và D_0 1 bít (theo bảng 4.7) ta thu được:

Quay trái C_0 1 bit thu được: *00000001 11111111 01111111 0000*.

Quay trái D_0 1 bit thu được: *11001100 11110001 00000000 0000*.

Ghép C_0D_0 ta thu được kết quả:

00000001 11111111 01111111 00001100 11001111 00010000 00000000

Tiếp theo đưa kết quả này qua bộ hoán vị chọn lọc PC_2. Dựa vào bảng 4.8 ta thu được chuỗi khóa K_1: *10110000 10111110 01100110 00010011 00101010 10000010*

14 (1)	17 (0)	11 (1)	24 (1)	1 (0)	5 (0)	3 (0)	28 (0)
15 (1)	6 (0)	21 (1)	10 (1)	23 (1)	19 (1)	12 (1)	4 (0)
26 (0)	8 (1)	16 (1)	7 (0)	27 (0)	20 (1)	13 (1)	2 (0)
41 (0)	52 (0)	31 (0)	37 (1)	47 (0)	55 (0)	30 (1)	40 (1)
51 (0)	45 (0)	33 (1)	48 (0)	44 (1)	49 (0)	39 (1)	56 (0)
34 (1)	53 (0)	46 (0)	42 (0)	50 (0)	36 (0)	29 (1)	32 (0)

Ta tính: ***$E(R_0)$ XOR K_1 = 101100001000000110011001110100110010100110001110***

```
  000000000011111111111111110000000000001100001100
XOR
  101100001011111001100110000100110010101010000010
  ------------------------------------------------
  101100001000000110011001110100110010100110001110
```

Kết quả thu được chia làm 8 khối, mỗi khối 6 bít để đưa lần lượt vào các hộp từ S_1 đến S_8:

Khối 1: 101100 → 2 bít chọn dòng 10 → dòng 2, 4 bít chọn cột 0110 → cột 6

Khối 2: 001000 → 2 bít chọn dòng 00 → dòng 0, 4 bít chọn cột 0100 → cột 4

Khối 3: 000110 → 2 bít chọn dòng 00 → dòng 0, 4 bít chọn cột 0011 → cột 3

Khối 4: 011001 → 2 bít chọn dòng 01 → dòng 1, 4 bít chọn cột 1100 → cột 12

Khối 5: 110100 → 2 bít chọn dòng 10 → dòng 2, 4 bít chọn cột 1010 → cột 10

Khối 6: 110010 → 2 bít chọn dòng 10 → dòng 2, 4 bít chọn cột 1001 → cột 9

Khối 7: 100110 → 2 bít chọn dòng 10 → dòng 2, 4 bít chọn cột 0011 → cột 3

Khối 8: 001110 → 2 bít chọn dòng 00 → dòng 0, 4 bít chọn cột 0111 → cột 7

Tra lần lượt các bảng thay thế hộp *S* (bảng 4.5) ta thu được kết quả đầu ra của hàm *F* như sau:

Đầu ra S_1 (dòng *2*, cột *6*) là *2* mã hóa *4* bít thu được: *0010*

Đầu ra S_2 (dòng *0*, cột *4*) là *6* mã hóa *4* bít thu được: *0110*

Đầu ra S_3 (dòng *0*, cột *3*) là *14* mã hóa *4* bít thu được: *1110*

Đầu ra S_4 (dòng *1*, cột *12*) là *1* mã hóa *4* bít thu được: *0001*

Đầu ra S_5 (dòng *2*, cột *10*) là *12* mã hóa *4* bít thu được: *1100*

Đầu ra S_6 (dòng *2*, cột *9*) là *0* mã hóa *4* bít thu được: *0000*

Đầu ra S_7 (dòng *2*, cột *3*) là *13* mã hóa *4* bít thu được: *1101*

Đầu ra S_8 (dòng *0*, cột *7*) là *1* mã hóa *4* bít thu được: *0001*

Kết quả đầu ra của các hộp *S* là: *00100110 11100001 11000000 11010001*. Đưa kết quả này qua bộ hoán vị *P* (bảng 4.4) ta thu được:

$F(R_0,K_1)$ = 11000011 00010101 00001011 00010101

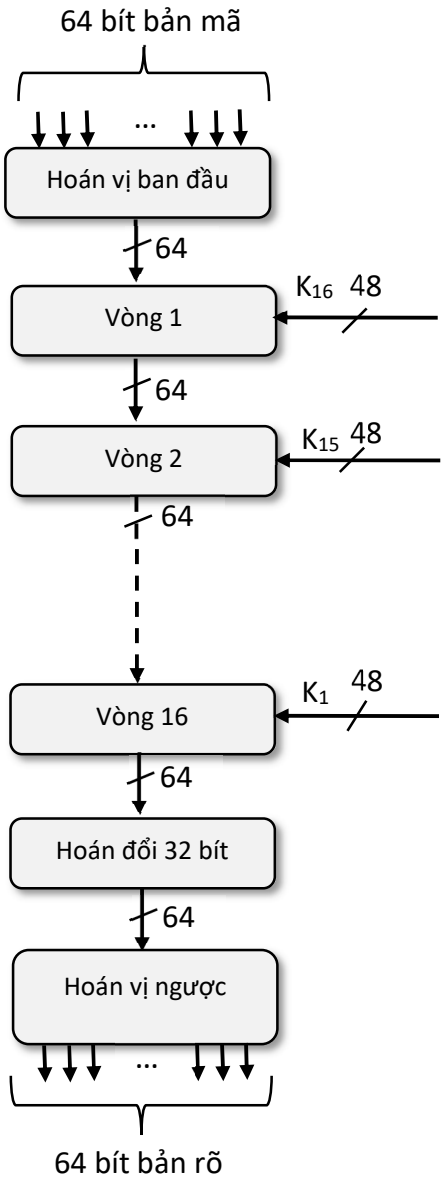
16 (1)	*7 (1)*	*20 (0)*	*21 (0)*	*29 (0)*	*12 (0)*	*28 (1)*	*17 (1)*
1 (0)	*15 (0)*	*23 (0)*	*26 (1)*	*5 (0)*	*18 (1)*	*31 (0)*	*10 (1)*
2 (0)	*8 (0)*	*24 (0)*	*14 (0)*	*32 (1)*	*27 (0)*	*3 (1)*	*9 (1)*
19 (0)	*13 (0)*	*30 (0)*	*6 (1)*	*22 (0)*	*11 (1)*	*4 (0)*	*25 (1)*

Cuối cùng, ***$R_1 = L_0$ XOR $F(R_0,K_1)$ = 00111100000101010111001101000000***

```
XOR   11111111000000000111100001010101
      11000011000101010000101100010101
      ----------------------------------
      00111100000101010111001101000000
```

4.1.3 Thuật toán giải mã

Thuật toán giải mã DES sử dụng giống như thuật toán mã hóa chỉ khác là khóa được sử dụng trong các vòng theo chiều ngược lại. Vòng 1 sử dụng khóa K_{16} và vòng 16 sử dụng khóa K_1. Thuật toán giải mã DES được minh họa trên hình 4.5.



Hình 4. 5 Minh họa thuật toán giải mã DES

4.1.4 Hiệu ứng lan truyền

Một tính chất quan trọng cần thiết của mọi thuật toán mã hóa là chỉ cần thay đổi nhỏ trong bản rõ hay trong khóa sẽ dẫn đến thay đổi lớn trong bản mã. Cụ thể, chỉ cần thay đổi một bít trong bản rõ hay khóa thì dẫn đến sự thay đổi của nhiều bít trong bản mã. Tính chất này được gọi là hiệu ứng lan truyền. Nhờ có tính chất này mà người thám mã không thể giới hạn miền tìm kiếm của bản rõ hay của khóa nên phải thực hiện vét cạn khóa.

DES là thuật toán mã hóa có hiệu ứng này. Để chứng minh ta xét 2 trường hợp sau:

Trường hợp 1: Khóa giống nhau, bản rõ 64 bít chỉ khác nhau 1 bít sau:

00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000

10000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000

được mã hóa bằng khóa: *0000001 1001011 0100100 1100010 0011100 0011000 0011100 0110010*. Bảng 4.9 chỉ ra số bít khác nhau trong bản mã tại mỗi vòng. Qua bảng này ta thấy chỉ sau qua 2 vòng mã thì đã có 21 bít khác nhau trong bản mã. Kết quả cuối cùng có 34 bít khác nhau trong bản mã.

.

Bảng 4. 9 Số bít khác nhau khi mã hóa DES cùng khóa khác bản rõ 1 bít

Vòng	***Số bít khác nhau trong bản mã***
1	6
2	21
3	35
4	39
5	34
6	32
7	31
8	29
9	42
10	44
11	32
12	30
13	30
14	26
15	29
16	34

Trường hợp 2: Cùng bản rõ, khóa khác nhau 1 bít. Xét bản rõ sau:

01101000 10000101 00101111 01111010 00010011 01110110 11101011 10100100

với 2 khóa chỉ khác nhau 1 bít:

1110010 1111011 1101111 0011000 0011101 0000100 0110001 11011100

0110010 1111011 1101111 0011000 0011101 0000100 0110001 11011100

Bảng 4.10 chỉ ra số bít khác nhau trong bản mã, cũng chỉ một vài vòng thì số bit đã khác nhau khoảng một nửa để tạo ra hiệu ứng lan truyền.

Bảng 4. 10 Số bít khác nhau khi mã hóa DES một bản rõ, khóa khác 1 bít

Vòng	***Số bít khác nhau trong bản mã***
1	2
2	14
3	28
4	32
5	30
6	32
7	35
8	34
9	40
10	38
11	31
12	33
13	28
14	26
15	34
16	35

4.1.5 Điểm mạnh của thuật toán DES

Để xem xét điểm mạnh của DES ta đi phân tích khả năng thám mã bằng 3 kỹ thuật thông dụng là: vét cạn khóa (Brute Force Attack); vi sai (differential cryptanalysis) và tuyến tính (linear cryptanalysis).

Thám mã theo phương pháp vét cạn khóa là cố gắng thử mọi khóa có thể. Nếu khóa có độ dài là 8 bít thì số khóa có thể là 256 khóa. Do đó tối đa cần 256 lần thử là tìm ra khóa. DES sử dụng khóa có độ dài 56 bít. Do đó, không gian khóa là 2^{56} nên thám mã theo kỹ thuật vét cạn là không khả thi vì vào năm 1979, Diffie và Hellman tuyên bố rằng với một máy tính chuyên dụng thì bản mã hóa DES có thể được phá bằng cách thử

mọi trường hợp của khóa trong vòng một ngày – giá của máy tính đó là 20 triệu đôla. Vào năm 1981, Diffie đã tăng lên là cần hai ngày để tìm kiếm và giá của chiếc máy tính đó là 50 triệu đôla.

Thám mã theo phương pháp vi sai: Năm 1990 Biham và Shamir đã giới thiệu phương pháp phá mã vi sai. Phương pháp vi sai tìm khóa ít tốn thời gian hơn vét cạn khóa. Tuy nhiên, phương pháp thám mã này lại đòi hỏi phải có 2^{47} cặp bản rõ - bản mã được lựa chọn. Vì vậy phương pháp này là bất khả thi dù rằng số lần thử có thể ít hơn phương pháp vét cạn.

Thám mã theo phương pháp tuyến tính: Năm 1997 Matsui đưa ra phương pháp thám mã tuyến tính. Trong phương pháp này, cần phải biết trước 2^{43} cặp bản rõ - bản mã. Tuy nhiên, 2^{43} cũng là một con số lớn nên thám mã tuyến tính cũng không phải là một phương pháp khả thi.

4.2 Mã hóa DES 2 lần (Double DES) và DES 3 lần (Triple DES)

Một trong những lỗ hổng tiềm ẩn của mã hóa DES là có thể bị tấn công theo kiểu vét cạn khóa do độ dài của khóa không lớn. Để khắc phục điểm yếu này là sử dụng DES nhiều lần với khóa khác nhau cho cùng một bản rõ. Đơn giản nhất là sử dụng DES 2 lần với 2 khóa khác nhau và được gọi là Double DES.

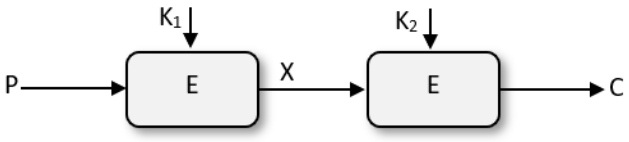
Ứng với bản rõ *P* và hai khóa K_1 và K_2 thì bản mã *C* được tạo ra như sau:

$$C = E(K_2, E(K_1, P))$$

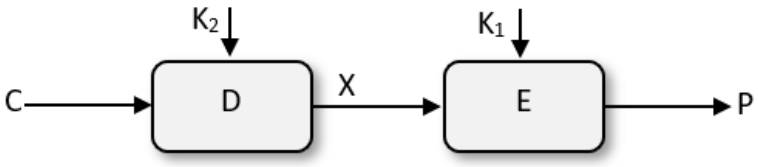
Giải mã cũng cần sử dụng khóa nhưng với thứ tự ngược lại.

$$P = D(K_1, D(K_2, C))$$

Hình 4.6 và 4.7 minh họa mã hóa và giải mã Double DES.



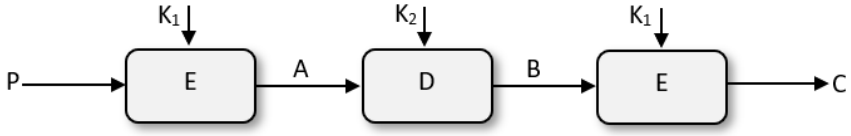
Hình 4. 6 Mã hóa Double DES



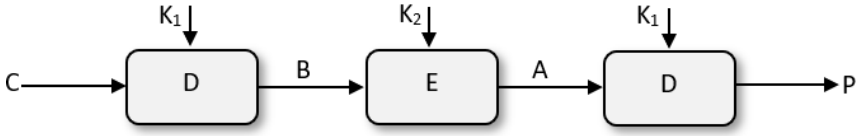
Hình 4. 7 Giải mã Double DES

Sử dụng Double DES tương đương với DES có khóa 112 bít. Tuy nhiên, Double DES vẫn bị tấn công bằng phương pháp gặp nhau ở giữa (meet-in-the-middle). Thuật

toán tấn công này dựa vào việc quan sát, nếu ta có $C = E(K_2, E(K_1, P))$ thì theo hình 4.6 và 4.7 $X = E(K_1, P) = D(K_2, C)$. Do đó, với một cặp đã biết *(P,C)* kẻ thám mã sẽ thực hiện các bước như sau: Đầu tiên, sẽ mã hóa *P* với 2^{56} khả năng của khóa K_1 và lưu các kết quả vào một bảng, rồi sắp xếp bảng này theo giá trị của *X*. Kế tiếp, giải mã *C* sử dụng tất cả 2^{56} khả năng của khóa K_2, ứng với mỗi bản giải mã được tạo ra thì đem so khớp với các giá trị trong bảng ở bước đầu tiên. Nếu khớp xảy ra, thì kiểm tra hai khóa tìm được với một cặp *(P,C)* khác đã biết. Nếu hai khóa tạo ra bản mã chính xác thì các khóa này chính là khóa cần tìm. Do đó, để tăng độ khó cho việc thám mã sử dụng thuật toán tấn công gặp nhau ở giữa thì người ta sử dụng mã hóa DES 3 lần với 3 khóa khác nhau theo công thức $C = E(E(E(K_1, P), K_2), K_3)$. Trong thực tế, để đảm bảo tính tương thích ngược với DES một lần người ta thường dùng mã hóa DES 3 lần theo nguyên tắc Mã hóa – Giải mã – Mã hóa (EDE) sử dụng 2 khóa như hình 4.8 và 4.9.



Hình 4. 8 Mã hóa DES 3 lần sử dụng 2 khóa



Hình 4. 9 Giải mã DES 3 lần sử dụng 2 khóa

Như đã để cập bên trên, để tương thích ngược với DES thì người ta sử dụng EDE thay vì EEE. Thật vậy, nếu chọn khóa $K_1 = K_2$ thì khi đó $C = E(K_1, D(K_1, E(K_1, P))) = E(K_1, P)$. Như vậy, DES 3 lần trở thành DES một lần. Nếu sử dụng 3 khóa khác nhau thì DES 3 lần sử dụng EDE cũng trở thành DES 1 lần nếu ta chọn $K_1 = K_2$ hoặc $K_2 = K_3$. Thật vậy, nếu chọn $K_1 = K_2$ thì $C = E(K_3, D(K_2, E(K_1, P))) = E(K_3, P)$ và nếu chọn $K_2 = K_3$ thì $C = E(K_3, D(K_2, E(K_1, P))) = E(K_1, P)$.

4.3 Chuẩn mã nâng cao (AES – Advanced Encryption Standard)

4.3.1 Giới thiệu

Vào năm 1999, cục tiêu chuẩn quốc gia Hoa Kỳ (NIST) đã ban hành một phiên bản mới của tiêu chuẩn DES chỉ ra rằng DES chỉ nên được sử dụng cho các hệ thống kế cũ và DES ba lần được sử dụng. DES ba lần có hai ưu điểm đảm bảo cho việc sử dụng rộng rãi trong vài năm tới. Đầu tiên, với độ dài khóa 168-bit, nó khắc phục được lỗ hổng đối với cuộc tấn công vét cạn của DES. Thứ hai, thuật toán mã hóa cơ bản trong DES ba lần cũng giống như trong DES. Thuật toán này đã được giám sát kỹ lưỡng hơn bất

kỳ thuật toán mã hóa nào khác trong một khoảng thời gian dài và không có cuộc tấn công phá mã hiệu quả nào dựa trên thuật toán thay vì vét cạn được tìm thấy. Do đó, DES ba lần có khả năng chống phá mã rất tốt. Nếu bảo mật là yếu tố duy nhất được xem xét, thì DES ba lần sẽ là lựa chọn thích hợp cho thuật toán mã hóa tiêu chuẩn trong nhiều thập kỷ tới.

Hạn chế chính của DES ba lần là thuật toán tương đối chậm trong phần mềm. DES ban đầu được thiết kế để triển khai bằng phần cứng giữa những năm 1970 và không tạo ra mã phần mềm hiệu quả. DES ba lần, có số vòng gấp ba lần DES, do đó thực hiện chậm hơn DES ban đầu. Một nhược điểm phụ là cả DES và DES ba lần đều sử dụng kích thước khối 64-bit. Vì lý do cả hiệu quả và bảo mật, kích thước khối lớn hơn là cần thiết.

Vì những nhược điểm này, DES ba lần không phải là ứng cử viên thích hợp để sử dụng lâu dài. Để thay thế, vào năm 1997 NIST đã đưa ra lời kêu gọi đề xuất Tiêu chuẩn mã hóa nâng cao (AES) mới, tiêu chuẩn này phải có sức mạnh bảo mật bằng hoặc tốt hơn DES ba lần và cải thiện đáng kể hiệu quả. Ngoài các yêu cầu chung này, NIST quy định rằng AES phải là mật mã khối đối xứng với độ dài khối 128 bit và hỗ trợ độ dài khóa có thể là 128, 192 và 256 bit.

Trong vòng đánh giá đầu tiên, 15 thuật toán được đề xuất đã được chấp nhận. Vòng thứ hai thu hẹp còn 5 thuật toán. NIST đã hoàn thành quá trình đánh giá của mình và xuất bản tiêu chuẩn cuối cùng vào tháng 11 năm 2001. NIST đã chọn Rijndael làm thuật toán AES được đề xuất. Hai nhà nghiên cứu đã phát triển và gửi Rijndael cho AES đều là những nhà mật mã học đến từ Bỉ: Tiến sĩ Joan Daemen và Tiến sĩ Vincent Rijmen.

Cuối cùng, AES được thiết kế để thay thế DES ba lần, nhưng quá trình này sẽ mất một số năm. NIST dự đoán rằng DES ba lần vẫn sẽ là một thuật toán được sử dụng trong tương lai gần.

Bảng 4.11 liệt kê tham số của AES tùy thuộc vào kích thước của khóa. Trong phần này ta lựa chọn khóa 128 bits là kích thước thông dụng thường được triển khai trong thực tế.

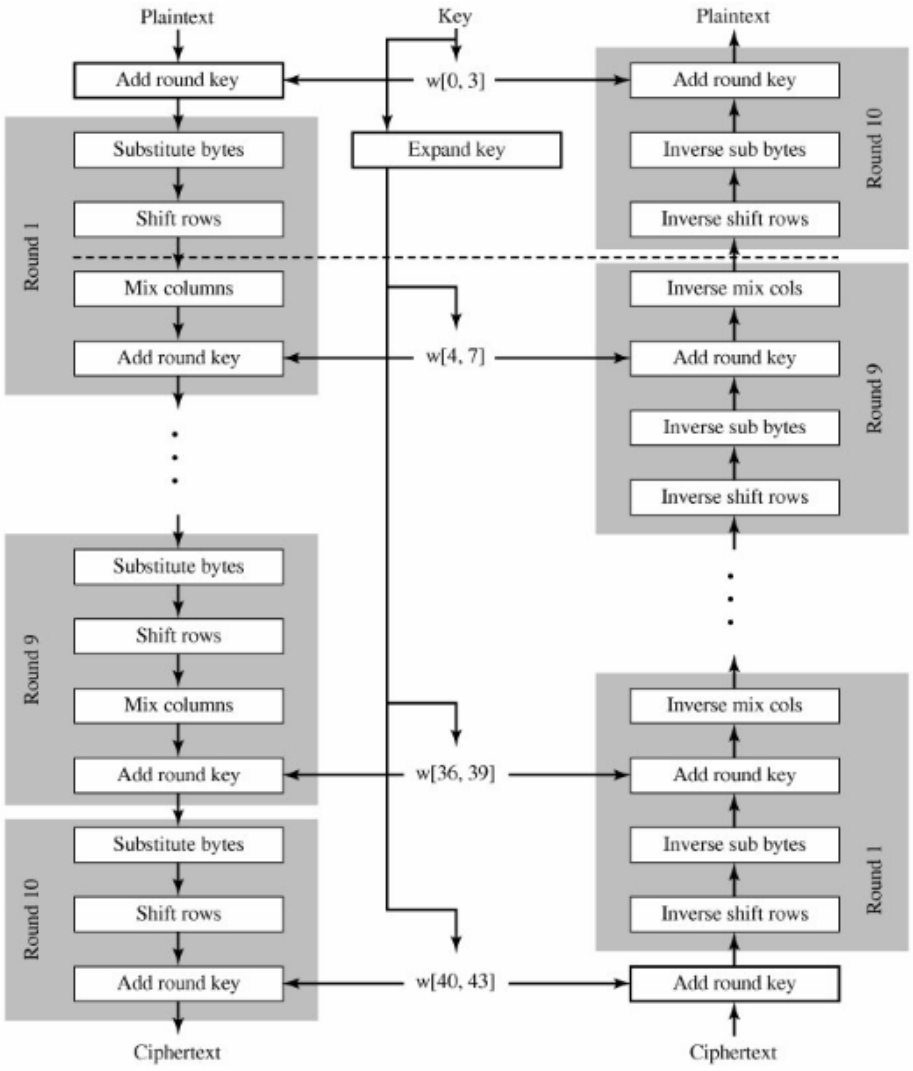
Bảng 4. 11 Tham số của AES

Kích thước khóa (words/bytes/bits)	4/16/128	6/24/192	8/32/256
Kích thước khối của bản rõ (words/bytes/bits)	4/16/128	4/16/128	4/16/128
Số vòng	10	12	14
Kích thước khóa tại mỗi vòng (words/bytes/bits)	4/16/128	4/16/128	4/16/128

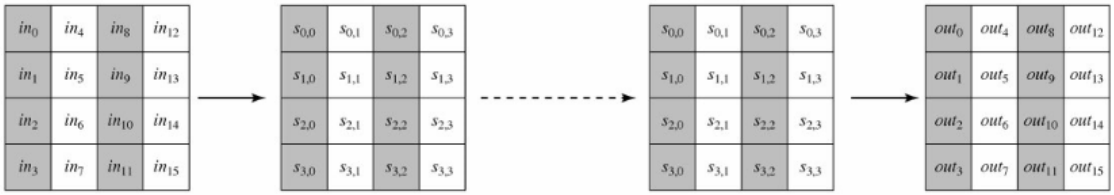
Kích thước khóa mở rộng (words/bytes)	44/176	52/208	60/240

4.3.2 Mã hóa và giải mã

Cấu trúc tổng thể của mã hóa AES được mô tả trên hình 4.10. Đầu vào cho thuật toán mã hóa và giải mã là một khối 128 bít, khối bít này được mô tả là một ma trận vuông, mỗi ô là 1 byte. Khối này được sao chép vào một mảng trạng thái, được sửa đổi ở mỗi giai đoạn mã hóa hoặc giải mã. Sau giai đoạn cuối cùng, mảng trạng thái này được sao chép vào một ma trận đầu ra. Các hoạt động này được mô tả trong hình 4.11. Tương tự, khóa 128 bit được mô tả như một ma trận vuông, mỗi phần tử là một byte. Khóa này sau đó được mở rộng thành một mảng các từ (word), mỗi từ là bốn byte và tổng chiều dài khóa là 44 từ cho khóa 128 bit như hình 4.12. Lưu ý rằng thứ tự của các byte trong ma trận là theo cột. Vì vậy, bốn byte đầu tiên của bản rõ 128 bit đầu vào chiếm cột đầu tiên của ma trận, bốn byte thứ hai chiếm cột thứ hai, v.v. Tương tự, bốn byte đầu tiên của khóa mở rộng, tạo thành một từ, chiếm cột đầu tiên của ma trận w.



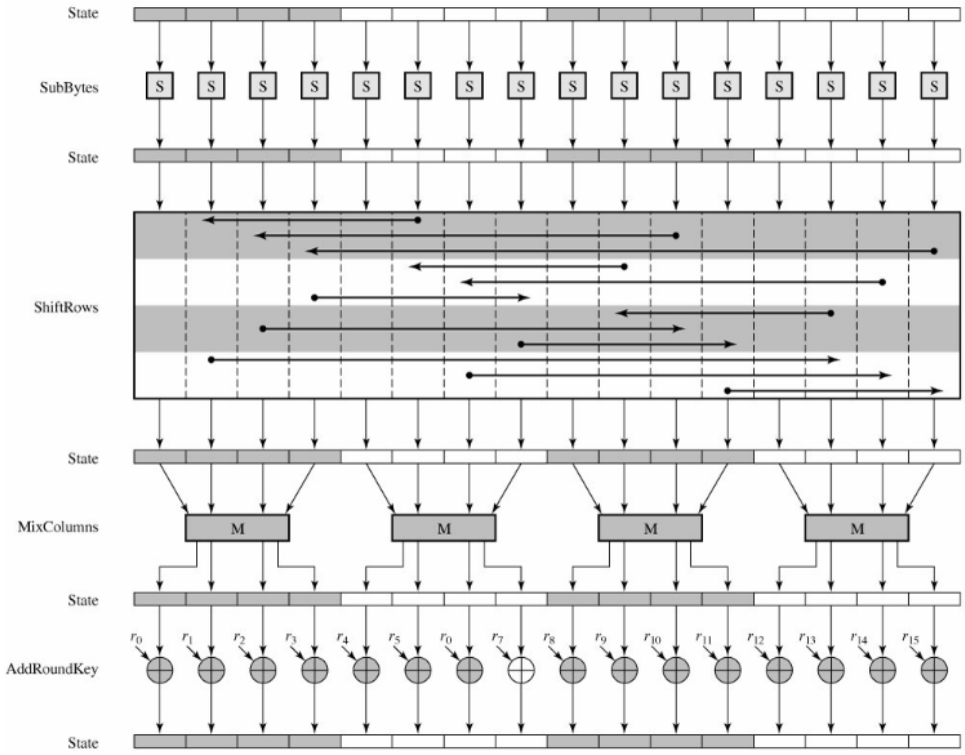
Hình 4. 10 Cấu trúc mã hóa và giải mã AES



Hình 4. 11 Đầu vào, mảng trạng thái và đầu ra

Hình 4. 12 Khóa và mở rộng khóa

Cấu trúc của thuật toán AES tương đối đơn giản. Cả thuật toán mã hóa và giải mã đều bắt đầu giai đoạn AddRoundKey, tiếp theo là 9 vòng, mỗi vòng đầy đủ 4 giai đoạn: Thay thế các bytes (Substitute bytes) sử dụng hộp S để thực hiện việc thay thế từng byte của khối; dịch các dòng (ShiftRows) đơn giản là thực hiện hoán vị; trộn cột (MixColumns) là phép thay thế sử dụng các phép toán số học trên Z_{256}; AddRoundKey đơn giản chỉ là phép XOR của khối hiện tại với một phần của khóa được mở rộng. Vòng cuối cùng chỉ có 3 giai đoạn. Hình 4.13 minh họa một vòng mã hóa đầy đủ.

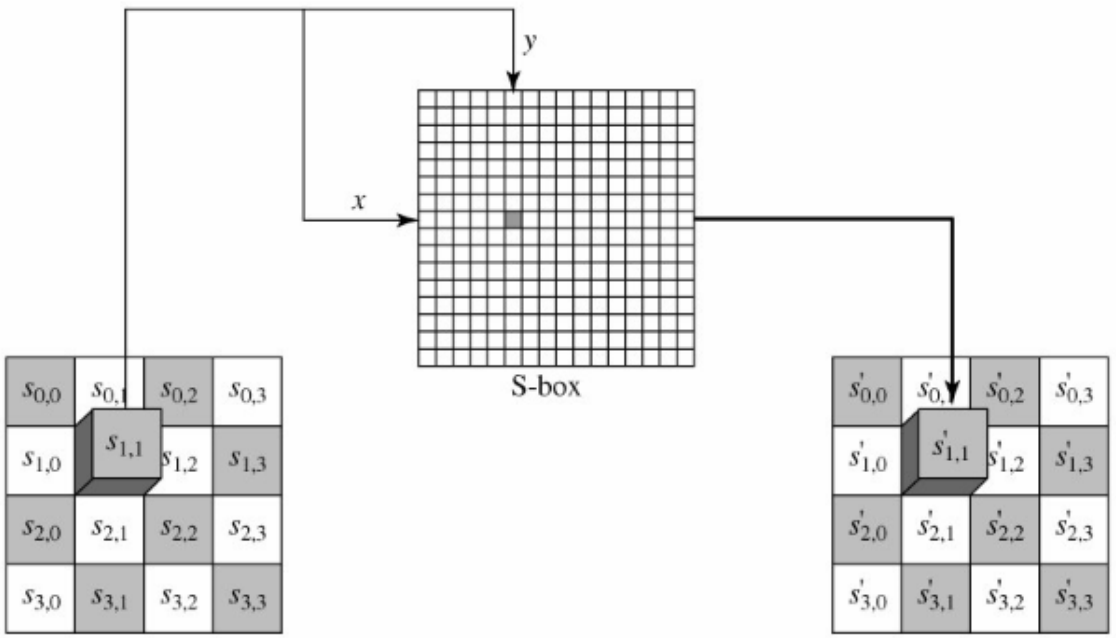


Hình 4. 13 Minh họa một vòng mã AES

Tiếp theo ta đi xét chi tiết các giai đoạn trong một vòng mã hóa.

a) Thay thế byte

Thay thế byte đơn giản chỉ là tra cứu trong bảng 16 x 16, mỗi ô là 1 byte và được gọi là hộp S như bảng 4.12 và hộp S đảo bảng 4.13. Minh họa việc tra cứu hộp S như hình 4.14.



Hình 4. 14 Phép thay thế byte sử dụng hộp S

Bảng 4. 12 Hộp S

		y														
		0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**A**	**B**	**C**	**D**	**E**	**F**
x	**0**	63	7C	77	7B	F2	6B	6F	C5	30	01	67	2B	FE	D7	AB	76
	1	CA	82	C9	7D	FA	59	47	F0	AD	D4	A2	AF	9C	A4	72	C0
	2	B7	FD	93	26	36	3F	F7	CC	34	A5	E5	F1	71	D8	31	15
	3	04	C7	23	C3	18	96	05	9A	07	12	80	E2	EB	27	B2	75
	4	09	83	2C	1A	1B	6E	5A	A0	52	3B	D6	B3	29	E3	2F	84
	5	53	D1	00	ED	20	FC	B1	5B	6A	CB	BE	39	4A	4C	58	CF
	6	D0	EF	AA	FB	43	4D	33	85	45	F9	02	7F	50	3C	9F	A8
	7	51	A3	40	8F	92	9D	38	F5	BC	B6	DA	21	10	FF	F3	D2
	8	CD	0C	13	EC	5F	97	44	17	C4	A7	7E	3D	64	5D	19	73
	9	60	81	4F	DC	22	2A	90	88	46	EE	B8	14	DE	5E	0B	DB
	A	E0	32	3A	0A	49	06	24	5C	C2	D3	AC	62	91	95	E4	79
	B	E7	C8	37	6D	8D	D5	4E	A9	6C	56	F4	EA	65	7A	AE	08
	C	BA	78	25	2E	1C	A6	B4	C6	E8	DD	74	1F	4B	BD	8B	8A
	D	70	3E	B5	66	48	03	F6	0E	61	35	57	B9	86	C1	1D	9E
	E	E1	F8	98	11	69	D9	8E	94	9B	1E	87	E9	CE	55	28	DF
	F	8C	A1	89	0D	BF	E6	42	68	41	99	2D	0F	B0	54	BB	16

Bảng 4. 13 Hộp S đảo (inverse S box)

		y														
		0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**A**	**B**	**C**	**D**	**E**	**F**
x	**0**	52	09	6A	D5	30	36	A5	38	BF	40	A3	9E	81	F3	D7	FB
	1	7C	E3	39	82	9B	2F	FF	87	34	8E	43	44	C4	DE	E9	CB
	2	54	7B	94	32	A6	C2	23	3D	EE	4C	95	0B	42	FA	C3	4E
	3	08	2E	A1	66	28	D9	24	B2	76	5B	A2	49	6D	8B	D1	25
	4	72	F8	F6	64	86	68	98	16	D4	A4	5C	CC	5D	65	B6	92
	5	6C	70	48	50	FD	ED	B9	DA	5E	15	46	57	A7	8D	9D	84
	6	90	D8	AB	00	8C	BC	D3	0A	F7	E4	58	05	B8	B3	45	06
	7	D0	2C	1E	8F	CA	3F	0F	02	C1	AF	BD	03	01	13	8A	6B
	8	3A	91	11	41	4F	67	DC	EA	97	F2	CF	CE	F0	B4	E6	73
	9	96	AC	74	22	E7	AD	35	85	E2	F9	37	E8	1C	75	DF	6E
	A	47	F1	1A	71	1D	29	C5	89	6F	B7	62	0E	AA	18	BE	1B
	B	FC	56	3E	4B	C6	D2	79	20	9A	DB	C0	FE	78	CD	5A	F4
	C	1F	DD	A8	33	88	07	C7	31	B1	12	10	59	27	80	EC	5F
	D	60	51	7F	A9	19	B5	4A	0D	2D	E5	7A	9F	93	C9	9C	EF
	E	A0	E0	3B	4D	AE	2A	F5	B0	C8	EB	BB	3C	83	53	99	61
	F	17	2B	04	7E	BA	77	D6	26	E1	69	14	63	55	21	0C	7D

Ví dụ minh họa phép thay thế byte như hình 4.15

EA	04	65	85
83	45	5D	96
5C	33	98	B0
F0	2D	AD	C5

→

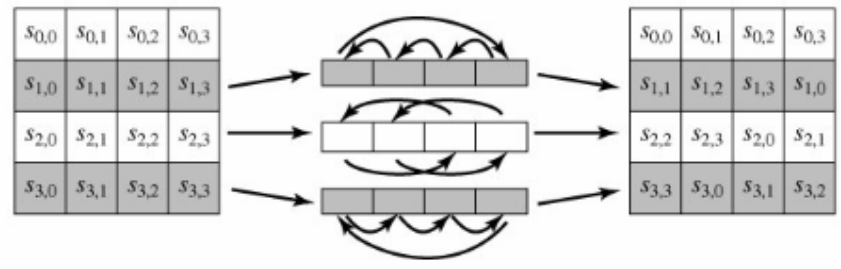
87	F2	4D	97
EC	6E	4C	90
4A	C3	46	E7
8C	D8	95	A6

Hình 4. 15 Minh họa phép thay thế byte

Để tìm byte thay thế của byte EA, ta tra dòng E và cột A trong hộp S thu được byte 87. Như vậy, byte EA sẽ được thay thế bằng byte 87. Tương tự, byte 04 ta tra dòng 0 và cột 4 thu được F2. Ta làm tương tự cho các byte còn lại sẽ thu được ma trận kết quả sau khi thực hiện phép thay thế.

b) Dịch dòng (Shiftrows)

Hình 4.16 minh họa phép dịch dòng. Dòng đầu tiên của ma trận trạng thái được giữ nguyên, dòng thứ hai quay trái 1 byte, dòng thứ 3 quay trái 2 byte và dòng cuối cùng quay trái 3 byte. Hình 4.17 ví dụ minh họa phép dịch dòng.



Hình 4. 16 Minh họa phép dịch dòng

87	F2	4D	97
EC	6E	4C	90
4A	C3	46	E7
8C	D8	95	A6

→

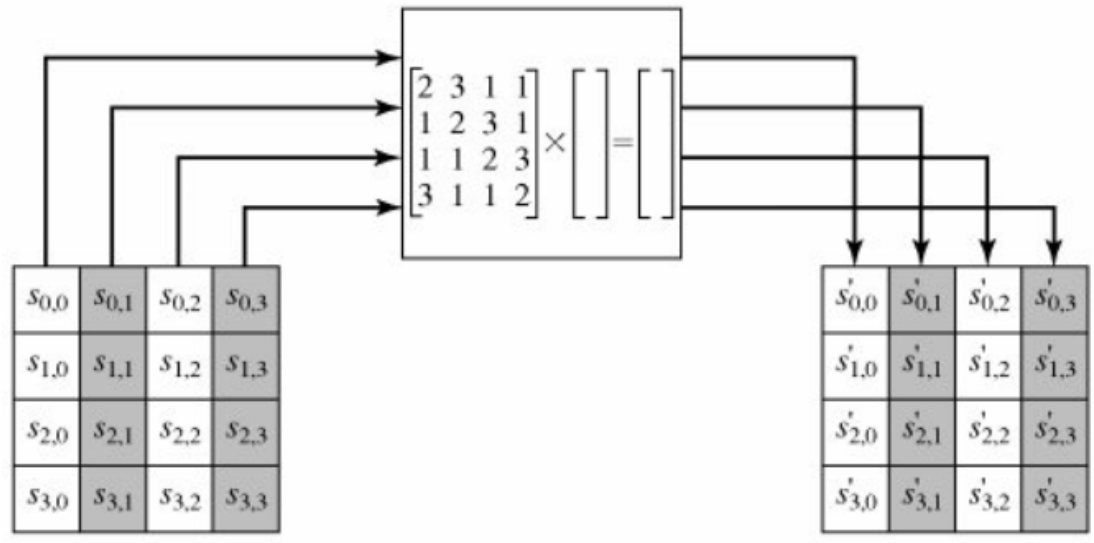
87	F2	4D	97
6E	4C	90	EC
46	E7	4A	C3
A6	8C	D8	95

Hình 4. 17 Ví dụ minh họa phép dịch dòng

Đối với thuật toán giải mã ta sử dụng phép dịch dòng ngược. Tức là, dòng đầu tiên của ma trận trạng thái giữ nguyên, dòng thứ 2 quay phải 1 byte, dòng thứ 3 quay phải 2 bytes và dòng cuối cùng quay phải 3 bytes.

c) Trộn cột

Phép trộn cột được thực hiện như minh họa trên hình 4.18.



Hình 4. 18 Minh họa phép trộn cột

Như vậy, kết quả của phép trộn cột sẽ được xác định theo công thức sau:

$$\begin{bmatrix} s'_{0,0} & s'_{0,1} & s'_{0,2} & s'_{0,3} \\ s'_{1,0} & s'_{1,1} & s'_{1,2} & s'_{1,3} \\ s'_{2,0} & s'_{2,1} & s'_{2,2} & s'_{2,3} \\ s'_{3,0} & s'_{3,1} & s'_{3,2} & s'_{3,3} \end{bmatrix} = \begin{bmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{bmatrix} \times \begin{bmatrix} s_{0,0} & s_{0,1} & s_{0,2} & s_{0,3} \\ s_{1,0} & s_{1,1} & s_{1,2} & s_{1,3} \\ s_{2,0} & s_{2,1} & s_{2,2} & s_{2,3} \\ s_{3,0} & s_{3,1} & s_{3,2} & s_{3,3} \end{bmatrix}$$

Áp dụng phép nhân hai ma trận ta thu được:

$$s'_{0,j} = (2 \cdot s_{0,j}) + (3 \cdot s_{1,j}) + s_{2,j} + s_{3,j}$$
$$s'_{1,j} = s_{0,j} + (2 \cdot s_{1,j}) + (3 \cdot s_{2,j}) + s_{3,j}$$
$$s'_{2,j} = s_{0,j} + s_{1,j} + (2 \cdot s_{2,j}) + (3 \cdot s_{3,j})$$
$$s'_{3,j} = (3 \cdot s_{0,j}) + s_{1,j} + s_{2,j} + (3 \cdot s_{3,j})$$

Trong đó, phép nhân (.) được thực hiện theo luật sau: Giả sử $s_{i,j}$ được biểu diễn dưới dạng 8 bít $b_7b_6b_5b_4b_3b_2b_1b_0$ khi nhân với 2 sẽ được thực hiện theo công thức sau:

$$2.s_{i,j} = \begin{cases} b_6b_5b_4b_3b_2b_1 0 \text{ nếu } b_7 = 0 \\ b_6b_5b_4b_3b_2b_1 0 + 00011011 \text{ nếu } b_7 = 1 \end{cases}$$

$$3 \cdot s_{i,j} = s_{i,j} + 2 \cdot s_{i,j}$$

Phép cộng (+) trong các công thức trên là phép XOR bit.

Hình 4.19 là ví dụ minh họa phép trộn cột.

87	F2	4D	97
6E	4C	90	EC
46	E7	4A	C3
A6	8C	D8	95

→

47	40	A3	4C
37	D4	70	9F
94	E4	3A	42
ED	A5	A6	BC

Hình 4. 19 Minh họa phép trộn cột

Ta diễn giải cách xác định phần tử đầu tiên trong ma trận sau khi thực hiện phép trộn cột. $s'_{0,0} = 2 \cdot (87) + 3.(6E) + 46 + A6$.

Chuyển các số từ hệ 16 sang hệ 2 thu được *87h = 10000111*. Do bít $b_7 = 1$ nên *2.(87) = 00001110 XOR 00011011 = 00010101, 6Eh = 01101110, 46h = 01000110, A6h = 10100110* và *3.(6E) = 6E + 2.(6E)*. Do bít b_7 của *6E* là 0 nên *2.(6E) = 11011100*. Do đó, *3.(6E) = 01101110 XOR 11011100 = 10110010*.

2.(87)	=	0	0	0	1	0	1	0	1
3.(6E)	=	1	0	1	1	0	0	1	0
46	=	0	1	0	0	0	1	1	0
A6	=	1	0	1	0	0	1	1	0
XOR		0	1	0	0	0	1	1	1 = 47h

Tính toán tương tự cho các phần tử còn lại ta thu được trạng thái sau khi thực hiện phép trộn cột.

Phép chuyển đổi đảo trộn cột (inverse mix column transform) trong thuật toán giải mã được thực hiện như sau:

$$\begin{bmatrix} s_{0,0} & s_{0,1} & s_{0,2} & s_{0,3} \\ s_{1,0} & s_{1,1} & s_{1,2} & s_{1,3} \\ s_{2,0} & s_{2,1} & s_{2,2} & s_{2,3} \\ s_{3,0} & s_{3,1} & s_{3,2} & s_{3,3} \end{bmatrix} = \begin{bmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{bmatrix} \times \begin{bmatrix} s'_{0,0} & s'_{0,1} & s'_{0,2} & s'_{0,3} \\ s'_{1,0} & s'_{1,1} & s'_{1,2} & s'_{1,3} \\ s'_{2,0} & s'_{2,1} & s'_{2,2} & s'_{2,3} \\ s'_{3,0} & s'_{3,1} & s'_{3,2} & s'_{3,3} \end{bmatrix}$$

Thay thế công thức của phép trộn cột vào thì ta thu được công thức sau.

$$\begin{bmatrix} s_{0,0} & s_{0,1} & s_{0,2} & s_{0,3} \\ s_{1,0} & s_{1,1} & s_{1,2} & s_{1,3} \\ s_{2,0} & s_{2,1} & s_{2,2} & s_{2,3} \\ s_{3,0} & s_{3,1} & s_{3,2} & s_{3,3} \end{bmatrix}$$

$$= \begin{bmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{bmatrix} \times \begin{bmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{bmatrix} \times \begin{bmatrix} s_{0,0} & s_{0,1} & s_{0,2} & s_{0,3} \\ s_{1,0} & s_{1,1} & s_{1,2} & s_{1,3} \\ s_{2,0} & s_{2,1} & s_{2,2} & s_{2,3} \\ s_{3,0} & s_{3,1} & s_{3,2} & s_{3,3} \end{bmatrix}$$

Như vậy, thì công thứ sau phải được thỏa mãn:

$$\begin{bmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{bmatrix} \times \begin{bmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Ta chứng minh phần tử đầu tiên thỏa mãn yêu cầu. Thật vậy, 2.(0E) + 0B + 0D + 3.(09) = 00011100 + 00001011 + 00001101 + 3.(09). Trong đó, 3.(09) = 09 + 2.(09) = 00001001 XOR 00010010 = 00011011. Cuối cùng ta thu được.

2.(0E)	=	0	0	0	1	1	1	0	0
0B	=	0	0	0	0	1	0	1	1
0D	=	0	0	0	0	1	1	0	1
3.(09)	=	0	0	0	1	1	0	1	1
XOR		0	0	0	0	0	0	0	1 = 1

Các phần tử còn lại người đọc tự chứng mình như là một bài tập.

d) Cộng với khóa (add round key)

Phép cộng với khóa là thực hiện phép XOR bít của 128 bít của ma trận trạng thái và 128 bít của khóa tương ứng của vòng. Hình 4.10 minh họa ví dụ thực hiện phép cộng khóa, ma trận đầu tiên là trạng thái và ma trận thứ 2 là khóa của vòng.

47	40	A3	4C
37	D4	70	9F
94	E4	3A	42
ED	A5	A6	BC

⊕

AC	19	28	57
77	FA	D1	5C
66	DC	29	00
F3	21	41	6A

=

EB	59	8B	1B
40	2E	A1	C3
F2	38	13	42
1E	84	E7	D2

Hình 4. 20 Ví dụ minh họa phép cộng khóa

e) Mở rộng khóa

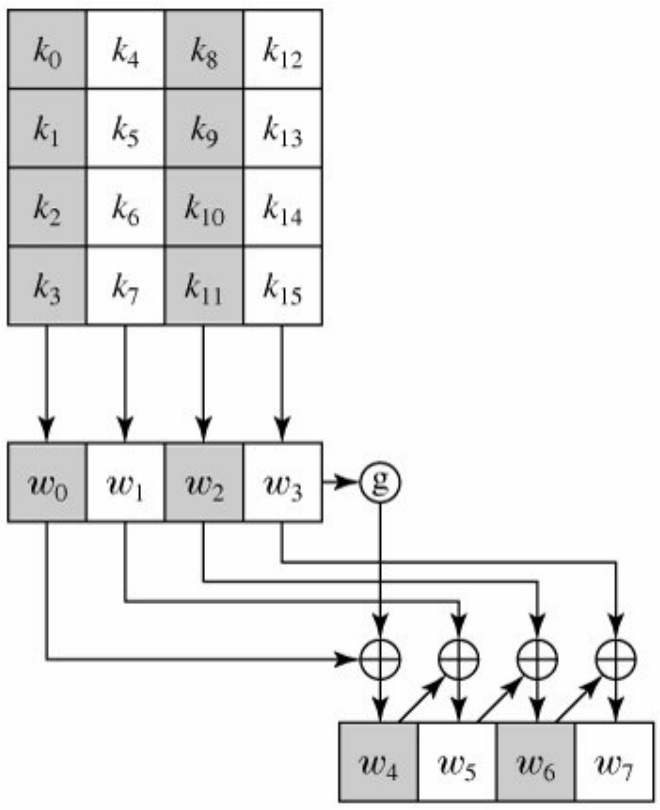
Thuật toán mở rộng khóa có đầu vào là 4 từ (16 bytes) khóa và tạo ra một mảng đầu ra 44 từ (176 bytes). Mã giả của thuật toán được mô tả như sau:

```
KeyExpansion (byte key[16], word w[44])
{
word temp
for(i=0;i<4;i++)
  w[i] = (key[4*i], key[4*i+1], key[4*i+2], key[4*i+3])
for(i=4,i<44;i++)
{
  temp = w[i-1]
  if(i mod 4 = 0)
    temp = SubWord(RotWord(temp)) XOR Rcon[i/4]
  w[i] = w[i-4] XOR temp
}
}
```

Trong đó, phép toán **RotWord** là thực hiện phép quay trái 1 byte, tức là đầu vào l từ có 4 byte [b_0, b_1, b_2, b_3] thì kết quả sau khi thực hiện phép quay trái 1 byte sẽ là [b_1, b_2, b_3, b_0]. Phép toán **SubWord** là phép thay thế byte sử dụng bảng S. Hằng số cho mỗi vòng khóa Rcon[j] = (RC[j], 0, 0, 0), với RC[1] = 1, RC[j] = 2.RC[j-1] và phép nhân (.) được thực hiện theo luật như trong thuật toán trộn cột. Giá trị của RC[j] được xác định như bảng 4.14 ở hệ thập lục phân (hexadecimal).

Bảng 4. 14 Giá trị của RC[j]

j	1	2	3	4	5	6	7	8	9	10
RC[j]	01	02	04	08	10	20	40	80	1B	36



Hình 4. 21 Minh họa cách xác định khóa của vòng 1

Ví dụ minh họa cách xác định khóa cho vòng thứ 9 khi khóa tại vòng 8 là EA D2 73 21 B5 8D BA D2 31 2B F5 60 7F 8D 29 2F tương ứng w[32] = [EA, D2, 73, 21], w[33] = [B5, 8D, BA, D2], w[34] = [31, 2B, F5, 60] và w[35] = [7F, 8D, 29, 2F]. Giá trị của khóa tại vòng 9 được xác định như bảng sau:

Bảng 4. 15 Ví dụ xác định khóa tại vòng 8

Giá trị i ở hệ thập phân	**temp**	**Sau khi thực hiện phép RotWord**	**Sau khi thực hiện phép SubWord**	**Rcon(9)**	**Sau khi XOR với Rcon**	**w[i-4]**	**w[i] = temp XOR w[i-4]**
36	7F8D292F	8D292F7F	5DA515D2	1B000000	46A515D2	EAD27321	AC7766F3
37	AC7766F3	AC7766F3	AC7766F3	1B000000	AC7766F3	B58DBAD2	19FABC21
38	19FABC21	19FABC21	19FABC21	1B000000	19FABC21	312BF560	28B14941
39	28B14941	28B14941	28B14941	1B000000	28B14941	7F8D292F	575C606E

Như vậy, khóa của vòng 9 sẽ là AC 77 66 F3 19 FA BC 21 28 B1 49 41 57 5C 60 6E.

Trong chương này giáo trình đã trình bày các kiến thức liên quan đến thuật toán mã hóa và giải mã chuẩn (DES) và chuẩn tăng cường AES. Đây là hai trong số các thuật toán mã hóa khóa bí mật được sử dụng rộng rãi cho các ứng dụng. Chương tiếp theo, giáo trình sẽ tập trung trình bày thuật toán mã hóa và giải mã của thuật toán mã hóa công khai RSA.

Câu hỏi và bài tập

Bài 1. Xác định kết quả của vòng 1 và 2 của thuật toán mã hóa DES khi mã hóa bản rõ là chuỗi 'ABCDEFGH' với khóa là 'abcdefgh' .

Bài 2. Hãy cài đặt thuật toán mã hóa và giải mã của thuật toán DES bằng ngôn ngữ lập trình C hoặc Java hoặc C#.

Bài 3. Tìm kết quả phép thay thế byte của thuật toán AES cho ma trận trạng thái đầu vào sau:

EA	04	65	85
83	45	5D	96
5C	33	98	B0
F0	2D	AD	C5

Bài 4. Tìm kết quả phép trộn cột của thuật toán AES cho ma trận trạng thái đầu vào sau:

87	F2	4D	97
6E	4C	90	EC
46	E7	4A	C3
A6	8C	D8	95

Bài 5. Tìm khóa cho vòng mã hóa 1 (round 1) của thuật toán AES với khóa ban đầu là A3 49 47 D7 F1 E2 B7 A8 45 46 C0 D4 7C 37 2C 1D.

Bài 6. Viết chương trình cài đặt thuật toán mã hóa và giải mã AES bằng ngôn ngữ lập trình C hoặc Java hoặc C#.

Bài 7. Sử dụng thư viện mã hóa có sẵn của ngôn ngữ lập trình Java hoặc C# hoặc PHP để viết chương trình cho phép chọn một file chứa văn bản rõ và một file chứa khóa sau đó tiến hành mã hóa và giải mã văn bản rõ sử dụng khóa theo thuật toán DES.

Bài 8. Sử dụng thư viện mã hóa có sẵn của ngôn ngữ lập trình Java hoặc C# hoặc PHP để viết chương trình cho phép chọn một file chứa văn bản rõ và một file chứa khóa sau đó tiến hành mã hóa và giải mã văn bản rõ sử dụng khóa theo thuật toán AES.

CHƯƠNG 5. MÃ HÓA CÔNG KHAI VÀ QUẢN LÝ KHÓA

Chương này tập trung trình bày nguyên lý của mã hóa bất đối xứng hay còn gọi là mã hóa khóa công khai, trong đó trình bày chi tiết một số mật mã bất đối xứng được sử dụng rộng rãi đó là RSA và trao đổi khóa Diffie-Hellman.

5.1 Mã khóa công khai

Mã hóa khóa công khai được phát triển để khắc phục hai nhược điểm chính của thuật toán mã hóa đối xứng hay còn gọi là mã khóa bí mật:

- Vấn đề phân phối khóa: Người gửi và người nhận phải chia sẻ khóa cho nhau bằng một cách nào đó hoặc sử dụng một trung tâm phân phối khóa mà trung tâm phân phối khóa này có thể bị tấn công;
- Vấn đề về chữ ký điện tử: Mật mã đã được sử dụng rộng rãi trong thương mại và tư nhân, thì các thông điệp và tài liệu điện tử sẽ cần đến chữ ký tương đương với các tài liệu giấy. Như vậy, người nhận sẽ yên tâm nhận được tài liệu từ người gửi xác định và người gửi không chối bỏ được trách nhiệm đối với tài liệu gửi.

Vào năm 1976 Whitfield Diffie và Martin Hellman đã tìm ra một phương pháp mã hóa khác mà có thể giải quyết được hai vấn đề trên, đó là mã hóa khóa công khai (public key cryptography) hay còn gọi là mã hóa bất đối xứng (asymmetric cryptography). Đây có thể xem là một bước đột phá quan trọng nhất trong lĩnh vực mã hóa.

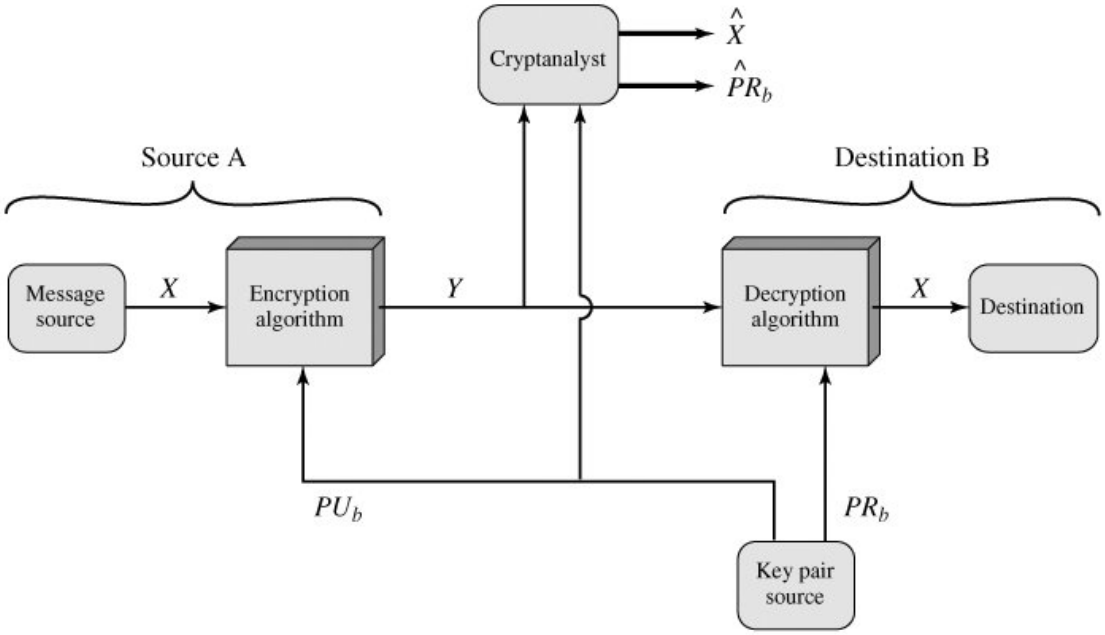
Thuật toán mã hóa khóa công khai sử dụng một khóa để mã hóa và một khóa có liên quan khác để giải mã. Các thuật toán mã hóa này có đặc điểm quan trọng là về mặt tính toán, việc xác định khóa giải mã nếu chỉ biết thuật toán mật mã và khóa dùng để mã hóa là khó khả thi. Ngoài ra, một số thuật toán như RSA thì một trong 2 khóa liên quan có thể được dùng để mã hóa, khóa còn lại được dùng để giải mã.

Lược đồ mã hóa bất đối xứng được minh họa trên hình 5.1, 5.2, 5.3 và bao gồm 6 thành phần chính sau:

- Bản rõ (plaintext): Là thông điệp hoặc dữ liệu có thể đọc được và là đầu vào của thuật toán mã hóa.
- Thuật toán mã hóa (Encryption algorithm): Thực hiện các phép biến đổi khác nhau trên bản rõ.
- Khóa công khai và khóa bí mật (Public and private keys): Đây là một cặp khóa đã được chọn để nếu một khóa được sử dụng để mã hóa thì khóa còn lại được sử dụng để giải mã. Các phép biến đổi được thực hiện bởi thuật

toán phụ thuộc vào khóa công khai hoặc khóa bí mật được cung cấp làm đầu vào.

- Bản mã (Ciphertext): Đây là thông điệp xáo trộn được tạo ra dưới dạng đầu ra. Nó phụ thuộc vào bản rõ và khóa. Đối với một thông điệp nhất định, hai khóa khác nhau sẽ tạo ra hai bản mã khác nhau.
- Thuật toán giải mã (Decryption algorithm): Thuật toán này nhận đầu vào là bản mã và khóa phù hợp để tạo ra bản rõ ban đầu ở đầu ra.



Hình 5. 1 Lược đồ bảo mật dữ liệu

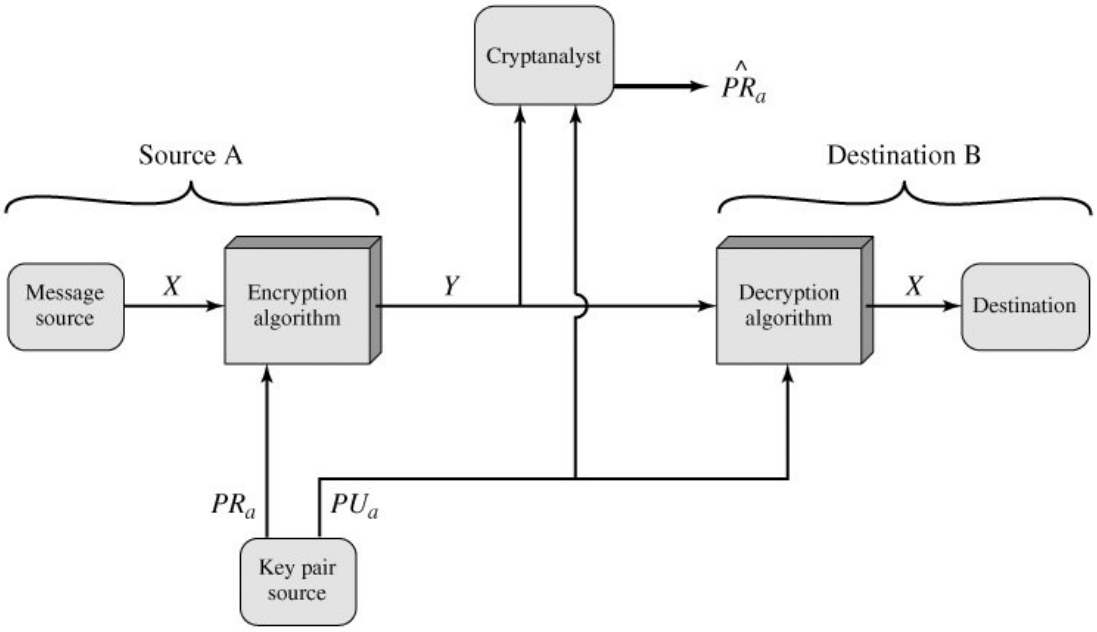
Trong lược đồ bảo mật dữ liệu bên B muốn nhận dữ liệu từ bên A gửi một cách an toàn thì bên B tạo ra cặp khóa công khai ký hiệu là PU_b và khóa bí mật ký hiệu là PR_b. Khóa công khai được bên B công bố công khai nên A có thể nhận được còn khóa bí mật thì B giữ. Bên A gửi thông điệp X cho bên B một cách bảo mật thì sử dụng khóa công khai của B để mã hóa bản rõ theo một thuật toán mã hóa khóa công khai E để thu được bản mã Y. Bên B nhận được bản mã Y sẽ sử dụng thuật toán giải mã D kết hợp với khóa bí mật để khôi phục lại bản rõ X. Quá trình này có thể tóm tắt bằng các phương trình sau:

$$Y = E(PU_b, X)$$

$$X = D(PR_b, Y)$$

Kẻ thám mã cũng có thể có được 2 thông tin là bản mã Y, khóa công khai PU_b, biết thuật toán mã hóa E và giải mã D. Nếu họ chỉ quan tâm đến thông điệp hiện tại thì sẽ tập trung tạo ra một bản ước lượng của X ký hiệu là $\hat{X}$. Tuy nhiên, kẻ thám mã thường

muốn đọc các thông báo trong tương lai nên họ sẽ cố gắng tìm khóa bí mật của bên B ký hiệu là $\widehat{PR_b}$.



Hình 5. 2 Lược đồ xác thực dữ liệu

Trong lược đồ xác thực dữ liệu bên A tạo ra cặp khóa công khai PU_a, khóa bí mật PR_a và mã hóa thông điệp cần gửi cho bên B bằng khóa bí mật của mình, công bố khóa công khai nên bên B có thể nhận được khóa này. Vì thông điệp được mã hóa bằng khóa bí mật của A nên chỉ có A mới có thể tạo được ra thông điệp được mã hóa. Do đó, toàn bộ thông điệp được mã hóa đóng vai trò như chữ ký điện tử. Ngoài ra, không thể thay đổi nội dung của thông điệp nếu không có khóa bí mật của A, nên thông điệp được xác thực cả về nguồn và tính toàn vẹn của dữ liệu. Quá trình này có thể tóm tắt bằng các phương trình sau:

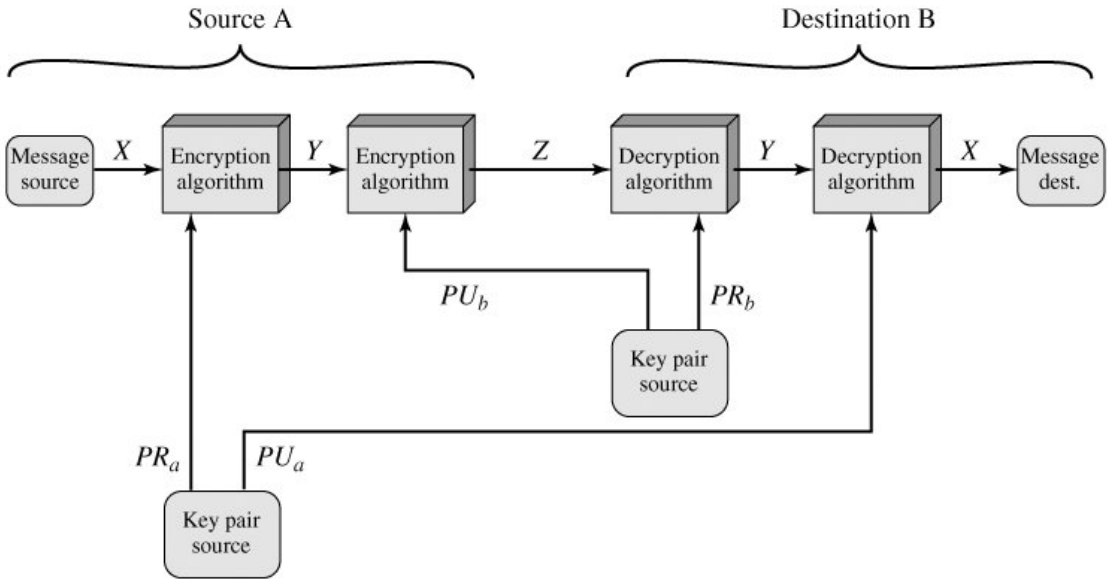
$$Y = E(PR_a, X)$$

$$X = D(PU_a, Y)$$

Kẻ thám mã cũng có thể có được 2 thông tin là bản mã Y, khóa công khai PU_a, biết thuật toán mã hóa E và giải mã D. Do đó họ hoàn toàn có thể biết nội dung của thông điệp hiện tại thông qua thuật toán giải mã với khóa công khai PU_a. Tuy nhiên, mong muốn của họ là tìm được khóa bí mật PR_a nhằm để gửi các thông điệp giả mạo đến B trong tương lai. Do đó, họ cố gắng tìm khóa bí mật của A ký hiệu là $\widehat{PR_a}$.

Lược đồ bảo mật dữ liệu không đảm bảo tính chống chối bỏ bởi vì khóa công khai của bên B (PU_b) được phân phối công cộng nên có nhiều người biết. Do đó bên A có thể chối bỏ thông điệp đã gửi cho bên B vì có thể có nhiều người khác cũng có khả năng gửi thông điệp mã hóa bằng khóa công khai PU_b đến cho B. Lược đồ xác thực dữ liệu

không đảm bảo tính mật của thông điệp được gửi vì nhiều người có khóa công khai của A (PU_a) nên có thể giải mã được thông điệp đã gửi. Do đó, vừa đảm bảo tính bảo mật và chống chối bỏ thì ta cần sử dụng lược đồ kết hợp giữa bảo mật và xác thực dữ liệu.



Hình 5. 3 Lược đồ xác thực và bảo mật dữ liệu

Để bên A gửi cho B thông điệp vừa đảm bảo tính mật và xác thực thì bên A cần có cặp khóa công khai (PU_a) và khóa bí mật PR_a, bên B cũng có cặp khóa công kha PU_b và bí mật PR_b. Cả 2 bên A và B đều công bố khóa công khai của mình, do đó bên A biết khóa công khai của B và ngược lại. Đầu tiên bên A sử dụng khóa bí mật của mình để mã hóa thông điệp X tạo ra thông điệp được mã hóa Y nhằm mục đích xác thực. Tiếp theo, thông điệp Y được mã hóa bằng khóa công khai của B để tạo ra thông điệp Z nhằm mục đích bảo mật, rồi truyền đến bên B. Bên B đầu tiên sẽ giải mã thông điệp Z bằng khóa bí mật của mình để khôi phục lại thông điệp Y. Tiếp theo, thông điệp Y được giải mã một lần nữa bằng khóa công khai của A để khôi phục lại thông điệp ban đầu X. Toàn bộ quá trình này được mô tả bằng các phương trình sau:

$$Y = E(PR_a, X)$$

$$Z = E(PU_b, Y) = E(PU_b, E(PR_a, X))$$

$$Y = D(PR_b, Z)$$

$$X = D(PU_a, Y) = D(PU_a, D(PR_b, Z))$$

Hệ thống mật mã khóa công khai được ứng dụng chủ yếu vào ba nhóm ứng dụng sau:

- Mã hóa/giải mã (Encryption/decryption): Bên gửi sử dụng khóa công khai của bên nhận để mã hóa thông điệp cần gửi.

- Chữ ký số (Digital signature): Bên gửi ký thông điệp bằng mã bí mật. Việc ký được thực hiện bằng một thuật toán mã hóa cho toàn bộ hoặc phân quan trọng của thông điệp.
- Trao đổi khóa (Key exchange): Hai bên phối hợp để trao đổi khóa phiên cho nhau.

Trong thực tế, có những thuật toán mã hóa thực hiện được cả 3 ứng dụng, có những thuật toán chỉ đáp ứng được 1 hoặc 2 ứng dụng. Bảng 5.1 chỉ ra các ứng dụng được hỗ trợ bởi các thuật toán mã hóa thông dụng hiện nay.

Bảng 5. 1 Ứng dụng của hệ thống mật mã khóa công khai

Thuật toán	Mã hóa/giải mã	Chữ ký số	Trao đổi khóa
RSA	Có	Có	Có
Đường cong Elliptic (Đường cong Elliptic)	Có	Có	Có
Diffie-Hellman	Không	Không	Có
DSS	Không	Có	Không

Trong phạm vi của giáo trình chỉ tập trung vào thuật toán RSA và Diffie-Hellman.

Hệ thống mật mã khóa công khai cần đáp ứng các yêu cầu sau:

- Về mặt tính toán, bên B có thể dễ dàng tạo cặp khóa công khai và bí mật PU_b, PR_b.
- Về mặt tính toán, bên A biết khóa công khai của bên B và bản rõ M thì dễ dàng tạo ra bản mã tương ứng: $C = E(PU_b, M)$.
- Bên B dễ dàng giải mã bản mã sử dụng khóa bí mật của mình để khôi phục lại bản rõ ban đầu: $M = D(PR_b, C) = D[PR_b, E(PU_b, M)]$.
- Kẻ thám mã biết mã công khai PU_b và bản mã C muốn xác định khóa bí mật PR_b là không khả thi về mặt tính toán.
- Kẻ thám mã biết mã công khai PU_b và bản mã C muốn xác định bản rõ M là không khả thi về mặt tính toán.

5.2 Hệ mật mã RSA

5.2.1 Giới thiệu

Trong một bài báo của mình hai nhà khoa học Diffie và Hellman đã đề xuất một hướng tiếp cận mới đối với hệ thống mật mã và trên thực tế, thách thức các nhà khoa học mật mã đưa ra một thuật toán mật mã đáp ứng các yêu cầu đối với hệ thống mã công khai. Một thuật toán đáp ứng được thách thức trên đã được đề xuất bởi 3 tác giả là Ron Rivest, Adi Shamir và Len Adleman tại học viện MIT vào năm 1977 và được đặt tên là RSA được ghép từ 3 chữ cái đầu của tên các tác giả. Hệ mật RSA kể từ khi ra đời cho đến này được chấp nhận và triển khai một cách rộng rãi cho hệ thống mật mã khóa công khai. Lược đồ RSA là mã hóa khối, trong đó cả bản rõ và bản mã đều có độ dài là các số nguyên có giá trị trong phạm vi từ 0 đến n – 1 với n nào đó. Kích thước thông dụng cho n là 1024 bit tương đương 309 chữ số thập phân và n nhỏ hơn 2^{1024}.

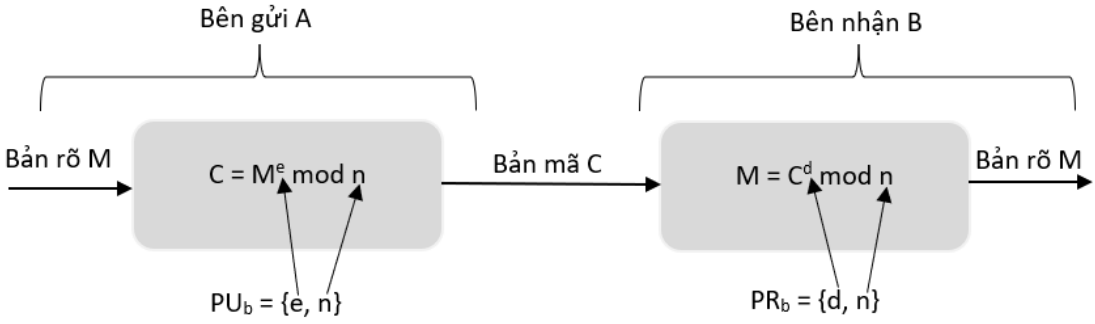
5.2.2 Mô tả thuật toán

Lược đồ mật mã RSA sử dụng biểu thức lũy thừa modulo. Bản rõ được mã hóa thành từng khối và mỗi khối có *i* bít sao cho $2^i < n$ (n là một số nguyên xác định trước), tức là $i < log_2(n)$. Thuật toán RSA được thực hiện qua 3 pha sau:

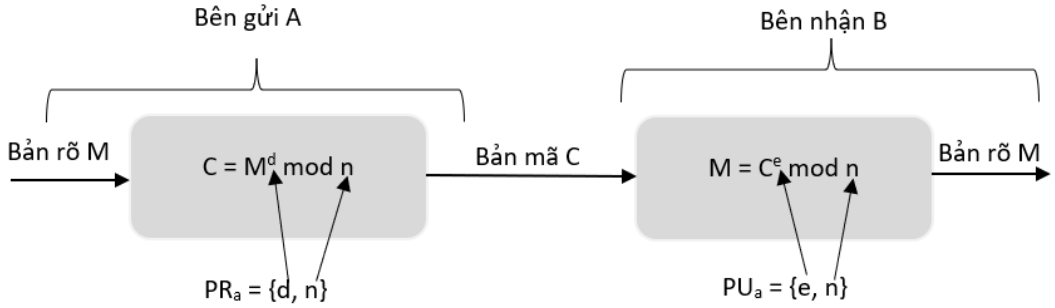
- **Pha 1**: Tạo khóa bao gồm 6 bước sau:
  - ✓ Bước 1: Chọn 2 số nguyên tố lớn p và q khác nhau.
  - ✓ Bước 2: Tính $n = p \times q$
  - ✓ Bước 3: Tính $\phi(n) = (p-1) \times (q-1)$
  - ✓ Bước 4: Lựa chọn số nguyên e sao cho e và $\phi(n)$ là nguyên tố cùng nhau.
  - ✓ Bước 5: Tính d theo công thức $d = e^{-1}\big(mod\ \phi(n)\big)$ (d là nghịch đảo của e modulo $\phi(n)$).
  - ✓ Bước 6: Xác định khóa công khai $PU = \{e, n\}$, khóa bí mật $PR = \{d, n\}$
- **Pha 2**: Mã hóa bản rõ M thành bản mã C. Có 2 khả năng xảy ra.
  - ✓ Khả năng 1: Bảo mật dữ liệu thì bản mã được tạo ra theo công thức sau: $C = M^e mod\ n.$
  - ✓ Khả năng 2: Xác thực dữ liệu thì bản mã được tạo ra theo công thức sau: $C = M^d mod\ n.$
- **Pha 3**: Giải mã bản mã C thành bản rõ M ứng với 2 khả năng.
  - ✓ Khả năng 1: Ứng với trường hợp bảo mật dữ liệu thì bản rõ được khôi phục theo công thức $M = C^d\ mod\ n.$

- ✓ Khả năng 2: Ứng với trường hợp xác thực dữ liệu thì bản rõ được khôi phục theo công thức $M = C^e \ mod \ n$.

Hình 5.4, 5.5 minh họa quá trình mã hóa và giải mã RSA ứng với mục đích bảo mật và xác thực thông tin. Đối với ứng dụng bảo mật bên nhận tạo cặp khóa công khai $PU_b = \{e, n\}$, khóa bí mật $PR_b = \{d, n\}$ và công bố khóa công khai cho bên gửi A. Bên gửi A sử dụng khóa công khai của bên nhận để mã hóa bản rõ thành bản mã, bên nhận sử dụng khóa bí mật của mình để giải mã nhằm khôi phục lại bản rõ ban đầu. Đối với ứng dụng xác thực dữ liệu thì bên gửi A tạo ra cặp khóa công khai $PU_a = \{e, n\}$ và khóa bí mật $PR_a = \{d, n\}$, công bố khóa công khai cho bên nhận B. Bên gửi sẽ mã hóa bản rõ sử dụng khóa bí mật của mình, còn bên nhận sẽ sử dụng khóa công khai của bên gửi để giải mã thu được bản rõ ban đầu.



Hình 5. 4 Mã hóa và giải mã RSA cho ứng dụng bảo mật



Hình 5. 5 Mã hóa và giải mã RSA cho ứng dụng xác thực

Ta minh họa thuật toán RSA thông qua ví dụ cụ thể sau:

- Pha 1: Tạo khóa
  - ✓ Chọn 2 số nguyên tố p = 17 và q = 11
  - ✓ Tính $n = p \times q = 17 \times 11 = 187$
  - ✓ Tính $\phi(n) = (p-1) \times (q-1) = 16 \times 10 = 160$

- ✓ Tìm e sao cho e là số nguyên tố cùng nhau với $\phi(n) = 160$ và nhỏ hơn $\phi(n)$. Ta chọn e = 7.
- ✓ Tìm d sao cho $d.e \equiv 1(mod\ 160)$ và *d < 160*. Ta tìm được *d = 23* vì $d \times e = 23 \times 7 = 161 = 160 + 1$, d có thể được tìm bằng cách sử dụng thuật toán Ơclit mở rộng đã được đề cập trong chương 2.
- ✓ Khóa công khai PU = {e, n} = {7, 187}, khóa bí mật PR = {d, n} = {23, 187}.

- Pha 2: Mã hóa bản rõ M = 88
  - ✓ Trường hợp 1: Bảo mật dữ liệu, ta tính bản mã $C = M^e \bmod n = 88^7 \bmod 187$.

    Để tính ta phân tích $88^7 mod\ 187 = (88^1 \times 88^2 \times 88^4) mod\ 187 = [(88\ mod\ 187) \times (88^2\ mod\ 187) \times (88^4\ mod\ 187)] mod\ 187$.

    88 mod 187 = 88

    88^2 mod 187 = 7747 mod 187 = 77

    88^4 mod 187 = [(88^2 mod 187) x (88^2 mod 187)] mod 187 = (77 x 77) mod 187 = 5929 mod 187 = 132

    Vậy 88^7 mod 187 = (88 x 77 x 132) mod 187 = 894,432 mod 187 = 11

    Do đó, C = 11.
  - ✓ Trường hợp 2: Xác thực dữ liệu, ta xác định bản mã C theo công thức sau $C = M^d \bmod n = 88^{23} \bmod 187$

    Đầu tiên ta phân tích $88^{23} = 88 \times 88^2 \times (88^2)^2 \times (88^4)^2 \times (88^4)^2$. *Như vậy,* $88^{23} \bmod 187 = [(88 \bmod 187) \times (88^2 \bmod 187) \times (88^2 \bmod 187)^2 \bmod 187 \times (88^4 \bmod 187)^2 \bmod 187 \times (88^4 \bmod 187)^2 \bmod 187] \bmod 187$.

    $88 \bmod 187 = 88$

    $88^2 \bmod 187 = 77$

    $88^4 \bmod 187 = 77 \times 77 \bmod 187 = 132$

    $88^8 \bmod 187 = 132 \times 132 \bmod 187 = 17424 \bmod 187 = 33$

    Do đó, $C = 88^{23} \bmod 187 = (88 \times 77 \times 132 \times 33 \times 33) \bmod 187 = 11$
- Pha 3: Giải mã
  - ✓ Trường hợp 1: Bảo mật dữ liệu ta xác định $M = C^d \bmod n = 11^{23} \bmod 187$.

    $11^{23} \bmod 187 = [(11 \bmod 187) \times (11^2 \bmod 187) \times (11^2 \bmod 187)^2 \bmod 187 \times (11^4 \bmod 187)^2 \bmod 187 \times (11^4 \bmod 187)^2 \bmod 187] \bmod 187$.

    $11 \bmod 187 = 11$

$11^2 \bmod 187 = 121$

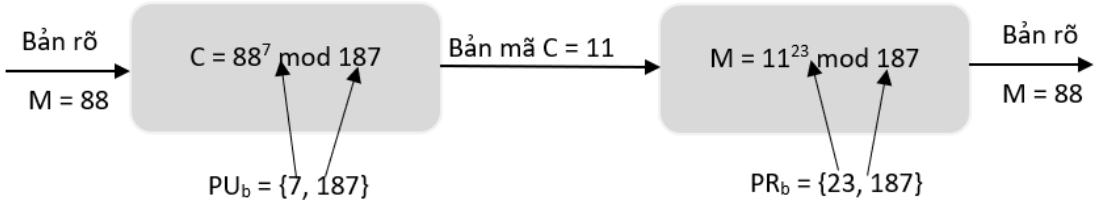
$11^4 \bmod 187 = (121 \times 121) \bmod 187 = 55$

$11^8 \bmod 187 = (55 \times 55) \bmod 187 = 33$

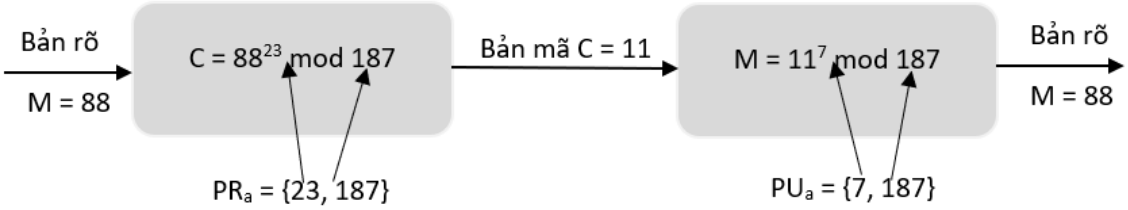
Vậy, $M = 11^{23} \bmod 187 = (11 \times 121 \times 55 \times 33 \times 33) \bmod 187 = 88$

- ✓ Trường hợp 2: Xác thực dữ liệu ta xác định $M = C^e \bmod n = 11^7 \bmod 187$.

  Ta có $11^7 \bmod 187 = [(11 \bmod 187) \times (11^2 \bmod 187) \times (11^2 \bmod 187)^2 \bmod 187] \bmod 187 = (11 \times 121 \times 55) \bmod 187 = 88$.



Hình 5. 6 Ví dụ minh họa mã hóa giải mã RSA cho ứng dụng bảo mật



Hình 5. 7 Ví dụ minh họa mã hóa giải mã RSA cho ứng dụng xác thực

5.3 Quản lý khóa

Một trong những vai trò chính của mã hóa khóa công khai là sử dụng để giải quyết bài toán phân phối khóa. Trên thực tế, có hai khía cạnh khác biệt đối với việc sử dụng mật mã khóa công khai về vấn đề này: Phân phối khóa công khai và sử dụng hệ thống khóa mã hóa khóa công khai để phân phối khóa bí mật.

5.3.1 Phân phối khóa công khai

Một số kỹ thuật được đề xuất để phân phối khóa công khai:

- Thông báo công khai;
- Thẩm quyền khóa công khai;
- Chứng thực khóa công khai.

a) Thông báo công khai

Khi hai người sử dụng muốn truyền dữ liệu với nhau bằng phương pháp mã hóa khóa công khai, trước tiên họ phải trao đổi khóa công khai cho nhau. Vì đây là khóa công khai nên không cần giữ bí mật việc trao đổi này, khóa có thể truyền công khai trên kênh thường như Hình 5.8.

Hình 5. 8 Phân phối khóa công khai một cách tự phát

Phương pháp trao đổi khóa này rất thuận lợi nhưng có một nhược điểm là bất kỳ ai cũng có thể giả mạo người khác để quảng bá khóa công khai của mình. Tức là, một người dùng bất kỳ có thể giả mạo người dùng A và gửi khóa công khai đến cho các người tham gia khác. Cho tới khi người dùng A phát hiện ra hành vi giả mạo và cảnh báo những người tham gia khác, kẻ giả mạo có thể đọc tất cả các tin nhắn được mã hóa dành cho A và có thể sử dụng các khóa giả mạo để xác thực.

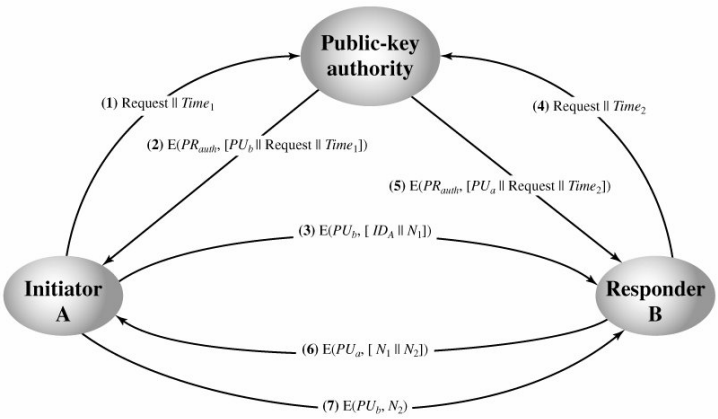
b) Thẩm quyền khóa công khai

Phương pháp trao đổi khóa an toàn chống được sự giả mạo được triển khai như hình 5.9. Trung tâm thẩm quyền khóa công khai duy trì một thư mục động chứa khóa công khai của tất cả các người tham gia. Ngoài ra, mỗi người tham gia đều biết khóa công khai của trung tâm và chỉ trung tâm mới biết khóa bí mật (khóa riêng) tương ứng. Quá trình trao đổi khóa giữa 2 người A và B thông qua trung tâm được diễn ra các bước như sau:

- Bước 1: Bên A gửi một thông điệp có gắn thời gian đến trung tâm thẩm quyền để yêu cầu khóa công khai hiện tại của bên B.
- Bước 2: Trung tâm thẩm quyền trả lời lại cho A một thông điệp được mã hóa bằng khóa bí mật của mình PR_{auth}. Do đó, bên A có thể giải mã được thông điệp này sử dụng khóa công khai của trung tâm. Nên A yên tâm rằng thông điệp có nguồn gốc từ trung tâm và nội dung của thông điệp bao gồm: Khóa công khai của B (PU_b) để A sử dụng mã hóa các thông điệp gửi đến

cho B; Yêu cầu ban đầu để cho phép A đối sánh phản hồi này với yêu cầu tương ứng trước đó và xác minh rằng yêu cầu ban đầu không bị thay đổi trước khi trung tâm có thẩm quyền tiếp nhận; Thời gian ban đầu, vì vậy A có thể xác định rằng đây không phải là một thông báo cũ từ trung tâm có thẩm quyền chứa khóa khác với khóa công khai hiện tại của B.

- Bước 3: A lưu trữ khóa công khai của B và sử dụng nó để mã hóa một thông điệp gửi tới B có chứa số định danh của A (ID_A) và số ngẫu nhiên chỉ sử dụng một lần (N1) để xác định duy nhất giao dịch này.
- Bước 4: Bên B nhận khóa công khai của A từ trung tâm có thẩm quyền tương tự như cách mà bên A nhận khóa công khai của B.
- Bước 5: Tại thời điểm này việc phân phối khóa công khai của A và B đã được thực hiện một cách bảo mật, A, B có thể trao đổi thông tin an toàn cho nhau. Tuy nhiên, cần 2 bước bổ sung sau để đảm bảo quá trình trao đổi giữa A và B diễn ra thực sự an toàn.
- Bước 6: Bên B gửi một thông điệp chứa số ngẫu nhiên N1 nhận được từ A và số ngẫu nhiên N2 do B tạo đến bên A được mã hóa bằng khóa công khai của A (PU_a). Bởi vì chỉ có B có thể giải mã được thông điệp do A gửi ở bước 3. Do đó, sự hiện diện của N1 trong thông điệp ở bước 6 để đảm bảo cho A rằng chính là B.
- Bước 7: A trả lại B thông điệp chứa N2 được mã hóa bằng mã công khai của B (PU_b) để đảm bảo cho B rằng chính là A.



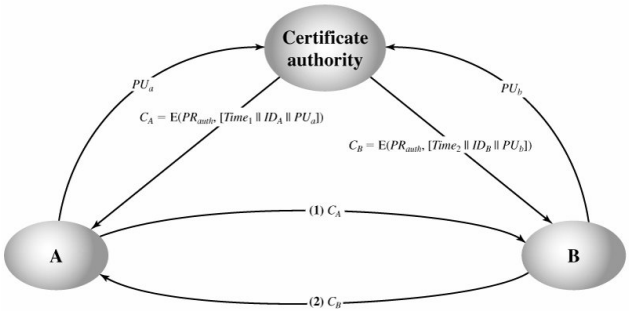
Hình 5. 9 Trao đổi khóa công khai thông qua thẩm quyền khóa công khai

c) Chứng thực khóa công khai (public key certificates)

Phương pháp trao đổi khóa công khai dựa vào thẩm quyền khóa công khai an toàn, chống được sự giả mạo nhưng có nhược điểm là tổ chức quản lý khóa công khai có thể là một nút cổ chai trong hệ thống, bởi vì người dùng phải yêu cầu cơ quan quản lý khóa

công khai cung cấp khóa của mọi người dùng khác mà họ muốn trao đổi thông tin. Để khắc phục nhược điểm này có một phương pháp tiếp cận khác được đề xuất lần đầu bởi Kohnfelder, là sử dụng các chứng thực mà người tham gia có thể sử dụng để trao đổi khóa mà không cần liên hệ với cơ quan quản lý khóa công khai, theo cách đáng tin cậy như thể khóa được lấy trực tiếp từ cơ quan này. Về bản chất, chứng thực bao gồm khóa công khai ghép với số nhận dạng của chủ sở hữu khóa, toàn bộ thông tin này được ký bởi bên thứ ba đáng tin cậy. Thông thường, bên thứ ba là cơ quan cấp chứng thực, chẳng hạn như cơ quan chính phủ hoặc tổ chức tài chính, được cộng đồng người dùng tin cậy. Người dùng có thể gửi khóa công khai của mình cho cơ quan quản lý khóa một cách an toàn và nhận lại chứng thực. Sau đó, người dùng có thể phân phối chứng thực của mình. Bất kỳ ai cần khóa công khai của người dùng này đều có thể nhận chứng thực và xác minh rằng nó hợp lệ bằng chữ ký đáng tin cậy đính kèm. Các bước nhận và phân phối chứng thực được mình họa trên hình 5.10 và các yêu cầu đối với phương pháp trao đổi khóa này như sau:

- Yêu cầu 1: Bất kỳ người tham gia nào cũng có khả năng đọc chứng thực để xác định tên và khóa công khai của chủ sở hữu chứng thực.
- Yêu cầu 2: Bất kỳ người tham gia nào cũng có thể xác minh rằng chứng thực có nguồn gốc từ cơ quan cấp chứng thực và không phải là giả mạo.
- Yêu cầu 3: Chỉ tổ chức phát hành chứng thực mới có thể tạo và cập nhật chứng thực.
- Yêu cầu 4: Bất kỳ người tham gia nào cũng có thể xác minh tính hiện thời của chứng thực.



Hình 5. 10 Mô hình trao đổi khóa công khai sử dụng chứng thực

Đối với bên A, tổ chức quản lý chứng thực cung cấp cho chứng thực dưới dạng C_A $E(PR_{auth}, [T||ID_A||PU_a])$. Trong đó, PR_{auth} là khóa bí mật của tổ chức cấp chứng thực, T là nhãn thời gian để phản ánh tính hiện thời của chứng thực, ID_A là định danh của A và || là phép ghép. Sau đó, A có thể chuyển chứng thực này cho bất kỳ người tham gia nào

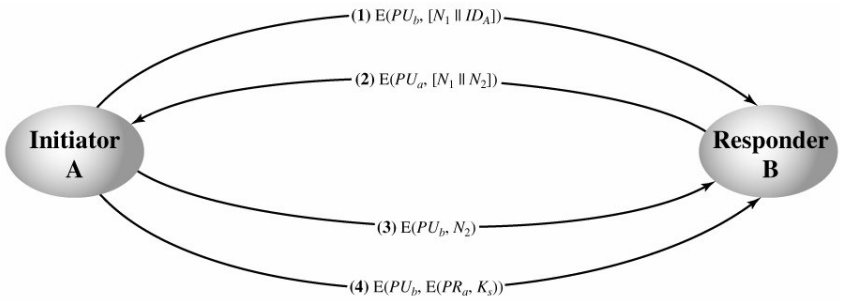
khác, những người này đọc và xác minh chứng thực như sau: $D(PU_{auth}, C_A) = D(PU_{auth}, E(PR_{auth}, [T||ID_A||PU_a])) = (T||ID_A||PU_a)$. Người nhận sử dụng khóa công khai của tổ chức cấp chứng thực, PU_{auth} để giải mã chứng thực. Bởi vì, chứng thực chỉ có thể đọc được bằng cách sử dụng khóa công khai của tổ chức, điều này xác minh rằng chứng thực đến từ tổ chức phát hành chứng thực. Các phần tử ID_A và PU_A cung cấp cho người nhận tên và khóa công khai của chủ sở hữu chứng chỉ. Nhãn thời gian T xác định tính hiện thời của chứng thực.

5.3.2 Phân phối khóa bí mật sử dụng hệ mật mã khóa công khai

Do đặc điểm toán học của phương pháp mã hóa khóa công khai, thời gian mã hóa và giải mã của phương pháp này chậm hơn so với phương án mã hóa đối xứng. Trong thực tế, đối với vấn đề bảo đảm tính bảo mật, người ta vẫn sử dụng phương pháp mã hóa đối xứng. Mã hóa khóa công khai được dùng để thiết lập khóa bí mật cho mỗi phiên trao đổi dữ liệu. Lúc này khóa bí mật được gọi là khóa phiên (session key), các phiên trao đổi dữ liệu khác nhau sẽ dùng các khóa bí mật khác nhau.

Hình 5.11 minh họa trao đổi khóa bị mật sử dụng hệ thống mã hóa khóa công khai. Quá trình trao đổi khóa bí mật được thực hiện qua các bước sau:

- Bước 1: Bên A sử dụng khóa công khai của bên B (PU_b) để mã hóa thông điệp bao gồm một số ngẫu nhiên chỉ sử dụng 1 lần (nonce) N_1 và định danh của A (ID_A).
- Bước 2: Bên B gửi một thông điệp đến bên A bao gồm số ngẫu nhiên sử dụng 1 lần của A (N_1), cùng một số ngẫu nhiên mới được tạo bởi B (N_2) được mã hóa bằng khóa công khai của A (PU_a). Bởi vì chỉ có B mới có thể giải mã được thông điệp (1) do A gửi, sự hiện diện của N_1 trong thông điệp (2) đảm bảo cho A rằng chính là B.
- Bước 3: Bên A trả về cho bên B thông điệp chứa N_2 được mã hóa bằng mã công khai của B (PU_b) để đảm bảo cho B rằng thông điệp này do A gửi.
- Bước 4: Bên A lựa chọn khóa bí mật K_s và gửi thống điệp được mã hóa $M = E(PU_b, E(PR_a, K_s))$ cho B. Mã hóa thông điệp bằng mã công khai của B để đảm bảo rằng chỉ B mới có thể đọc được, mã hóa bằng khóa riêng của A (PR_a) để đảm bảo rằng chỉ có A mới có thể gửi thông điệp này.
- Bước 5: B giải mã để khôi phục lại khóa bí mật K_s như sau: $K_s = D(PU_a, D(PR_b, M))$.



Hình 5. 11 Trao đổi khóa bí mật sử dụng hệ mật mã khóa công khai

5.4 Trao đổi khóa Diffie Hellman

Mục đích của thuật toán là để hai người dùng có thể trao đổi khóa một cách an toàn và khóa này được sử dụng để mã hóa cho các thông điệp trao đổi sau đó. Thuật toán được thực hiện như sau:

Đầu tiên 2 bên sử dụng công khai số nguyên tố q và α là primary root của q. Tiếp theo, bên A chọn một số nguyên ngẫu nhiên X_A < q và tính $Y_A = \alpha^{X_A} \, mod \, q$. Tương tự, bên B chọn một số ngẫu nhiên X_B < q và tính $Y_B = \alpha^{X_B} \, mod \, q$. Cả 2 bên giữ X bí mật và gửi Y công khai cho nhau. Cuối cùng, bên A tính được khóa $K_A = (Y_B)^{X_A} mod \, q$ và B tính được $K_B = \ (Y_A)^{X_B} mod \, q$. Hai giá trị A và B tính được là trùng nhau. Thật vậy:

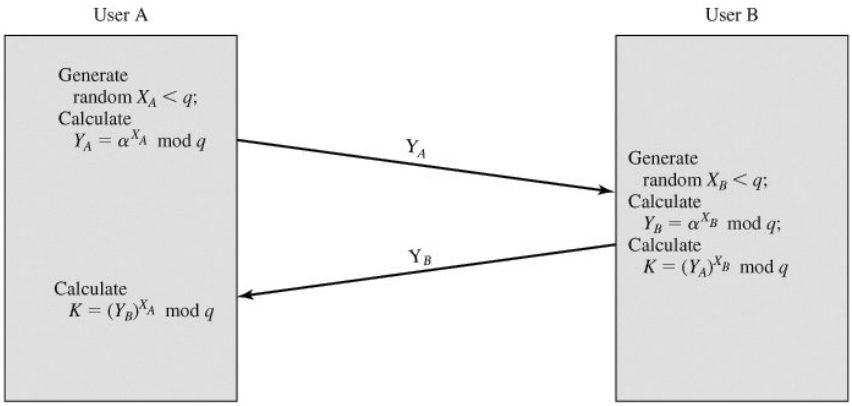
$$K_A = \ (Y_B)^{X_A} mod \, q = (\alpha^{X_B} \, mod \, q)^{X_A} \, mod \, q = (\alpha^{X_B})^{X_A} \, mod \, q = \alpha^{X_A . X_B} mod \, q$$

$$K_B = \ (Y_A)^{X_B} mod \, q = (\alpha^{X_A} \, mod \, q)^{X_B} \, mod \, q = (\alpha^{X_A})^{X_B} \, mod \, q = \alpha^{X_A . X_B} mod \, q$$

Như vậy: $K_A = \ K_B = K = \ \alpha^{X_A . X_B} mod \, q$. Khóa K này có thể sử dụng làm khóa bị mất cho thuật toán mã hóa đối xứng.

Tóm lại, hai bên *A* và *B* đã trao đổi khóa một cách an toàn cho nhau. Bởi vì, kẻ tấn công chỉ có các thông tin sau để tìm khóa: *q, α, Y_A* và *Y_B*. Do đó, kẻ tấn công phải tính lô ga rít rời rạc để tìm khóa. Ví dụ, để xác định khóa bí mật của *B* thì kẻ tấn công cần phải tính $X_B = dlog_{\alpha,q} (Y_B)$. Phép tính lô ga rít rời rạc là không khả thi khi số nguyên tố lớn.

Quá trình trao đổi khóa an toàn theo thuật toán Diffie Hellman được minh họa trên hình 5.12.



Hình 5. 12 Minh họa trao đổi khóa an toàn theo thuật toán Diffie Hellman

Tuy nhiên, phương pháp trao đổi khóa theo thuật toán Diffie Hellman không chống được hình thức tấn công kẻ ở giữa (man in the middle attack). Quá trình tấn công được diễn ra như sau:

- Bước 1: C chuẩn bị tấn công bằng cách tạo ra 2 khóa riêng ngẫu nhiên X_{C1} và X_{C2} và tạo ra 2 khóa công khai Y_{C1} và Y_{C2}.
- Bước 2: Khi A gửi Y_A cho B.
- Bước 3: C chặn Y_A và truyền Y_{C1} cho B và tính khóa $K_2 = (Y_A)^{X_{C2}} mod\ q$.
- Bước 4: B nhận được Y_{C1} và tính $K_1 = (Y_{C1})^{X_B} mod\ q$.
- Bước 5: B truyền Y_B cho A.
- Bước 6: C chặn Y_B và truyền Y_{C2} cho A và tính khóa $K_1 = (Y_B)^{X_{C1}} mod\ q$.
- Bước 7: A nhận Y_{C2} và tính $K_2 = (Y_A)^{X_{C2}} mod\ q$.

Tại thời điểm này, A và B nghĩ rằng họ đã chia sẻ được khóa bí mật nhưng thực tế A và kẻ tấn công C chia sẻ khóa K_2, B và C chia sẽ khóa K_1. Tất cả các trao đổi dữ liệu trong tương lai giữa A và B bị tổn hại theo cách sau:

- A mã hóa thông điệp M: $E(K_2,M)$.
- C chặn thông điệp do A gửi và giải mã để thu được M.
- C có thể gửi thông điệp gốc cho B ($E(K_1,M)$) hoặc gửi thông điệp bất kỳ M' ($E(K_1,M')$. Như vậy, C chỉ muốn xem trộm thông tin hoặc muốn sửa thông tin của A gửi cho B

Trong chương này giáo trình đã trình bày các kiến thức liên quan đến thuật toán mã hóa công khai RSA và trao đổi khóa. Thuật toán RSA là một trong những thuật toán mã hóa khóa công khai được sử dụng rộng rãi trong thực tế. Chương tiếp theo, giáo trình sẽ tập trung trình bày các biện pháp để an toàn IP, web và thư điện tử.

Câu hỏi và bài tập

Bài 1. Chọn p = 13 và q = 17. Hãy xác định khóa bí mật và công khai theo thuật toán RSA.

Bài 2. Hãy thực hiện mã hóa và giải mã theo thuật toán RSA với p = 3, q = 11 và e = 7 cho M = 6 theo hai trường hợp là bảo mật và xác thực.

Bài 3. Người sử dụng A và B sử dụng thuật toán Diffie-Hellman để trao đổi khóa với q = 71 và $\alpha = 7$. Hãy xác định:

a) Y_A = ? khi chọn $X_A = 5$.

b) Y_B = ? khi chọn $X_B = 12$.

c) Khóa bí mật K = ?.

Bài 4. Hãy cài đặt thuật toán mã hóa và giải mã RSA bằng ngôn ngữ lập trình C hoặc Java hoặc C#.

Bài 5. Viết chương trình cài đặt thuật toán trao đổi khóa Diffie-Hellman bằng ngôn ngữ lập trình C hoặc Java hoặc C#.

Bài 6. Sử dụng thư viện mã hóa có sẵn của ngôn ngữ lập trình Java hoặc C# hoặc PHP để viết chương trình cho phép người sử dụng chọn một file chứa văn bản rõ, tiến hành mã hóa và giải mã văn bản rõ sử dụng thuật toán RSA.

CHƯƠNG 6. AN TOÀN IP VÀ WEB

Chương này tập trung trình bày về bảo mật hệ thống mạng ở cấp độ IP, bảo mật website và an toàn thư điện tử. Để đáp ứng được mục đích này, nội dung trình bày chủ yếu tập trung vào các giao thức IPSec, giao thức SSL, giao thức TLS, dịch vụ PGP, dịch vụ S/MIME.

6.1 An toàn IP

Ngày nay, có khá nhiều các cơ chế bảo mật dành riêng cho các ứng dụng trong một số lĩnh vực như: thư điện tử (S/MIME, PGP), máy khách/máy chủ (Kerberos), truy cập web (SSL) và các lĩnh vực khác. Tuy nhiên, vấn đề an toàn giữa các lớp giao thức mạng luôn được nhiều người quan tâm.

Một doanh nghiệp có thể thiết lập một mạng IP riêng để kiểm soát các trang web không đáng tin cậy. Khi triển khai bảo mật hệ thống mạng ở cấp độ IP, doanh nghiệp đó có thể đảm bảo mạng an toàn không chỉ cho các ứng dụng có cơ chế bảo mật mà còn cho nhiều ứng dụng khác.

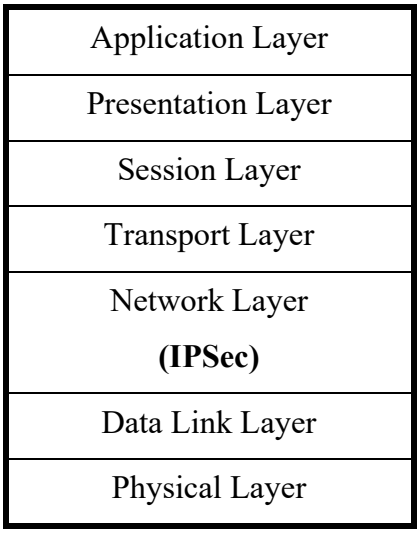
Bảo mật cấp độ IP bao gồm ba chức năng: xác thực, bảo mật và quản lý khóa.

- Chức năng xác thực đảm bảo rằng gói tin nhận được là đúng từ phía người gửi. Ngoài ra, cơ chế này đảm bảo rằng gói tin không bị thay đổi trong quá trình truyền tải.
- Chức năng bảo mật cho phép các nút giao tiếp mã hóa thông báo để ngăn chặn việc nghe trộm bởi các bên thứ ba.
- Chức năng quản lý khóa liên quan đến việc trao đổi khóa an toàn.

Internet là một hệ thống thông tin toàn cầu gồm các mạng máy tính được liên kết với nhau. Hệ thống này truyền thông tin theo kiểu nối chuyển gói dữ liệu (packet switching) dựa trên một giao thức liên mạng đã được chuẩn hóa (giao thức IP). Hệ thống này bao gồm hàng ngàn mạng máy tính nhỏ hơn của các doanh nghiệp, của các viện nghiên cứu và các trường đại học, của người dùng cá nhân và các chính phủ trên toàn cầu... Hiện nay, rất nhiều các cá nhân, tổ chức, cơ quan chính phủ sử dụng các dịch vụ trên mạng Internet như: kinh doanh thương mại điện tử, giao dịch điện tử, chính phủ điện tử… Tuy nhiên, ngoài những tiện ích không thể phủ nhận mà mạng Internet đem lại, người dùng còn đối mặt với hàng loạt các nguy cơ mà trong đó nguy cơ hàng đầu là bị đánh cắp thông tin, thay đổi thông tin truyền tải một cách có chủ đích. Vấn đề đặt ra là làm thế nào để hạn chế nguy cơ mất an toàn trong việc truyền dữ liệu qua mạng và có thể chống lại các cuộc tấn công trong quá trình truyền tải các dữ liệu đó. Để bảo mật các dữ liệu qua mạng Internet thì việc sử dụng giao thức IPSec là một trong những giải pháp hiện nay.

6.1.1. Giới thiệu về giao thức IPSec

IP Security (IPSec – Internet Protocol Security) là một giao thức được chuẩn hoá bởi IETF (Internet Engineering Task Force) từ năm 1998. Giao thức IPSec được xây dựng nhằm mục đích: nâng cấp các cơ chế mã hoá và xác thực thông tin cho chuỗi thông tin truyền đi trên mạng bằng giao thức IP. Hay nói cách khác, IPSec là sự tập hợp của các chuẩn mở được thiết lập để đảm bảo vấn đề bảo mật dữ liệu, tính toàn vẹn dữ liệu và chứng thực dữ liệu giữa các thiết bị mạng.



Hình 6. 1 Giao thức IPSec trong mô hình OSI

Giao thức IPSec cung cấp một cơ cấu bảo mật ở tầng mạng (Network layer) trong mô hình OSI. Các giao tiếp trong một mạng trên cơ sở IP đều dựa trên các giao thức IP. Do đó, khi một cơ chế bảo mật cao được tích hợp với giao thức IP, toàn bộ mạng được bảo mật, bởi vì các giao tiếp đều đi qua tầng mạng trong mô hình OSI.

Giao thức IPSec được thiết kế như phần mở rộng của giao thức IP và được thực hiện trong cả hai phiên bản IPv4 và IPv6. Đối với IPv4, việc áp dụng IPSec là một tùy chọn, nhưng đối với IPv6, giao thức bảo mật này được triển khai bắt buộc.

6.1.2. Đánh giá giao thức IPSec

Giao thức IPSec có những ưu nhược điểm sau:

Ưu điểm:

- IPSec có tính năng an toàn bảo mật cao. Khi IPSec được triển khai trên tường lửa hoặc bộ định tuyến của một mạng riêng thì tính năng an toàn của IPSec có thể áp dụng cho toàn bộ các truy cập vào ra của mạng riêng đó, các thành phần khác không cần phải xử lý thêm các công việc liên quan đến bảo mật.
- IPSec được thực hiện bên dưới lớp TCP và UDP, đồng thời nó hoạt động trong suốt đối với các lớp này. Do vậy, không cần phải thay đổi phần mềm

hay cấu hình lại các dịch vụ khi IPSec được triển khai.

- IPSec có thể được cấu hình để hoạt động một cách trong suốt đối với các ứng dụng đầu cuối, điều này giúp che giấu những chi tiết cấu hình phức tạp mà người dùng phải thực hiện khi kết nối đến mạng nội bộ từ xa thông qua mạng Internet.

Nhược điểm:

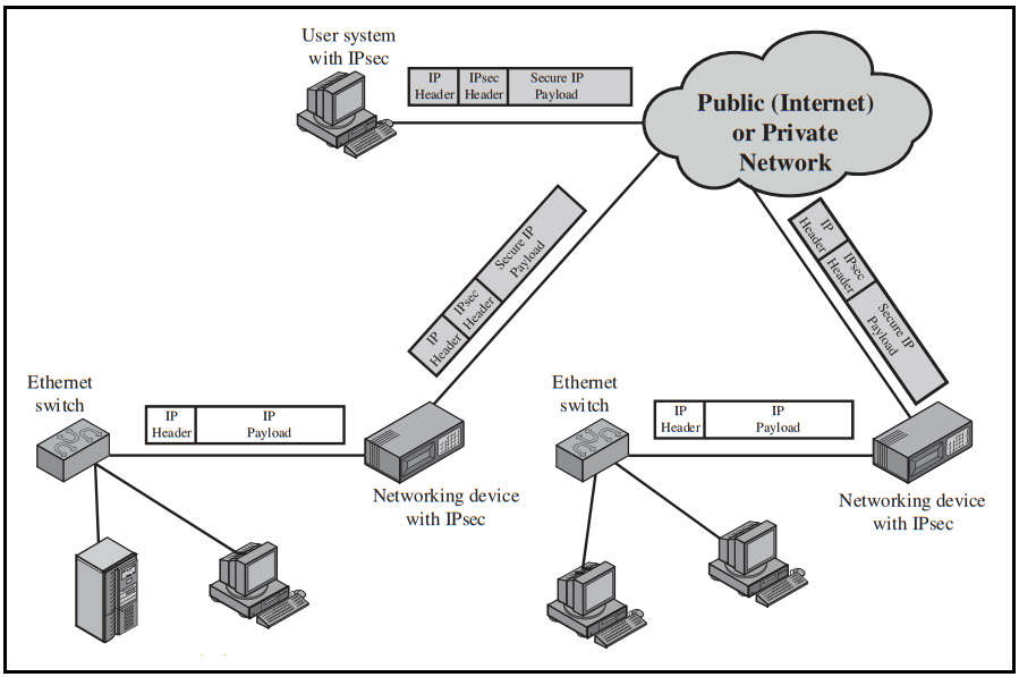
- Tất cả các gói được xử lý theo IPSec sẽ bị tăng kích thước do phải thêm vào các tiêu đề khác nhau, điều này làm cho thông lượng hiệu dụng của mạng giảm xuống. Vấn đề này có thể được khắc phục bằng cách nén dữ liệu trước khi mã hóa, song các kĩ thuật như vậy vẫn còn đang nghiên cứu và chưa được chuẩn hóa.
- IPSec được thiết kế chỉ để hỗ trợ bảo mật cho lưu lượng IP, không hỗ trợ các dạng lưu lượng khác.
- Việc tính toán nhiều giải thuật phức tạp trong IPSec vẫn còn là một vấn đề khó đối với các trạm làm việc và máy tính có cấu hình thấp.
- Việc phân phối các phần cứng và phần mềm mã hóa vẫn còn bị hạn chế đối với chính phủ tại một số quốc gia.

6.1.3. Ứng dụng của giao thức IPSec

Giao thức IPsec cung cấp khả năng bảo mật thông tin liên lạc qua mạng LAN, mạng riêng ảo, mạng WAN và mạng Internet. Cụ thể như sau:

- Kết nối văn phòng chi nhánh an toàn qua Internet: mỗi công ty có thể xây dựng một mạng riêng ảo an toàn qua Internet hoặc qua mạng WAN công cộng. Điều này cho phép một doanh nghiệp phụ thuộc nhiều vào Internet và giảm nhu cầu về mạng riêng, tiết kiệm chi phí và chi phí quản lý mạng.
- Truy cập từ xa an toàn qua Internet: người dùng cuối có hệ thống được trang bị với các giao thức bảo mật IP có thể thực hiện cuộc gọi nội mạng đến nhà cung cấp dịch vụ Internet (ISP) và truy cập an toàn vào mạng của công ty.
- Thiết lập các kết nối mạng extranet và intranet: IPsec có thể được sử dụng để bảo mật thông tin liên lạc với các tổ chức khác, đảm bảo tính xác thực và tính bảo mật cũng như cung cấp cơ chế trao đổi khóa.
- Tăng cường an ninh thương mại điện tử: mặc dù một số trang Web và các ứng dụng thương mại điện tử có các giao thức bảo mật được tích hợp sẵn, nhưng khi sử dụng IPsec sẽ tăng cường khả năng bảo mật đó. IPsec đảm bảo rằng tất cả lưu lượng do quản trị viên mạng chỉ định đều được mã hóa

và xác thực, đồng thời thêm một lớp bảo mật bổ sung cho bất kỳ lưu lượng nào được cung cấp tại lớp ứng dụng.



Hình 6. 2 Ứng dụng giao thức IPSec

Tính năng chính của IPsec cho phép hỗ trợ nhiều loại ứng dụng và có thể mã hóa, xác thực tất cả lưu lượng truy cập ở cấp độ IP. Do đó, tất cả các ứng dụng được đề xuất như: đăng nhập từ xa, máy khách / máy chủ, e-mail, truyền tệp, truy cập Web, ... đều có thể được bảo mật.

6.1.4. Kiến trúc IPSec

Giao thức IPSec được xây dựng dựa trên nền của nhiều kỹ thuật cơ sở khác nhau như mật mã, xác thực, trao đổi khóa…

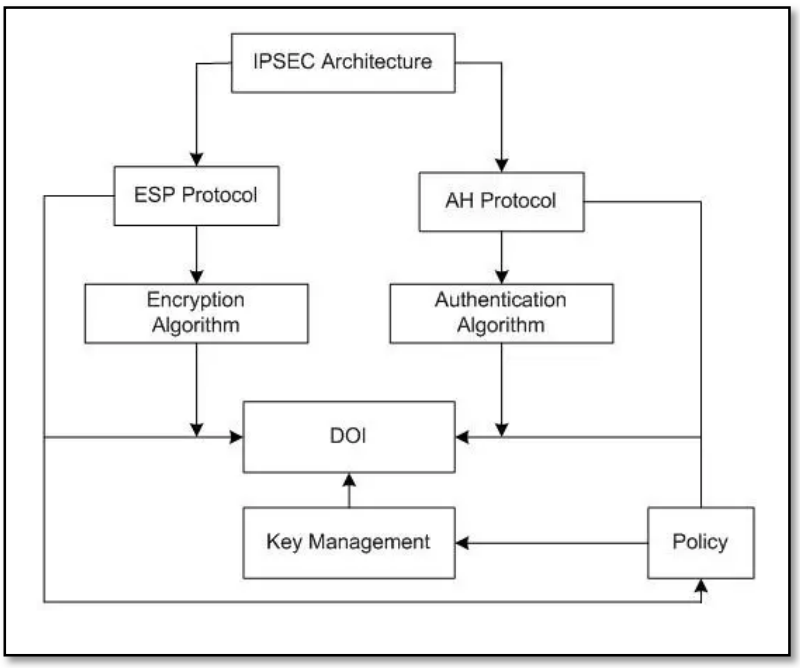
Kiến trúc của giao thức IPSec được xây dựng dựa trên các thành phần cơ bản và được định nghĩa qua một số chuẩn (RFC), bao gồm RFC 2401/2402/2406/2408 và một số chuẩn khác. Kiến trúc này được mô tả trong Hình 6.3.

- Kiến trúc IPSec (RFC 2401): quy định các cấu trúc, các khái niệm và yêu cầu của IPSec.
- Giao thức ESP (RFC 2406, ESP – Encapsulating Security Payload): mô tả giao thức ESP, là một giao thức mật mã và xác thực thông tin trong IPSec. ESP đảm bảo về bảo mật nội dung thông báo và luồng vận chuyển giới hạn, có lựa chọn cung cấp dịch vụ xác thực và hỗ trợ phạm vi rộng các mã, các chế độ mã, bộ đệm, cụ thể:

  + Các thuật toán mã hóa: DES, Triple DES, RC5, IDEA, CAST,…

+ Chuỗi khối mã hóa (Cipher Bock Chaining - CBC) và các chế độ khác.

+ Bộ đệm cần thiết để lấp đầy các kích thước khối, các trường cho luồng vận chuyển.



Hình 6. 3 Kiến trúc giao thức IPSec

- Giao thức AH (RFC 2402, AH – Authentication Header): định nghĩa một giao thức khác với chức năng gần giống ESP. Như vậy khi triển khai IPSec, người sử dụng có thể chọn dùng ESP hoặc AH, mỗi giao thức có ưu và nhược điểm riêng. AH cung cấp sự hỗ trợ cho an toàn dữ liệu và xác thực của các gói IP:

  + Hệ thống đầu cuối/chuyển mạch có thể xác thực người sử dụng/ứng dụng.

  + Ngăn tấn công theo dõi địa chỉ bằng việc theo dõi các chỉ số dãy.

  AH dựa trên sử dụng MAC: HMAC–MD5–96 hoặc HMAC – SHA -1-96. Do đó, các bên cần chia sẻ khóa bí mật.
- Thuật toán mã hóa (Encryption Algorithm): định nghĩa các thuật toán mã hoá và giải mã sử dụng trong IPSec. IPSec chủ yếu dựa vào các thuật toán mã hoá đối xứng.
- Thuật toán xác thực (Authentication Algorithm): định nghĩa các thuật toán xác thực thông tin sử dụng trong AH và ESP.
- Quản lý khoá (RFC 2408): mô tả các cơ chế quản lý và trao đổi khóa trong IPSec.
- Miền thực thi (DOI – Domain of Interpretation): định nghĩa môi trường thực

thi IPSec. IPSec không phải là một công nghệ riêng biệt mà là tổ hợp của nhiều cơ chế, giao thức và kỹ thuật khác nhau, trong đó mỗi giao thức, cơ chế đều có nhiều chế độ hoạt động khác nhau. Việc xác định một tập các chế độ cần thiết để triển khai IPSec trong một tình huống cụ thể là chức năng của miền thực thi. Xét về mặt ứng dụng, IPSec thực chất là một giao thức hoạt động song song với IP nhằm cung cấp hai chức năng cơ bản mà IP nguyên thuỷ chưa có, đó là mã hoá và xác thực gói dữ liệu. Một cách khái quát có thể xem IPSec là một tổ hợp gồm hai thành phần:

+ Giao thức đóng gói, gồm AH và ESP: bảo vệ truyền thông IP, dựa vào SA (khóa, địa chỉ, các thuật toán mật mã).

+ Giao thức trao đổi khóa IKE (Internet Key Exchange): để thiết lập các SA (Security Association) cho AH hoặc ESP và duy trì/quản lý các kết nối.

6.1.5. Tính năng của giao thức IPSec

Giao thức IPSec được cung cấp một số tính năng bảo mật dữ liệu trong VPN, cụ thể như sau:

- Bảo mật dữ liệu (Data Confidentiality): đảm bảo dữ liệu được an toàn, tránh những kẻ tấn công phá hoại bằng cách thay đổi nội dung hoặc đánh cắp dữ liệu quan trọng. Việc bảo vệ dữ liệu được thực hiện bằng các thuật toán mã hóa như DES, 3DES và AES. Tuy nhiên, đây là một tính năng tùy chọn trong IPSec.
- Toàn vẹn dữ liệu (Data Integrity): đảm bảo rằng dữ liệu không bị thay đổi trong suốt quá trình trao đổi. Nó sử dụng thuật toán băm (hash) để kiểm tra dữ liệu bên trong gói tin có bị thay đổi hay không. Những gói tin nào bị phát hiện là đã bị thay đổi thì sẽ bị loại bỏ. Một số thuật toán băm như: MD5, SHA-1.
- Chứng thực nguồn dữ liệu (Data Origin Authentication): mỗi điểm cuối của VPN dùng tính năng này để xác định đầu phía bên kia có thực sự là người muốn kết nối đến mình hay không. Tuy nhiên, tính năng này không tồn tại một mình mà phụ thuộc vào tính năng toàn vẹn dữ liệu. Việc chứng thực dựa vào những kĩ thuật: Pre-shared key, RSA-encryption, RSA-signature.
- Tránh trùng lặp (Anti-replay): đảm bảo gói tin không bị trùng lặp bằng việc đánh số thứ tự. Gói tin nào trùng sẽ bị loại bỏ, đây là tính năng tùy chọn.

6.2 An toàn Web

Hiện nay, hầu hết các cá nhân, doanh nghiệp, các cơ quan tổ chức, chính phủ đều sử dụng các website để phục vụ cho các hoạt động như kinh doanh, giới thiệu các sản phẩm, trao đổi công việc, tin tức … Khi đó, việc cập nhật các thông tin dữ liệu trên các

website ngày càng nhiều, nhưng rủi ro về mất dữ liệu, bị tấn công, đánh cắp thông tin là có thể xảy ra. Để giúp website hoạt động hiệu quả và tránh được những rủi ro như mất dữ liệu, bị tấn công thì các nhà quản trị mạng cần quan tâm hơn đến các phương pháp bảo mật website cũng như các công cụ bảo mật trang web hiệu quả.

Thông thường, các website có các lỗ hổng và đe dọa an toàn như: tính toàn vẹn, tính bảo mật, từ chối dịch vụ và xác thực. Để công tác bảo mật website trở nên hiệu quả hơn, các nhà quản trị web có thể sử dụng các cách thức cài đặt, bổ sung các cơ chế bảo mật cho Web ở lớp vận chuyển (transport layer) như: SSL/TLS, HTTPS và SSH.

6.2.1. Giới thiệu về SSL

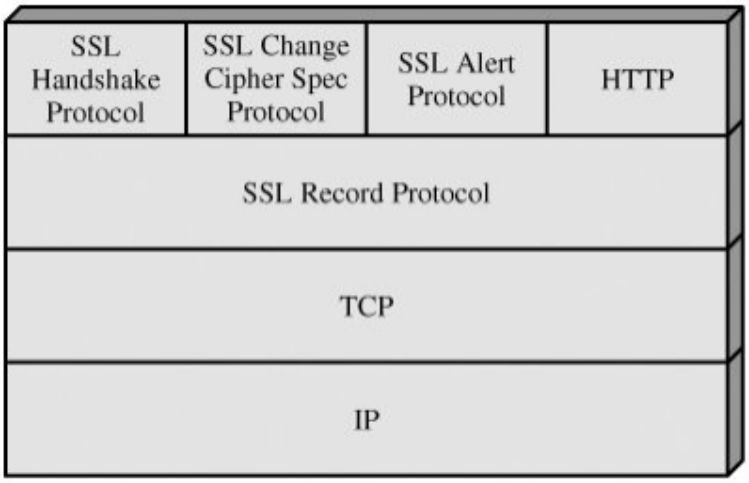
SSL (Secure Socket Layer) là dịch vụ an toàn tầng vận chuyển (transport layer) được phát triển bởi Netscape. SSL cung cấp khả năng mã hóa để bảo mật các kết nối giữa máy khách và máy chủ (thường được định nghĩa là hai hệ thống điện tử tương tác với nhau).

SSL có thể sử dụng để hỗ trợ các giao dịch an toàn cho rất nhiều ứng dụng khác nhau trên Internet. SSL không phải là một giao thức đơn lẻ, mà là một tập các thủ tục đã được chuẩn hoá để thực hiện các nhiệm vụ bảo mật sau:

- Xác thực máy chủ: cho phép người sử dụng xác thực được máy chủ muốn kết nối. Lúc này, phía browser sử dụng các kỹ thuật mã hoá công khai để chắc chắn rằng chứng chỉ và khoá công khai của máy chủ là có giá trị và được cấp phát bởi một CA trong danh sách các CA đáng tin cậy của máy trạm.
- Xác thực máy trạm: cho phép phía máy chủ xác thực được người sử dụng muốn kết nối. Phía máy chủ cũng sử dụng các kỹ thuật mã hoá công khai để kiểm tra xem chứng chỉ và khoá công khai của máy chủ có giá trị hay không và được cấp phát bởi một CA trong danh sách các CA đáng tin cậy không.
- Mã hoá kết nối: tất cả các thông tin trao đổi giữa máy trạm và máy chủ được mã hoá trên đường truyền nhằm nâng cao khả năng bảo mật.

6.2.1.1. Kiến trúc SSL

SSL sử dụng giao thức TCP để cung cấp dịch vụ bảo mật đầu cuối tin cậy. Kiến trúc SSL được đặc tả như sau:



Hình 6. 4 Kiến trúc SSL

- Giao thức bản ghi SSL (SSL Record Protocol): cung cấp hai dịch vụ cho các kết nối SSL:
  - Tính bảo mật: giao thức bắt tay xác định một khóa bí mật được chia sẻ để sử dụng mã hóa. Sử dụng các thuật toán để mã hóa: IDEA, RC2-40, DES-40, DES, 3DES, Fortezza, RC4-40, RC4-128. Bản tin được nén trước khi mã hóa.
  - Tính toàn vẹn của thông báo: Giao thức bắt tay xác định khóa bí mật dùng chung được sử dụng để tạo mã xác thực thông báo (MAC).
- Giao thức thay đổi đặc tả mã SSL (SSL Change Cipher Spec Protocol):

Đây là một trong ba giao thức chuyên biệt của SSL sử dụng thủ tục bản ghi SSL. Thủ tục này bao gồm một thông báo duy nhất, với mục đích làm cho trạng thái đang chờ xử lý được sao chép vào trạng thái hiện tại, cập nhật bộ mật mã thành được sử dụng trên kết nối này.

- Giao thức cảnh báo SSL (SSL Alert Protocol):

Giao thức cảnh báo được sử dụng để chuyển các cảnh báo liên quan đến SSL tới các thành viên. Cũng như các ứng dụng khác sử dụng SSL, thông báo cảnh báo được nén và mã hóa theo trạng thái hiện tại. Mỗi thông báo trong thủ tục này bao gồm hai byte:

- Byte đầu tiên nhận giá trị cảnh báo, hoặc để truyền đạt mức độ nghiêm trọng của thông báo. Nếu mức độ nghiêm trọng, SSL sẽ chấm dứt kết nối ngay lập tức. Các kết nối khác trong cùng một phiên có thể tiếp tục, nhưng không có kết nối mới nào trong phiên này có thể được thiết lập.
- Byte thứ hai chứa một mã cho biết cảnh báo cụ thể như: thông báo lạ, bản ghi MAC lỗi, lỗi giải nén, lỗi bắt tay (Handshake), tham số không hợp lệ. Các cảnh

báo như: không chứng nhận, chứng nhận lỗi, chứng nhận không được hỗ trợ, chứng nhận bị thu hồi, chứng nhận quá hạn, chứng nhận không được biết đến…

- Giao thức bắt tay SSL (SSL HandShake Protocol):

Giao thức này cho phép máy chủ và máy khách: xác thực lẫn nhau, thỏa thuận thuật toán mã hoá và MAC, thỏa thuận khoá mã sẽ dùng để bảo vệ dữ liệu gửi trong bản ghi SSL. Giao thức này được sử dụng trước khi thực hiện truyền dữ liệu đi.

6.2.1.2. Hoạt động của SSL

Giao thức SSL hoạt động dựa trên hai nhóm con giao thức là giao thức "bắt tay" và giao thức "bản ghi". Giao thức "bắt tay" xác định các tham số giao dịch giữa hai đối tượng có nhu cầu trao đổi thông tin hoặc dữ liệu, còn giao thức "bản ghi" xác định khuôn dạng cho tiến hành mã hoá và truyền tin hai chiều giữa hai đối tượng đó. Giao thức SSL "bắt tay" sẽ sử dụng SSL "bản ghi" để trao đổi một số thông tin giữa máy chủ và máy trạm trong lần đầu thiết lập kết nối SSL.

Trong giao dịch SSL, các bước thực hiện trong quá trình "bắt tay" cụ thể như sau:

1. Máy trạm sẽ gửi cho máy chủ số phiên bản SSL đang dùng, các tham số của thuật toán mã hoá, dữ liệu được tạo ra ngẫu nhiên (chữ ký số) và một số thông tin khác mà máy chủ cần để thiết lập kết nối với máy trạm.

2. Máy chủ gửi cho máy trạm số phiên bản SSL đang dùng, các tham số của thuật toán mã hoá, dữ liệu được tạo ra ngẫu nhiên và một số thông tin khác mà máy trạm cần để thiết lập kết nối với máy chủ. Ngoài ra máy chủ cũng gửi chứng chỉ của nó đến máy trạm và yêu cầu chứng chỉ của máy trạm nếu cần.

3. Máy trạm sử dụng một số thông tin mà máy chủ gửi đến để xác thực máy chủ. Nếu như máy chủ không được xác thực thì người sử dụng sẽ được cảnh báo và kết nối không được thiết lập. Còn nếu như xác thực được máy chủ thì phía máy trạm sẽ thực hiện tiếp bước 4.

4. Sử dụng tất cả các thông tin được tạo ra trong giai đoạn bắt tay ở trên, máy trạm (cùng với sự cộng tác của máy chủ và phụ thuộc vào thuật toán được sử dụng) sẽ tạo ra premaster secret cho phiên làm việc, mã hoá bằng khoá công khai mà máy chủ gửi đến trong chứng chỉ ở bước 2, và gửi đến máy chủ.

5. Nếu máy chủ có yêu cầu xác thực máy trạm, thì phía máy trạm sẽ đánh dấu vào phần thông tin riêng chỉ liên quan đến quá trình "bắt tay" này mà hai bên đều biết. Trong trường hợp này, máy trạm sẽ gửi cả thông tin được đánh dấu và chứng chỉ của mình cùng với premaster secret đã được mã hoá tới máy chủ.

6. Máy chủ sẽ xác thực máy trạm. Trường hợp máy trạm không được xác thực,

phiên làm việc sẽ bị ngắt. Còn nếu máy trạm được xác thực thành công, máy chủ sẽ sử dụng khoá bí mật để giải mã premaster secret, sau đó thực hiện một số bước để tạo ra master secret.

7. Máy trạm và máy chủ sẽ sử dụng master secret để tạo ra các khóa phiên, đó chính là các khoá đối xứng được sử dụng để mã hoá và giải mã các thông tin trong phiên làm việc và kiểm tra tính toàn vẹn dữ liệu.

8. Máy trạm sẽ gửi thông báo đến máy chủ thông báo rằng các thông điệp tiếp theo sẽ được mã hoá bằng khoá phiên. Sau đó nó gửi một lời nhắn đã được mã hoá để thông báo rằng phía máy trạm đã kết thúc giai đoạn "bắt tay".

9. Máy chủ gửi lại thông báo đến máy trạm thông báo rằng các thông điệp tiếp theo sẽ được mã hoá bằng khoá phiên. Sau đó nó gửi một lời nhắn đã được mã hoá để thông báo rằng máy chủ đã kết thúc giai đoạn "bắt tay".

10. Lúc này giai đoạn "bắt tay" đã hoàn thành, và phiên làm việc SSL bắt đầu. Cả hai phía máy trạm và máy chủ sẽ sử dụng các khoá phiên để mã hoá và giải mã thông tin trao đổi giữa hai bên, kiểm tra tính toàn vẹn dữ liệu.

6.2.1.3. Phân loại chứng chỉ SSL

Hiện nay, chứng chỉ SSL được sử dụng rất phổ biến trên các website và trình duyệt web tại Việt Nam. Do số lượng website ở Việt Nam hiện nay cũng rất đa dạng nên các phiên bản SSL cũng được cập nhật và sử dụng theo từng dạng website khác nhau. Các loại chứng chỉ SSL được sử dụng ở Việt Nam gồm có:

- Chứng chỉ xác thực tên miền DV-SSL (Domain Validated SSL): DV-SSL dành cho các khách hàng cá nhân với khả năng mã hóa cơ bản với giá rẻ. DV-SSL chỉ yêu cầu xác minh quyền sở hữu tên miền. Thời gian đăng ký và xác minh rất nhanh.
- Chứng chỉ xác thực tổ chức OV-SSL (Organization Validation SSL): OV-SSL dành cho các tổ chức và doanh nghiệp có độ tin cậy cao. Ngoài việc xác minh quyền sở hữu tên miền còn phải xác minh doanh nghiệp đăng ký đang tồn tại và hoạt động bình thường. Tên doanh nghiệp cũng sẽ được hiển thị chi tiết trên chứng chỉ OV được cấp.
- Chứng chỉ xác thực mở rộng EV-SSL (Exented Validation SSL): EV-SSL có độ tin cậy cao nhất chỉ dành cho các tổ chức và doanh nghiệp đang hoạt động. Tuân thủ nghiêm ngặt các quy định của tổ chức CA-Browser Forum trong quá trình xác minh doanh nghiệp. Khi người dùng Internet truy cập vào các website được trang bị chứng chỉ số EV, thanh địa chỉ của trình duyệt sẽ chuyển sang màu xanh lá cây, đồng thời hiển thị tên doanh nghiệp sở hữu website đó. Điều này gia tăng độ tin cậy của website đó đối với người dùng.

- Wildcard SSL:Wildcard SSL dành cho các website có nhu cầu sử dụng SSL cho nhiều subdomain khác nhau. Wildcard SSL khác với các loại SSL bình thường là có thể chạy cho không giới hạn tên miền phụ với một chứng chỉ SSL duy nhất.
- Chứng chỉ UC/SAN SSL: được thiết kế cho các ứng dụng kết nối của Microsoft như Microsoft Exchange Server, Microsoft Office Communications, Lync và cũng là giải pháp tiết kiệm cho các môi trường khác như Share Hosting & QA Testing.

6.2.1.4. An toàn bảo mật khi sử dụng chứng chỉ SSL

Chứng chỉ bảo mật SSL đã đem lại rất nhiều lợi ích về bảo mật cho website và trình duyệt web của người dùng. Cụ thể như sau:

- Xác thực website, giao dịch.
- Nâng cao hình ảnh, thương hiệu và uy tín doanh nghiệp.
- Bảo mật các giao dịch giữa khách hàng và doanh nghiệp, các dịch vụ truy nhập hệ thống.
- Bảo mật webmail và các ứng dụng như Outlook Web Access, Exchange và Office Communication Server.
- Bảo mật các ứng dụng ảo hóa như Citrix Delivery Platform hoặc các ứng dụng điện toán đám mây.
- Bảo mật dịch vụ FTP.
- Bảo mật truy cập control panel.
- Bảo mật các dịch vụ truyền dữ liệu trong mạng nội bộ, file sharing, extranet.
- Bảo mật VPN Access Servers, Citrix Access Gateway …

6.2.2. Giới thiệu về TLS

Giao thức TLS (Transport Layer Security) là một dạng giao thức bảo mật cung cấp mức độ riêng tư cao, cũng như tính toàn vẹn của dữ liệu khi giao tiếp qua mạng và internet. Giao thức TLS là tiêu chuẩn được sử dụng trong việc bảo mật các ứng dụng web và các website trên khắp thế giới. TLS là một bản nâng cấp của SSL, nó là sự kế thừa và thay thế cho hệ thống SSL (Secure Socket Layer) cũ hơn.

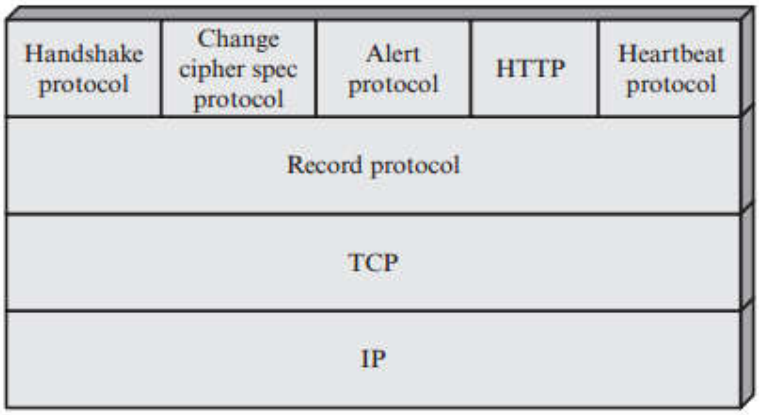
TLS là một dịch vụ đa năng được triển khai dưới dạng một tập hợp các giao thức dựa trên TCP. Giao thức TLS có thể được cung cấp như một phần của bộ giao thức cơ bản, do đó sẽ minh bạch đối với các ứng dụng. Ngoài ra, TLS có thể được nhúng trong các gói cụ ứng dụng thể. Ví dụ, hầu hết các trình duyệt đều được trang bị TLS và hầu hết các máy chủ Web đã triển khai giao thức này.

Giao thức TLS không chỉ được kết hợp phổ biến với các trình duyệt web mà còn được sử dụng cho một số các ứng dụng khác có yêu cầu về bảo mật như: Instant

Messaging, Email, VoIP networks và các ứng dụng khác. Việc sử dụng và phát triển rộng rãi của nó đã cung cấp thêm tính bảo mật, quyền riêng tư và hiệu suất tốt hơn, đặc biệt là kể từ khi phát hành TLS 1.3. Giao thức TLS được sử dụng tích cực trong phần lớn các trình duyệt (Browsers), như được biểu thị bằng biểu tượng ổ khóa, gợi ý mã hóa các trang web bạn đang xem. HTTPS được coi là hình thức trang web được bảo vệ nhiều hơn, nhờ vào việc sử dụng giao thức TLS để bảo vệ người dùng truy cập bất cứ khi nào dữ liệu được trao đổi.

6.2.2.1. Kiến trúc TLS

TLS được thiết kế để sử dụng giao thức TCP cung cấp dịch vụ bảo mật đầu cuối đáng tin cậy. TLS không phải là một giao thức đơn lẻ mà là lớp hai giao thức. Kiến trúc TLS được đặc tả như sau:



Hình 6. 5 Kiến trúc TLS

Về cơ bản, kiến trúc của TLS cũng tương tự như SSL. Tuy nhiên, có bổ sung thêm Heartbeat Protocol.

- Giao thức Heartbeat (Heartbeat Protocol): đưa ra các tín hiệu định kỳ để thông báo về trạng thái hoạt động hoặc đồng bộ hoá các phần khác nhau trong hệ thống. Giao thức Heartbeat được sử dụng để theo dõi tính khả dụng của một giao thức. Trong quá trình hoạt động, giao thức Heartbeat đưa ra hai loại thông báo: heartbeat_request và heartbeat_response.

6.2.2.2. Hoạt động của TLS

TLS bảo mật thông tin liên lạc bằng cách sử dụng cơ sở hạ tầng khóa công khai bất đối xứng để khởi tạo kết nối giữa Client – Server, sau đó sử dụng khóa đối xứng để mã hóa trong phần còn lại của phiên truyền dữ liệu. Quá trình mã hóa được thực hiện như sau:

Giao thức mã hóa bất đối xứng sử dụng hai khóa khác nhau để mã hóa thông tin liên lạc giữa hai bên:

- Khóa bí mật (private key): khóa này do chủ sở hữu trang web kiểm soát và nó được lưu giữ một cách riêng tư. Khóa này nằm trên máy chủ web và được sử dụng để giải mã thông tin được mã hóa bởi khóa công khai.
- Khóa công khai (public key): khóa này khả dụng cho tất cả những người dùng muốn tương tác với máy chủ theo cách an toàn. Thông tin được mã hóa bằng khóa công khai chỉ có thể được giải mã bằng khóa bí mật.

Cặp khóa công khai và khóa bí mật có mối quan hệ được xác định bằng hàm toán học, nhưng được thiết kế để việc tính toán khóa bí mật từ khóa công khai gần như là bất khả thi với độ dài khóa đủ lớn. Thuật toán cũng đảm bảo thông báo được mã hóa bằng khóa công khai không thể giải mã bằng chính khóa công khai mà phải có khóa bí mật (được lưu trữ ở Server). Do đó Client có thể khởi tạo một khóa chung và trao đổi với Server mà không sợ hacker nghe lén khóa công khai và giải mã nó. Khóa chung này sau đó được dùng cho giao thức mã hóa đối xứng.

Với giao thức mã hóa đối xứng (symmetric), dữ liệu được mã hóa và giải mã bằng khóa bí mật mà cả người gửi và người nhận đều biết, thường có độ dài 128 bit, hoặc 256 bit. Mã hóa đối xứng hiệu quả hơn về mặt tính toán so với mã hóa bất đối xứng, nhưng có một khóa bí mật chung có nghĩa là nó cần được chia sẻ một cách an toàn. Đó là lý do ban đầu ta cần sử dụng mã hóa bất đối xứng để trao đổi khóa bí mật nhưng sau đó lại dùng mã hóa đối xứng để trao đổi các dữ liệu lớn.

Quá trình "bắt tay" của TLS được thực hiện như sau:

1. Client gửi một thông báo "client hello" tới Server. Thông báo này bao gồm phiên bản của SSL trên Client, các thiết lập về mật mã, dữ liệu giao dịch và các thông tin cần thiết khác mà Server cần để client có thể giao tiếp thông qua giao thức SSL.

2. Server phản hồi với một thông báo "server hello". Thông báo này cũng bao gồm phiên bản của SSL trên Server, thiết lập mật mã, dữ liệu giao dịch, khóa công khai và các thông tin cần thiết khác mà Client cần để giao tiếp thông qua SSL.

3. Client tiến hành xác nhận SSL certificate (khóa công khai và các thông tin khác mà Server vừa gửi) với CA (Certificate Authority), đây là các đơn vị thứ 3 cung cấp dịch vụ chứng thực số, ví dụ: GeoTrust, Digicert.... Nếu xác nhận thất bại thì Client sẽ từ chối giao tiếp, quá trình bắt tay SSL sẽ bị dừng lại. Nếu xác nhận thành công thì tiếp tục các bước sau.

4. Client sinh một khóa phiên mã hóa nó với khóa công khai và gửi nó đến Server.

5. Server dùng khóa bí mật để giải mã khóa phiên sau đó gửi kết quả thành công về cho Client. Kết quả này cũng sẽ được mã hóa đối xứng với khóa phiên mà vừa được giải mã.

Sau khi việc khởi tạo trên hoàn tất, quá trình giao tiếp còn lại trên các giao thức HTTPS/FTM/STMP,… được mã hóa hai chiều bằng khóa bí mật.

Bên cạnh khả năng mã hóa dữ liệu đầu cuối, TLS còn cho phép ứng dụng khách xác thực quyền sở hữu khóa công khai của máy chủ, để đảm bảo website/máy chủ đang được truy cập tới là chính xác. Điều này thường được thực hiện bằng cách sử dụng chứng chỉ số X.509 được cấp do bên thứ ba đáng tin cậy (được gọi là tổ chức phát hành chứng chỉ - CA) để xác nhận tính xác thực của khóa công khai. Trong một số trường hợp, máy chủ có thể sử dụng chứng chỉ tự ký và nó cần được máy khách chấp nhận một cách rõ ràng (trình duyệt sẽ hiển thị cảnh báo người dùng khi gặp chứng chỉ không đáng tin cậy). Các chứng chỉ riêng như vậy có thể được sử dụng trong các mạng riêng hoặc nơi có thể đảm bảo phân phối chứng chỉ an toàn.

6.2.2.3. Các phiên bản của TLS

Giao thức TLS là giao thức kế thừa từ SSL, trải qua nhiều năm phát triển, giao thức TLS có 4 phiên bản, cụ thể như sau:

- TLS 1.0: được phát hành vào năm 1999, TLS 1.0 có một số điểm tương đồng với SSL 3.0 và chỉ cần nâng cấp hoặc hiện đại hóa các giao thức và quy trình cụ thể để làm cho hệ thống phù hợp hơn với các máy tính và kết nối internet vào đầu những năm 2000. Phiên bản TLS này đã được phát hành theo RFC 2246.
- TLS 1.1: được phát hành vào năm 2006, phiên bản này có một số thay đổi và khác biệt so với phiên bản trước, bao gồm: thay thế IV ngầm định (Implicit IV), hoặc Véc tơ khởi tạo (Initialization Vector) để cung cấp khả năng bảo vệ tốt hơn chống lại cuộc tấn công mạng (Cyber Attacks). Ví dụ, sử dụng chuỗi khối mật mã (Cipher Block Chaining). Phiên bản TLS 1.1 đã thay đổi một số cách xử lý lỗi, các thông tin, hình thức tấn công mạng mới cũng đã được cập. Phiên bản TLS 1.1 được xuất bản theo RFC 4346.
- TLS 1.2: được phát hành vào năm 2008, phiên bản TLS 1.2 cung cấp các cải tiến về bảo mật, cải thiện tốc độ cho máy chủ và máy khách hoạt động thông qua quá trình bắt tay và sử dụng các thuật toán liên quan đến quy trình TLS. Cải tiến quan trọng nhất trong phiên bản TLS 1.2 là việc sử dụng các thuật toán an toàn hơn để dữ liệu tổng thể được đảm bảo an toàn. Phiên bản TLS này đã được xuất bản theo RFC 5246.
- TLS 1.3: được công bố vào năm 2016, tuy nhiên vẫn chưa thay thế hoàn toàn TLS 1.2, phiên bản TLS 1.3 được cập nhật nhằm mục đích cải thiện giao thức bảo mật hiện nay, đảm bảo internet an toàn hơn và phương thức truyền dữ liệu tổng thể an toàn hơn. Sự phát triển mới nhất của TLS 1.3 là để đối phó với các kỹ thuật tấn công ngày càng tinh vi, truy cập vào dữ liệu nhạy cảm, riêng tư trong

các giao dịch, thông tin thanh toán tại ngân hàng. Một số cải tiến của TLS 1.3 như: cho phép độ dài của cookie tăng lên; cải thiện quy trình bắt tay và yêu cầu chữ ký số cho tất cả giao tiếp dữ liệu; hỗ trợ trình duyệt web cho các phiên bản Chrome 56, Chrome 63 và Chrome dành cho Android. Phiên bản TLS 1.3 được xuất bản chính thức từ tháng 8/2018 theo RFC 8466.

6.3 An toàn thư điện tử

Trong hầu như tất cả các môi trường phân tán, thư điện tử là ứng dụng dựa trên mạng được sử dụng nhiều nhất. Người dùng có thể thực hiện gửi email cho những người khác được kết nối trực tiếp hoặc gián tiếp với Internet. Nhu cầu về các dịch vụ xác thực và bảo mật ngày càng tăng khi ngày càng nhiều người sử dụng dịch vụ email.

Để nâng cao độ an toàn thư điện tử, các cá nhân, doanh nghiệp, các tổ chức có thể sử dụng dịch vụ PGP (Pretty Good Privacy) để mã hóa dữ liệu qua mạng. Dịch vụ PGP kết hợp cả quyền riêng tư và xác thực mật mã để mã hóa, giải mã tất cả các loại file qua mạng. Chính sự linh hoạt này, cũng như sức mạnh của PGP trong việc ngăn chặn dữ liệu không thể truy cập trái phép, do đó PGP trở thành tiêu chuẩn mã hóa được sử dụng rộng rãi trên toàn thế giới.

Ngoài ra, người dùng có thể sử dụng tiện ích bảo mật thư điện tử mở rộng đa năng S/MIME (Secure/Multipurpose Internet Mail), đây là một tiêu chuẩn Internet để ký điện tử dữ liệu email dựa trên MIME và mã hóa khóa công khai. Tiện ích này được phát triển bởi RSA Security Inc và dựa trên cơ chế mã hóa khóa công khai của công ty. Hầu hết các dịch vụ và phần mềm email đều sử dụng S/MIME để bảo mật liên lạc qua email.

6.3.1. Dịch vụ PGP

PGP (Pretty Good Privacy) là một dịch vụ về bảo mật và xác thực được sử dụng rộng rãi cho chuẩn an toàn thư điện tử. Phiên bản PGP đầu tiên được phát triển bởi Phil Zimmermann vào năm 1991. PGP được thiết kế để cung cấp tất cả bốn khía cạnh bảo mật bao gồm: quyền riêng tư, tính toàn vẹn, xác thực và tính không từ chối trong việc gửi email.

PGP sử dụng chữ ký số (sự kết hợp của mã hóa băm và khóa công khai) để cung cấp tính toàn vẹn, xác thực và không từ chối. PGP sử dụng kết hợp mã hóa khóa bí mật và mã hóa khóa công khai để cung cấp quyền riêng tư.

PGP là một gói phần mềm mã nguồn mở và miễn phí để bảo mật email. PGP cung cấp xác thực thông qua việc sử dụng chữ ký số, cung cấp tính bảo mật thông qua việc sử dụng mã hóa khối đối xứng.

6.3.1.1. Các dịch vụ sử dụng trong PGP

Hoạt động thực tế của PGP gồm bốn dịch vụ: xác thực, bảo mật, nén và khả năng tương thích với e-mail. Cụ thể như sau:

- *Thao tác PGP – xác thực:*
    1. Người gửi sẽ tạo một thông báo.
    2. Sử dụng SHA-1 để tạo mã băm 160 bit của thông báo.
    3. Mã băm được mã hóa bằng RSA sử dụng mã của người gửi khóa riêng tư và kết quả được thêm vào trước thông báo.
    4. Người nhận sử dụng RSA với khóa công khai của người gửi để giải mã và khôi phục mã băm.
    5. Người nhận tạo một mã băm mới cho thông báo và so sánh nó với mã băm được giải mã. Nếu cả hai khớp nhau, thông báo được chấp nhận như xác thực.
- *Thao tác PGP – bảo mật:*
    1. Người gửi tạo một thông báo và số ngẫu nhiên 128 bit để sử dụng làm khóa phiên cho thông báo.
    2. Thông báo được mã hóa bằng CAST-128 (hoặc IDEA hoặc 3DES) với khóa phiên.
    3. Khóa phiên được mã hóa bằng RSA sử dụng khóa công khai của người nhận và đính kèm với thông báo.
    4. Người nhận sử dụng RSA với khóa riêng để giải mã và khôi phục khóa phiên.
    5. Khóa phiên được sử dụng để giải mã thông báo.
- *Thao tác PGP - Bảo mật và xác thực:*

Có thể sử dụng cả hai dịch vụ trên cùng một thông báo. Trước tiên, một chữ ký được tạo cho thông báo và thêm vào trước thông báo. Sau đó, thông báo cùng với chữ ký được mã hóa bằng CAST-128 (hoặc IDEA hoặc 3DES) và khóa phiên được mã hóa bằng RSA (hoặc ElGamal).

- *Thao tác PGP – nén*

Theo mặc định, PGP nén thông báo sau khi ký nhưng trước khi mã. Điều này có lợi là tiết kiệm không gian cho cả việc truyền email và lưu trữ file. Thuật toán nén được sử dụng là ZIP.

- *Thao tác PGP – tương thích thư điện tử*

Khi PGP được sử dụng, một phần của khối được truyền đi đã được mã hóa. Nếu chỉ dịch vụ chữ ký được sử dụng, thì bản tóm tắt thông báo được mã hóa (với khóa riêng của người gửi). Nếu dịch vụ bảo mật được sử dụng, thông báo cùng với chữ ký (nếu có) sẽ được mã hóa (bằng khóa đối xứng dùng một lần). Do đó, một phần hoặc toàn bộ khối

kết quả bao gồm một luồng các octet 8 bit. Tuy nhiên, nhiều hệ thống thư điện tử chỉ cho phép sử dụng các khối bao gồm văn bản ASCII. Để giải quyết hạn chế này, PGP cung cấp dịch vụ chuyển đổi luồng nhị phân 8 bit thành luồng ký tự ASCII có thể in được.

Lược đồ được sử dụng cho mục đích này là chuyển đổi cơ số 64. Mỗi nhóm gồm 3 octet dữ liệu nhị phân được ánh xạ thành 4 ký tự ASCII, đồng thời gắn thêm CRC để phát hiện lỗi truyền. Khi kích thước đoạn thông báo quá lớn, PGP sẽ chia nhỏ thông báo.

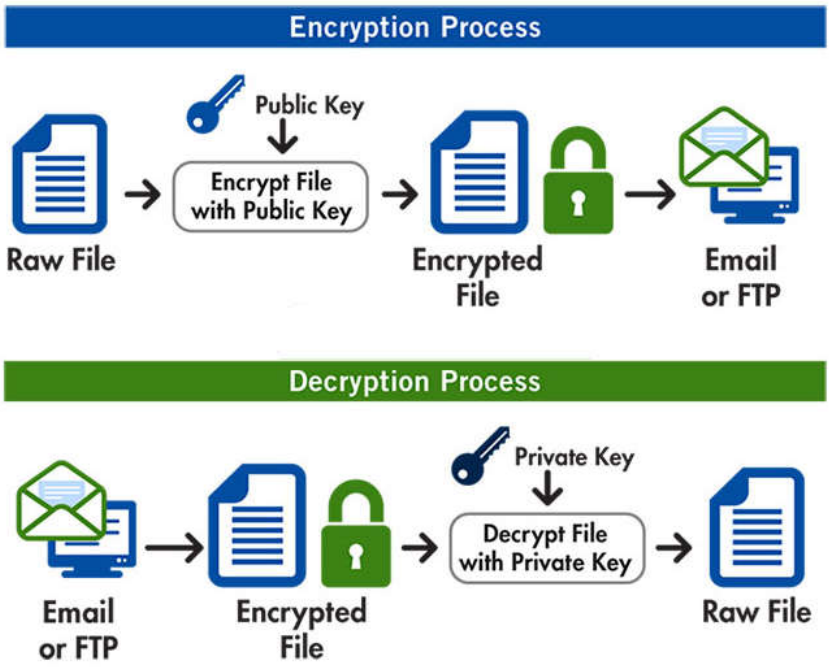
6.3.1.2. Quy trình hoạt động của dịch vụ PGP

PGP hoạt động bằng cách đặt các lớp bảo mật được mã hóa lên trên phần nội dung dựa trên văn bản của ứng dụng.

Trong trường hợp email của máy Client, PGP bảo mật nội dung email bằng thuật toán mã hóa, xáo trộn văn bản theo cách mà nếu bị chặn thì cũng không thể đọc được. Khi nội dung của một email được xáo trộn, khóa tương ứng cần thiết để "mở khóa" mã đó cũng được mã hóa, thông thường sẽ sử dụng khóa công khai (public key) do RSA hoặc Diffie-Hellman cung cấp. Sau đó, email được mã hóa cùng với khóa mã hóa tương ứng được gửi đến người nhận.

Khi gói tin được gửi đến email máy Client của người nhận, ứng dụng sẽ sử dụng khóa riêng (private key) để mở khóa mã hóa email và sau đó giải mã nội dung.

Quy trình hoạt động của dịch vụ PGP được được minh họa trong hình 6.6:



Hình 6. 6 Quy trình hoạt động của PGP

6.3.1.3. Nhược điểm của mã hóa PGP

Mặc dù PGP có nhiều ưu điểm nổi bật, nhưng vẫn tồn tại một số nhược điểm sau:

- Quản lý khó khăn: các phiên bản khác nhau của PGP làm phức tạp thêm việc quản lý.
- Khả năng tương thích: cả người gửi và người nhận đều phải có phiên bản PGP tương thích. Ví dụ: nếu người gửi mã hóa email bằng cách sử dụng PGP với một trong các kỹ thuật mã hóa, người nhận có phiên bản PGP khác không thể đọc dữ liệu.
- Độ phức tạp: PGP là một kỹ thuật phức tạp, các chương trình bảo mật khác sử dụng mã hóa đối xứng một khóa hoặc mã hóa không đối xứng sử dụng hai khóa khác nhau. PGP sử dụng phương pháp kết hợp thực hiện mã hóa đối xứng với hai khóa. PGP phức tạp hơn so với các phương pháp đối xứng hoặc bất đối xứng truyền thống.
- Không có cơ chế khôi phục mật khẩu: quản trị viên máy tính không thể lấy lại mật khẩu bị quên hoặc bị mất. Trong những tình huống như vậy, quản trị viên nên sử dụng một chương trình tiện ích, ứng dụng khác để lấy mật khẩu. Ví dụ: một kỹ thuật viên có quyền truy cập vật lý vào PC có thể được sử dụng để lấy mật khẩu. Tuy nhiên, PGP không cung cấp một chương trình phục hồi đặc biệt như vậy; Phương pháp mã hóa rất mạnh, vì vậy nó không lấy lại mật khẩu bị quên dẫn đến mất tin nhắn hoặc mất tập tin.

6.3.2. Dịch vụ S/MIME

Dịch vụ S/MIME là một tiện ích tăng cường an toàn cho thư điện tử, được nâng cấp lên tiêu chuẩn định dạng thư điện tử Internet MIME dựa trên công nghệ bảo mật dữ liệu của RSA. Mặc dù cả PGP và S/MIME đều theo tiêu chuẩn IETF, nhưng S/MIME được sử dụng trong thương mại và các tổ chức, trong khi PGP vẫn là lựa chọn để bảo mật email cá nhân cho nhiều người dùng. S/MIME sử dụng mật mã khóa công khai để mã hóa và giải mã dữ liệu tin nhắn và chữ ký số. Khóa riêng được giữ an toàn, trong khi khóa chung có thể được phân phối rộng rãi. Một mối quan hệ toán học đặc biệt giữa hai khóa tồn tại sao cho dữ liệu được mã hóa bằng một khóa chỉ có thể được giải mã bởi khóa kia.

Khi người gửi ký một email với S/MIME, phần mềm ứng dụng email của người gửi tạo ra một mã băm hoặc thông báo có độ dài cố định của email, sau đó mã hóa email bằng khóa cá nhân của người gửi để tạo chữ ký điện tử đi kèm với email. Khi người nhận nhận được email, phần mềm email của họ sử dụng khóa công khai người gửi (cũng được bao gồm trong email) để xác minh rằng email thực sự do người gửi đã gửi và nội dung của nó không bị thay đổi khi chuyển tiếp. Nếu một người nào đó chặn được email và thay đổi nội dung, khi đó hàm băm do người nhận tính toán sẽ không khớp với hàm

băm trong chữ ký. Ngoài ra, chữ ký chỉ có thể được tạo bằng khóa cá nhân của người gửi, do đó khi không có khóa sẽ không thể tạo một chữ ký mới hợp lệ để khớp với nội dung đã thay đổi.

S/MIME tăng cường tính an toàn, có hỗ trợ của S/MIME trong nhiều tác nhân thư điện tử như MS Outlook, Mozilla, Mac Mail, …

- Các chức năng S/MIME:

  + Dữ liệu được đóng gói: bao gồm nội dung được mã hóa thuộc bất kỳ loại nào và các khóa mã hóa nội dung được mã hóa cho một hoặc nhiều người nhận.

  + Dữ liệu đã ký: chữ ký điện tử được hình thành dựa trên bản tóm tắt của nội dung được ký và sau đó mã hóa nội dung đó bằng khóa riêng của người ký. Nội dung cộng với chữ ký sau đó được mã hóa bằng cách sử dụng mã hóa base64. Chỉ người nhận có S/MIME mới có thể xem tin nhắn dữ liệu đã ký.

  + Dữ liệu có chữ ký rõ ràng: cũng như dữ liệu đã ký, chữ ký điện tử của nội dung được hình thành. Tuy nhiên, trong trường hợp này, chỉ có chữ ký điện tử được mã hóa bằng base64. Do đó, người nhận không có S/MIME vẫn có khả năng xem nội dung email, mặc dù họ không thể xác thực chữ ký.

  + Dữ liệu đã ký và được đóng gói: chỉ có các thực thể được ký và mã hóa mới có thể được lắp ghép với nhau, do đó, dữ liệu được mã hóa mới có thể được ký và dữ liệu đã ký hoặc dữ liệu được ký mới có thể được mã hóa.

- Các thuật toán mã hoá S/MIME: các chữ ký điện tử DSS và RSA, các hàm hash: SHA-1 và MD5, mã khoá phiên: Elgamal & RSA, mã thông báo: AES, Triple-DES, RC2/40, …;MAC: HMAC với SHA-1. Việc sử dụng thuật toán nào để mã hóa sẽ được quyết định trong quá trình thực hiện.

- Quá trình chứng nhận S/MIME: S/MIME sử dụng chứng nhận X.509 phiên bản 3. Quản trị việc sử dụng kết hợp sơ đồ phân cấp CA của X.509 và Web. Mỗi client có một danh sách các giấy chứng nhận cho CA và các giấy chứng nhận về cặp khoá công khai/bí mật của mình. Chứng nhận này cần được ký bởi các CA tin cậy.

Trong chương này, giáo trình đã trình bày các kiến thức liên quan đến các nội dung về an toàn IP, an toàn Web và an toàn thư điện tử. Các nội dung trình bày sẽ giúp người đọc hiểu rõ hơn về cách thức bảo mật thông tin, đảm bảo an toàn thông tin trong quá trình truyền nhận thông tin trên mạng.

Câu hỏi và bài tập

Câu 1. Nêu mục đích IPSec, các tham số, AH và ESP?

Câu 2. Nêu mục đích SSL và TLS. Trình bày kiến trúc và nhiệm vụ của các thành phần của chúng?

Câu 3. Trình bày giải pháp đề xuất của PGP cho hệ thống thư điện tử?

Câu 4. Trình bày giải pháp đề xuất của S/MIME cho hệ thống thư điện tử?

Câu 5. Tìm hiểu xác thực cơ bản HTTP trong Internet Explorer?

TÀI LIỆU THAM KHẢO

[1]. William Stallings**,** *Cryptography and Network Security Principles and Practices***,** Fourth Edition, Prentice Hall, 2005.

[2]. Đại học Bách khoa Hà Nội, Giáo trình an toàn và bảo mật thông tin.

[3]. Trường Đại học Kinh doanh và Công nghệ Hà Nội, Giáo trình bảo mật thông tin.

[4]. Trường Đại học Nha Trang, Bài giảng an toàn và bảo mật thông tin.