ĐẠI HỌC QUỐC GIA HÀ NỘI
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ
----o0o---

Giáo trình

AN TOÀN DỮ LIỆU

Biên soạn: Trịnh Nhật Tiến

Hà nội 2009

LỜI GIỚI THIỆU

Mục tiêu của An toàn thông tin (Information Security: ATTT) gồm có bảo đảm bí mật thông tin, bảo đảm toàn vẹn (bảo toàn) thông tin với người không được phép truy cập, bảo đảm xác thực nguồn gốc của thông tin, bảo đảm sẵn sàng cung cấp thông tin cho người được phép,…

Để bảo đảm ATTT, người ta sử dụng nhiều công cụ như mã hóa, giấu tin, chữ ký số, thủy vân ký, kiểm soát truy cập thông tin, tìm diệt Virus phá hoại thông tin, ...

Ngày nay người ta giao dịch từ xa qua mạng máy tính, để tránh những sự cố xảy ra khi giao dịch, người ta phải có luật giao dịch điện tử. Tại nhiều nước và nước ta đã có "luật giao dịch điện tử".

Nhờ mạng máy tính và Hệ thống bảo đảm ATTT, tại mọi nơi, mọi lúc, người ta có thể giao dịch với nhau để thực hiện các hoạt động kinh tế xã hội, như mua bán qua mạng, thanh toán qua mạng, đấu thầu từ xa, bỏ phiếu từ xa, giao dịch chứng khoán từ xa,…

An toàn dữ liệu (Data Security) bao gồm việc bảo vệ dữ liệu lưu giữ trong máy tính và bảo vệ dữ liệu đang trên đường truyền tin. Như vậy để tìm hiểu nghiên cứu về An toàn dữ liệu, ta phải tìm hiểu nghiên cứu về An toàn máy tính (Computer Security) và An toàn mạng máy tính (Computer Network Security)

An toàn dữ liệu trong máy tính lại liên quan tới An ninh Hệ điều hành (Operation System Security) và An ninh Cơ sở dữ liệu (Data Base Security)

Tuy nhiên trong khuôn khổ của một giáo trình cho những người bắt đầu tìm hiểu về An toàn dữ liệu, tài liệu này không thể trình bày tất cả các vấn đề liên quan tới việc bảo vệ dữ liệu, mà chỉ trình bày một số kiến thức cơ bản về An toàn dữ liệu trong thời gian 45 tiết (30 tiết lý thuyết + 15 tiết thực hành trên máy tính) Cụ thể tài liệu này sẽ trình bày: tổng quan An toàn dữ liệu, các kiến thức toán học sẽ dùng trong môn học, và một số công cụ chủ yếu được sử dụng trong An toàn dữ liệu.

Lý giải sâu về các công cụ sử dụng trong An toàn dữ liệu, sinh viên tham khảo trong các tài liệu và trong Xemina chuyên đề An toàn thông tin.

Giáo trình này được biên soạn dựa trên các tài liệu tham khảo ghi cuối giáo trình. Công việc của chúng tôi là cấu trúc lại, diễn đạt lại các vấn đề theo cách hiểu của mình và để học sinh dễ tiếp thu, trình bày lại các vấn đề dựa trên kinh nghiệm giảng dạy môn học này cho một số cơ sở đào tạo đại học trong thời gian gần 10 năm. Các ký hiệu và các công thức được ấn loát rõ hơn, dễ hiểu hơn.

Tài liệu này có thể dùng cho sinh viên đại học ngành Công nghệ thông tin (CNTT) hay những người bắt đầu tìm hiểu về lĩnh vực An toàn thông tin (ATTT)

Chắc chắn tài liệu này không trách khỏi những thiếu xót về mặt nội dung cũng như trình bày. Chúng tôi xin nhận được những góp ý của bạn bè và đồng nghiệp.

Xin chân thành cám ơn !

Người biên soạn giáo trình

MỤC LỤC

Chương 1. TỔNG QUAN VỀ AN TOÀN THÔNG TIN

1.1. Vấn đề an toàn thông tin

1.1.1. Tại sao cần Bảo đảm An toàn thông tin ?

Ngày nay, sự xuất hiện Internet và mạng máy tính đã giúp cho việc trao đổi thông tin trở nên nhanh gọn, dễ dàng. E-mail cho phép người ta nhận hay gửi thư ngay trên máy tính của mình, E-business cho phép thực hiện các giao dịch buôn bán trên mạng, ...

Tuy nhiên lại phát sinh những vấn đề mới. Thông tin quan trọng nằm ở kho dữ liệu hay đang trên đường truyền có thể bị trộm cắp, có thể bị làm sai lệch, có thể bị giả mạo. Điều đó có thể ảnh hưởng tới các tổ chức, các công ty hay cả một quốc gia. Những bí mật kinh doanh, tài chính là mục tiêu của các đối thủ cạnh tranh. Những tin tức về an ninh quốc gia là mục tiêu của các tổ chức tình báo trong và ngoài nước.

Theo số liệu của CERT (Computer Emegency Response Team: Đội cấp cứu MT), số lượng các vụ tấn công trên Internet mỗi ngày một nhiều, qui mô của chúng mỗi ngày một lớn và phương pháp tấn công ngày càng hoàn thiện. Ví dụ cùng lúc tin tặc đã tấn công vào cả 100 000 máy tính có mặt trên mạng Internet, những máy tính của các công ty, các trường học, các cơ quan nhà nước, các tổ chức quân sự, các nhà băng, ... cùng lúc ngưng hoạt động.

Khi trao đổi thông tin trên mạng, những tình huống mới nảy sinh: Người ta nhận được một bản tin trên mạng, thì lấy gì làm bảo đảm rằng nó là của đối tác đã gửi cho họ. Khi nhận được tờ Sec điện tử hay Tiền điện tử trên mạng, thì có cách nào để xác nhận rằng nó là của đối tác đã thanh toán cho ta. Tiền đó là tiền thật, hay tiền giả ?

Thông thường, người gửi văn bản quan trọng phải ký phía dưới. Nhưng khi truyền trên mạng, văn bản hay giấy thanh toán có thể bị trộm cắp và phía dưới nó có thể dán một chữ ký khác. Tóm lại với cách thức ký như cũ, chữ ký rất dễ bị giả mạo.

Để giải quyết tình hình trên, vấn đề bảo đảm ***An toàn thông tin*** (**ATTT**) đã được đặt ra trong lý luận cũng như trong thực tiễn.

Thực ra vấn đề này đã có từ ngàn xưa, khi đó nó chỉ có tên là "***bảo mật***", mà kỹ thuật rõ đơn giản, chẳng hạn trước khi truyền thông báo, người gửi và người nhận thỏa thuận một số từ ngữ mà ta quen gọi là tiếng "***lóng***".

Khi có điện tín điện thoại người ta dùng mật mã cổ điển, phương pháp chủ yếu là thay thế hay hoán vị các ký tự trong bản tin "gốc" để được bản tin "mật mã".

Người khác "khó" thể đọc được.

Với sự phát triển mạnh mẽ của Công nghệ thông tin (CNTT), ***An toàn thông tin*** đã trở thành một khoa học thực thụ vì có đất phát triển.

1.1.2. Nội dung lý thuyết về An toàn thông tin.

1.1.2.1. Mục tiêu của An toàn thông tin.

* ***Bảo đảm bí mật*** (Bảo mật):

Thông tin không bị lộ đối với người không được phép.

* ***Bảo đảm toàn vẹn*** (Bảo toàn):

Ngăn chặn hay hạn chế việc bổ sung, loại bỏ và sửa dữ liệu không được phép.

* ***Bảo đảm xác thực*** (Chứng thực):

Xác thực đúng thực thể cần kết nối, giao dịch.

Xác thực đúng thực thể có trách nhiệm về nội dung thông tin (Xác thực nguồn gốc TT)

* ***Bảo đảm sẵn sàng***: Thông tin sẵn sàng cho người dùng hợp pháp.

1.1.2.2. Các nội dung An toàn thông tin.

a) Nội dung chính:

Để bảo vệ thông tin bên trong máy tính hay đang trên đường truyền tin, phải nghiên cứu về An toàn máy tính và An toàn truyền tin.

**** An toàn máy tính*** (Computer Security):

- Là sự bảo vệ các thông tin cố định bên trong máy tính (Static Informations)
- Là khoa học về bảo đảm an toàn thông tin trong máy tính.

**** An toàn truyền tin*** (Communication Security):

là sự bảo vệ thông tin trên đường truyền tin (Dynamic Informations) (Thông tin đang được truyền từ hệ thống này sang hệ thống khác)

là khoa học về bảo đảm an toàn thông tin trên đường truyền tin.

b) Nội dung chuyên ngành (Nội dung hệ quả từ nội dung chính):

Để bảo vệ thông tin bên trong máy tính hay đang trên đường truyền tin, phải nghiên cứu các nội dung chuyên ngành sau:

+ An toàn Dữ liệu (Data Security)

+ An toàn Cơ sở dữ liệu (CSDL) (Data base Security)

+ An toàn Hệ điều hành (Operation system Security)

+ An toàn mạng máy tính (Network Security)

1.1.2.3. Các chiến lược bảo đảm An toàn thông tin.

a) Cấp Quyền hạn tối thiểu ***(****Least Privilege****)***

* Nguyên tắc cơ bản trong an toàn nói chung là "*Hạn chế sự ưu tiên*".

Mỗi đối tượng sử dụng hệ thống (người quản trị mạng, người sử dụng,...) chỉ được cấp phát một số quyền hạn nhất định đủ dùng cho công việc của mình.

b) Phòng thủ theo chiều sâu ***(****Defense in Depth****)***

* Nguyên tắc tiếp theo trong an toàn nói chung là "*Bảo vệ theo chiều sâu*".

Cụ thể là tạo lập *nhiều lớp bảo vệ khác nhau* cho Hệ thống.

Thông tin / / / / /

Access rights Login/ Password Data Encryption Physical protection firewall

1.1.2.4. Các giải pháp bảo đảm An toàn thông tin.

a) Phương pháp che giấu, bảo đảm toàn vẹn và xác thực thông tin.

+ "***Che***" dữ liệu (Mã hóa): thay đổi hình dạng dữ liệu gốc, người khác khó nhận ra.

+ "***Giấu***" dữ liệu: Cất giấu dữ liệu này trong môi trường dữ liệu khác.

+ Bảo đảm toàn vẹn và xác thực thông tin.

Kỹ thuật: Mã hóa, Hàm băm, giấu tin, ký số, thủy ký,..

Giao thức bảo toàn thông tin, **G**iao thức xác thực thông tin, ...

b) Phương pháp kiểm soát lối vào ra của thông tin.

+ Kiểm soát, ngăn chặn các thông tin vào ra Hệ thống máy tính.

+ Kiểm soát, cấp quyền sử dụng các thông tin trong Hệ thống máy tính.

+ Kiểm soát, tìm diệt "sâu bọ" (Virus, "Trojan horse",..) vào ra Hệ thống máy tính.

Kỹ thuật: Mật khẩu (PassWord), Tường lửa (FireWall),

Mạng riêng ảo (Virtual Private Network),

Nhận dạng, Xác thực thực thể, Cấp quyền hạn.

c) Phát hiện và xử lý các lỗ hổng trong An toàn thông tin.

+ Các "lỗ hổng" trong các Thuật toán hay giao thức mật mã, giấu tin.

+ Các "lỗ hổng" trong các Giao thức mạng.

+ Các "lỗ hổng" trong các Hệ điều hành mạng.

+ Các "lỗ hổng" trong các Ứng dụng.

d) Phối hợp các phương pháp.

Xây dựng "hành lang", "đường đi" An toàn cho thông tin gồm 3 phần:

+ Hạ tầng mật mã khóa công khai (Public Key InfraStructure - PKI)

+ Kiểm soát lối vào - ra: Mật khẩu, Tường lửa, Mạng riêng ảo, Cấp quyền hạn.

+ Kiểm soát và Xử lý các lỗ hổng.

1.1.2.5. Các kỹ thuật bảo đảm An toàn thông tin.

+ Kỹ thuật Diệt trừ: VIRUS máy tính, Chương trình trái phép ("Ngựa Troire"),...

+ Kỹ thuật Tường lửa: Ngăn chặn truy cập trái phép, lọc thông tin không hợp phép.

+ Kỹ thuật Mạng riêng ảo: Tạo ra hành lang riêng cho thông tin "đi lại".

+ Kỹ thuật Mật mã: Mã hóa, ký số, các giao thức mật mã, chống chối cãi, …

+ Kỹ thuật giấu tin: Che giấu thông tin trong môi trường dữ liệu khác.

+ Kỹ thuật thủy ký: Bảo vệ bản quyền tài liệu số hóa.

+ Kỹ thuật Truy tìm "Dấu vết" kẻ trộm tin.

1.1.2.6. Các công nghệ bảo đảm An toàn thông tin.

+ Công nghệ chung: Tường lửa, Mạng riêng ảo, PKI, Thẻ thông minh, ..

+ Công nghệ cụ thể: SSL, TLS, PGP, SMINE, ..

1.1.3. Nội dung ứng dụng về An toàn thông tin.

1/ Phục vụ An ninh Quốc phòng: Thám mã, Lọc tin, Bắt trộm, ..

2/ Phục vụ các hoạt động xã hội: Bầu cử, bỏ phiếu từ xa, thăm dò từ xa,..

3/ Phục vụ các hoạt động hành chính: Chính quyền "điện tử".

Chứng minh thư điện tử, giấy phép điện tử,..

Gửi công văn, quyết định, .. từ xa trên mạng máy tính công khai.

4/ Phục vụ các hoạt động kinh tế: Thương mại điện tử.

Thỏa thuận hợp đồng, đấu giá, thanh toán trên mạng máy tính công khai.

Thẻ tín dụng điện tử, Thẻ rút tiền điện tử, Ví tiền điện tử, Tiền điện tử, Sec điện tử,..

5/ Phục vụ các hoạt động giáo dục, đào tạo:

Gửi các đề thi, bài thi qua mạng máy tính công khai, đào tạo từ xa (E-Learning),...

6/ Bảo vệ bản quyền thông tin số hóa: trong bộ nhớ hay trên đường truyền.

1.2. Công cụ đảm bảo an toàn thông tin

1.2.1. Mật mã (Cryptography)

1.2.1.1. Khái niệm Mật mã.

"***Mật mã***" có lẽ là kỹ thuật được dùng lâu đời nhất trong việc bảo đảm "***An toàn thông tin***". Trước đây "*mật mã* " chỉ được dùng trong ngành an ninh quốc phòng, ngày nay việc bảo đảm "***An toàn thông tin***" là nhu cầu của mọi ngành, mọi người (do các thông tin chủ yếu được truyền trên mạng công khai), vì vậy kỹ thuật "*mật mã*" là công khai cho mọi người dùng. Điều bí mật nằm ở "***khóa***" mật mã.

Hiện nay có nhiều kỹ thuật mật mã khác nhau, mỗi kỹ thuật có những ưu, nhược điểm riêng. Tuỳ theo yêu cầu của môi trường ứng dụng mà ta dùng kỹ thuật này hay kỹ thuật khác. Có những môi trường cần phải an toàn tuyệt đốI, bất kể thời gian và chi phí.

Có những môi trường lại cần giải pháp dung hoà giữa bảo mật và chi phí thực hiện.

Mật mã cổ điển chủ yếu dùng để "*che giấu*" dữ liệu. Với ***Mật mã hiện đại***, ngoài khả năng "*che giấu*" dữ liệu, còn dùng để thực hiện: Ký số (ký điện tử), tạo đại diện thông điệp, giao thức bảo toàn dữ liệu, giao thức xác thực thực thể, giao thức xác thực tài liệu, giao thức chứng minh "không tiết lộ thông tin", giao thức thỏa thuận, giao thức phân phối khóa, chống chối cãi trong giao dịch điện tử, giao thức chia sẻ bí mật, ...

Theo nghĩa hẹp, "***mật mã***" chủ yếu dùng để bảo mật dữ liệu, người ta quan niệm:

Mật mã học là Khoa học nghiên cứu mật mã: ***Tạo mã*** và ***Phân tích mã.***

Phân tích mã là kỹ thuật, nghệ thuật phân tích mật mã, kiểm tra tính bảo mật của nó hoặc phá vỡ sự bí mật của nó. Phân tích mã còn gọi là ***Thám mã***.

Theo nghĩa rộng, "***mật mã***" là một trong những công cụ hiệu quả bảo đảm An toàn thông tin nói chung: ***bảo mật, bảo toàn, xác thực, chống chối cãi***, ...

1.2.1.2. Khái niệm mã hóa (Encryption)

**** Mã hóa*** là quá trình chuyển thông tin có thể đọc được (gọi là ***Bản rõ***) thành thông tin

"**khó"** thể đọc được theo cách thông thường (gọi là ***Bản mã***)

Đó là một trong những kỹ thuật để bảo mật thông tin.

**** Giải mã*** là quá trình chuyển thông tin ngược lạI: từ ***Bản mã*** thành ***Bản rõ***.

**** Thuật toán mã hoá*** hay ***giải mã*** là thủ tục tính toán để thực hiện mã hóa hay giải mã.

**** Khóa mã hóa*** là một giá trị làm cho thuật toán mã hoá thực hiện theo cách riêng biệt và sinh ra bản rõ riêng. Thông thường khoá càng lớn thì bản mã càng an toàn. Phạm vi các giá trị có thể có của khoá được gọi là ***Không gian khoá.***

**** Hệ mã hóa*** là tập các thuật toán, các khóa nhằm che giấu thông tin, cũng như làm cho rõ nó.

1.2.1.3. Khái niệm ký số (Digital Signature)

Thông thường sau khi thỏa thuận một văn bản hợp tác, hợp đồng hay thừa nhận trách nhiệm về nội dung một tài liệu, người ta phải xác nhận sự đồng ý của mình bằng cách "***ký tay***" vào cuối văn bản hay tài liệu.

Bằng cách nào đó người ta phải thể hiện đó là "***chữ ký***" của họ (***chữ ký*** bằng "***tay***", **một dấu hiệu riêng** của họ), người khác "***khó thể***" giả mạo (bắt chước) được.

Mọi cách sao chép ***chữ ký*** trên tài liệu thông thường (trên giấy trắng), dễ bị phát hiện, vì bản sao có thể phân biệt được với bản gốc.

Nhưng "**ký**" trên tài liệu trong máy tính hay tài liệu truyền qua mạng máy tính

như thế nào, khi nội dung tài liệu được biểu diễn dưới dạng số hoá (chỉ dùng số 0 và 1), ta gọi là "**tài liệu số**".

Việc giả mạo và sao chép lại đối với "**tài liệu số**" là hoàn toàn dễ dàng, không thể phân biệt được bản gốc với bản sao. Hơn nữa, một tài liệu số có thể bị cắt dán, lắp ghép là hoàn toàn có thể và ta không thể phân biệt được bản gốc với bản sao.

Vậy một **chữ ký** như ***chữ ký*** bằng "***tay***" thông thường ***ở* cuối** "**tài liệu số**", ***không thể*** chịu trách nhiệm đối với toàn bộ nội dung tài liệu.

"Chữ ký" ***như thế nào*** thì mới thể hiện được trách nhiệm đối với toàn bộ "**tài liệu số**" **?**

Chắc chắn **chữ ký** đó phải được "***ký***" trên ***từng bít*** của tài liệu số**.**

Vậy "***ký***" trên tài liệu số được thực hiện như thế nào ?

Thực chất của việc "***ký số***" trên "**tài liệu số**" là "***mã hoá*** " "**tài liệu số**" đó.

Bản mã chính là **"Chữ ký *số*"** hay "***Chữ ký điện tử***" (Digital Signature)

Xác nhận "**chữ ký**" là ***kiểm tra*** việc mã hoá trên có đúng không.

Như vậy khi gửi một tài liệu số có chữ ký trên đó, người ta phải gửi cả 2 file: một file tài liệu và một file chữ ký. Nhờ đó mới kiểm tra được có đúng chữ ký đó ký trên tài liệu đi kèm hay không. Chúng ta sẽ hiểu rõ vấn đề này trong chương chữ ký số.

1.2.2. Giấu tin (Steganography)

1.2.2.1. Khái niệm Giấu tin.

Mã hoá thông tin là biến đổi thông tin "dễ hiểu" (hiển thị rõ ràng, có thể đọc được, có thể hiểu được) thành thông tin dưới dạng "bí mật" (khó thể hiểu được vì chỉ nhìn thấy những kí hiệu rời rạc vô nghĩa)

Thông tin mã hóa dễ bị phát hiện, vì chúng có hình dạng đặc biệt. Khi đó tin tặc sẽ tìm mọi cách để xác định bản rõ.

Giấu thông tin (Steganography) là giấu thông tin này vào trong một thông tin khác.

Thông tin được giấu (nhúng) vào bên trong một thông tin khác, sẽ khó bị phát hiện, vì người ta khó nhận biết được là đã có một thông tin được ***giấu*** (nhúng) vào bên trong một thông tin khác (gọi là ***môi trường giấu tin***)

Nói cách khác, giấu tin giống như "***ngụy trang***" cho thông tin, không gây ra cho tin tặc sự nghi ngờ. Ví dụ một thông tin ***giấu*** vào bên trong một bức tranh, thì **sự vô hình** của thông tin chứa trong bức tranh sẽ "***đánh lừa***" được sự chú ý của tin tặc.

Theo nghĩa rộng, ***giấu tin*** cũng là ***hệ mật mã***, nhằm đảm bảo tính bí mật của thông tin.

Tóm lại, giải pháp hữu hiệu để "**che giấu**" thông tin là kết hợp cả 2 phương pháp:

Mã hóa thông tin trước, sau đó ***giấu*** **bản mã** vào bên trong một thông tin khác.

Có thể kết hợp cả ba giải pháp: ***Nén*** thông tin, ***Mã hóa*** thông tin, ***Giấu*** thông tin.

1.2.2.2. Khái niệm Thủy ký (WaterMarking)

Theo nghĩa rộng, "***Giấu tin***" nhằm thực hiện hai việc: ***Bảo vệ thông tin cần giấu*** và ***Bảo vệ chính môi trường giấu tin.***

Giấu (nhúng) thông tin mật vào một thông tin khác, sao cho người ta khó phát hiện ra thông tin mật đó. Đó là bảo vệ thông tin cần giấu.

Loại giấu tin này được gọi là "***Steganography***".

Giấu (nhúng) thông tin vào một thông tin khác, nhằm bảo vệ chính đối tượng được dùng để giấu tin vào. Tức là giấu tin để bảo vệ chính môi trường giấu tin.

Tin được giấu có vai trò như ***chữ ký*** hay ***con dấu*** dùng để xác thực (chứng nhận) thông tin (là môi trường giấu tin) Loại "giấu tin" này được gọi là "***Watermarking***".

Ví dụ:

Giấu một thông tin sở hữu của người chủ vào trong tác phẩm (tài liệu số) của họ, nếu ai sử dụng trái phép tác phẩm đó, thì ***Tin được giấu*** sẽ là ***vật chứng*** để chứng minh quyền hợp pháp của người chủ. Đó là ứng dụng để bảo vệ bản quyền tác phẩm "số".

Ví dụ:

Khi giấu một thông tin vào trong một tác phẩm (tài liệu số), ta có thể dùng chính thông tin giấu để kiểm xem tác phẩm có bị thay đổi nội dung hay không. Vì nếu tác phẩm bị thay đổi nội dung, thì không thể lọc ra được thông tin giấu nguyên vẹn như lúc ban đầu.

Đó là ứng dụng: Dùng thông tin giấu để ***kiểm tra tính toàn vẹn*** của môi trường giấu tin.

1.2.3. Nén thông tin.

1.2.3.1. Khái niệm "Nén tin" (Nén dữ liệu)

Nén dữ liệu (Data Compression) là **kỹ thuật** chuyển dữ liệu dạng "dư thừa" sang dạng "ít dư thừa", dữ liệu thu được sau khi nén ***nhỏ hơn*** dữ liệu gốc rất nhiều. Như vậy đỡ tốn bộ nhớ để lưu trữ dữ liệu, mặt khác tiết kiệm thời gian và chi phí truyền dữ liệu trên mạng máy tính.

Như vậy việc nghiên cứu các kỹ thuật nén dữ liệu là điều rất cần thiết, góp phần nâng cao hiệu quả sử dụng các tài nguyên của hệ thống máy tính.

Song song với việc nén dữ liệu, phải có kỹ thuật giải nén, nhằm chuyển dữ liệu được nén sang dữ liệu ban đầu.

Ngoài thuật ngữ "nén dữ liệu", do bản chất của kỹ thuật, nó còn có tên gọi là: "Giảm độ dư thừa", "Mã hóa ảnh gốc".

Hầu hết các máy tính hiện nay được trang bị "Modem", nhằm nén và giải nén các thông tin truyền và nhận thông qua đường điện thoại.

Tỷ lệ nén dữ liệu (Compression rate)

Tỷ lệ nén là một trong các đặc trưng quan trọng nhất của phương pháp nén. Người ta định nghĩa tỷ lệ nén là:

Tỷ lệ nén = (**1/ r**) %

Với **r** là ***Tỷ số nén*** = kích thước dữ liệu gốc / kích thước dữ liệu thu được sau nén.

Tỷ số nén **r** = 10 / 1, nghĩa là dữ liệu gốc là 10 phần, sau khi nén chỉ còn 1 phần.

Với dữ liệu ảnh, kết quả "nén" thường là **10 :1**. Theo kết quả nghiên cứu gần đây tại Viện kỹ thuật Georgie, kỹ thuật nén "***Fractal***" cho tỷ số nén là **30 : 1**.

1.2.3.2. Các phương pháp "Nén tin".

Hiện nay có nhiều kỹ thuật nén dữ liệu, nhưng chưa có phương pháp nén nào được coi là vạn năng, vì nó phụ thuộc vào nhiều yếu tố và bản chất của dữ liệu gốc. Kỹ thuật nén dữ liệu thường chỉ dùng cho một lớp dữ liệu có chung đặc tính nào đó.

Một số kỹ thuật nén dữ liệu hiện nay:

Mã độ dài loạt (Run length coding)

Mã hoá độ dài biến động (Variable length coding)

Mã Huffman.

1.2.4. Tường lửa (Firewall)

1.2.4.1. Khái niệm Tường lửa.

"***Tường lửa***" trong công nghệ mạng thông tin được hiểu là một hệ thống gồm phần cứng, phần mềm hay hỗn hợp phần cứng - phần mềm, có tác dụng như một***tấm ngăn cách*** giữa các tài nguyên thông tin của mạng nội bộ với thế giới Internet bên ngoài.

Phạm vi hẹp hơn như trong một mạng nội bộ, người ta cũng bố trí "***Tường lửa***" để ***ngăn cách*** các miền an toàn khác nhau (Security Domain)

Thuật ngữ "***Tường lửa***" có nguồn gốc trong kỹ thuật xây dựng để ngăn chặn, hạn chế hỏa hoạn. Trong CNTT, "Tường lửa" là kỹ thuật được tích hợp vào hệ thống mạng

để chống lại sự truy cập trái phép. Kỹ thuật nhằm ***bảo vệ thông tin nội bộ***, mặt khác ***hạn chế sự xâm nhập của thông tin trái phép*** vào hệ thống.

Kỹ thuật này phục vụ cho An toàn Hệ thống máy tính là chính, nhưng cũng hỗ trợ bảo đảm An toàn truyền tin, ví dụ *chống trộm cắp, sửa đổi thông tin* (chẳng hạn làm sai lệch tin tức hay giả mạo chữ ký) trước khi đến tay người nhận.

* ***Nhiệm vụ của Tường lửa****:*

Quyết định người nào, dịch vụ nào bên ngoài được truy cập vào bên trong Hệ thống máy tính. Quyết định người nào, dịch vụ nào bên trong được truy cập ra bên ngoài Hệ thống máy tính.

Để bảo đảm An toàn thông tin, tất cả các trao đổi thông tin từ ngoài vào trong hay ngược lại đều phải thực hiện thông qua "***Tường lửa***".

1.2.4.2. Các thành phần của Tường lửa.

a) Về mặt vật lý: "*Tường lửa*" gồm có:

+ Một hay nhiều máy chủ kết nối với bộ định tuyến (Router) hoặc có chức năng đó.

+ Các phần mềm quản lý An ninh trên hệ thống máy chủ, Ví dụ Hệ quản trị xác thực (Athentication), Hệ cấp quyền (Authorization), Hệ kế toán (Accounting), ..

b) Về mặt chức năng: "*Tường lửa*" có các thành phần:

+ Bộ lọc Packet (Packet - Filtering Router)

+ Cổng ứng dụng (Application - level Gateway hay Proxy Server)

+ Cổng mạch (Circuite - level Gateway)

1.2.5. Mạng riêng ảo (VPN)

1.2.5.1. Khái niệm Mạng riêng ảo.

Mạng riêng ảo (Virtual Private Networks: **VPN**) không phải là giao thức, cũng không phải là phần mềm máy tính. Đó là một ***chuẩn công nghệ*** cung cấp sự ***liên lạc an toàn*** giữa hai thực thể bằng cách ***mã hóa các giao dịch*** trên mạng công khai (không an toàn, ví dụ như Internet)

Qua mạng công khai, một thông điệp được chuyển qua một số máy tính, Router, switch, … trước khi đến đích. Trên đường truyền tin, thông điệp có thể bị chặn lại, bị sửa đổi hoặc bị đánh cắp. Mục đích của ***Mạng riêng ảo*** là bảo đảm các yêu cầu sau:

Tính bí mật, riêng tư (Privacy): Người ngoài cuộc khó thể hiểu được liên lạc đó.

Tính toàn vẹn (Intergrity): Người ngoài cuộc khó thể thay đổi được liên lạc đó.

Tính xác thực (Authenticity): Người ngoài cuộc khó thể tham gia vào liên lạc đó.

1.2.5.2. *Các thành phần của Mạng riêng ảo.*

a) Định đường hầm (Tunneling):

Đó là một cơ chế dùng để ***đóng gói*** một giao thức vào trong một giao thức khác.

Trên Internet, "định đường hầm" cho phép những giao thức như IPX, ppleTalk,.. được mã hóa, sau đó đóng gói trong **IP**.

Trong VPN, "định đường hầm" che giấu giao thức lớp mạng nguyên thủy bằng cách mã hóa gói dữ liệu này vào trong một vỏ bọc **IP**. Đó là một gói **IP**, được truyền một cách an toàn qua mạng Internet. Khi nhận được gói **IP** trên, người nhận tiến hành gỡ bỏ vỏ bọc bên ngoài, giải mã dữ liệu trong gói này, và phân phối nó đến thiết bị truy cập thích hợp.

Đường hầm cũng là một ***đặc tính ảo*** trong VPN. Các công nghệ đường hầm được dùng phổ biến hiện nay cho truy cập VPN gồm có: giao thức định đường hầm điểm, PPTN, L2F, L2TP hoặc IP Sec, GRE (Generic Route Encapsulation)

b) Bảo mật:

Bảo mật bằng ***Mã hóa***, đó là việc chuyển dữ liệu có thể đọc được (Clear text), vào trong một định dạng "khó" thể đọc được (Cipher text)

c) Thỏa thuận về chất lượng dịch vụ (**QoS**: **Service Quality**):

Thỏa thuận về *Chất lượng dịch vụ* thường định ra giới hạn cho phép về *độ trễ trung bình* của gói tin trong mạng. Ngoài ra, các thỏa thuận này được phát triển thông qua các dịch vụ với nhà cung cấp.

Mạng riêng ảo là sự kết hợp của ***Định đường hầm + Bảo mật + Thỏa thuận QoS***.

1.3. Các bài toán trong an toàn thông tin

1.3.1. Các bài toán trong Lý thuyết.

1.3.1.1. Bài toán Bảo mật thông tin.

1/ Nén thông tin.

2/ Mã hóa thông tin.

3/ Giấu thông tin.

1.3.1.2. Bài toán Bảo toàn thông tin.

Bảo toàn thông tin hay bảo đảm tính toàn vẹn của thông tin:

Người ngoài cuộc khó thể thay đổi được (sửa chữa lại nội dung) thông tin.

1/ Phương pháp 1:

Bảo toàn thông tin bằng các kỹ thuật: Nén thông tin, Mã hóa thông tin, Giấu tin.

Với các kỹ thuật trên, người ngoài khó nhận ra thông tin, nên khó sửa đổi nội dung của nó.

2/ Phương pháp 2:

Bảo toàn thông tin bằng các kỹ thuật: Hàm băm tạo đại diện bản tin.

3/ Phương pháp 3:

Bảo toàn thông tin bằng các kỹ thuật: Chữ ký số (Digital Signature)

4/ Phương pháp 4:

Bảo toàn thông tin bằng các kỹ thuật: Thủy vân ký (WaterMarking)

1.3.1.3. Bài toán Xác thực.

1) Các loại xác thực:

Có 2 loại chính: Xác thực thực thể, xác thực trách nhiệm về nội dung bản tin.

2) Các phương pháp xác thực:

Xác thực thực thể bằng 3 cách chính:

+ Biết cái gì ? Ví dụ mật khẩu, khóa ký, giao thức "bắt tay", …

+ Có cái gì ? Ví dụ Điện thoại di động, thẻ ATM, …

+ Sở hữu riêng cái gì ? Ví dụ vân tay, giọng nói, … (PP Sinh trắc học)

3) Các công nghệ xác thực: PKI, Thẻ thông minh, …

1.3.1.4. Bài toán Cấp quyền, Phân quyền.

1) Cấp quyền cho người dùng hợp pháp.

2) Không cấp quyền cho người dùng bất hợp pháp.

3) Phân quyền cho các đối tượng khác nhau.

1.3.1.5. Bài toán liên quan.

1) Kiểm tra số nguyên tố lớn

2) Tính phần tử nguyên thủy

3) Tính toán số nguyên lớn

4) Nhận dạng trong xác thực.

5) Định danh trong xác thực.

6) Chứng minh không tiết lộ thông tin

7) Chống chối cãi

1.3.1.6. Bài toán Công cụ tính toán

Tính toán "Mềm", Tính toán song song, Tính toán "hiệu năng cao", Tính toán "lưới", …

1.3.2. Các bài toán trong Ứng dụng.

1.3.2.1. Bài toán Xây dựng cơ sở hạ tầng An toàn thông tin.

1) Tường lửa (Firewall)

2) Mạng riêng ảo (VPN)

3) Hệ thống cấp chứng chỉ số (CA)

4) Cơ sở hạ tầng mật mã khóa công khai (PKI: Public Key Infrastructure)

5) Cơ sở hạ tầng ATTT phục vụ cho Hệ thống tính toán khắp nơi và di động.

1.3.2.2. Bài toán Bảo vệ bản quyền bản tin số.

1) Ký số ("Ký nổi")

2) Thủy ký ("Ký chìm")

3) Giải pháp Lưu vết và thu hồi Thiết bị thu bất hợp pháp.

1.3.2.3. Bài toán kinh tế xã hội.

1) Kiểm tra từ xa. 2) Bỏ phiếu từ xa.

3) Đấu thầu từ xa. 4) Giao dịch chứng khoán từ xa.

Chương 2. CƠ SỞ TOÁN HỌC

Lý thuyết mật mã là một ngành khoa học được xây dựng dựa trên cơ sở toán học, đặc biệt là lý thuyết số học. Chương này sẽ hệ thống lại *một số kiến thức toán học* cần thiết, được sử dụng trong lý thuyết mật mã và An toàn dữ liệu.

2.1. Một số khái niệm trong số học

2.1.1. Ước chung lớn nhất, bội chung nhỏ nhất

2.1.1.1. Khái niệm

1) Ước số và bội số

Cho hai số nguyên a và b, b ≠ 0. Nếu có một số nguyên q sao cho a = b*q, thì ta nói rằng a ***chia hết*** cho b, kí hiệu **b\a**. Ta nói b là ***ước*** của a, và a là ***bội*** của b.

Ví dụ:

Cho a = 6, b = 2, ta có 6 = 2*3, ký hiệu 2 \ 6. Ở đây 2 là ước của 6 và 6 là bội của 2.

Cho các số nguyên a, b ≠ 0, tồn tại cặp số nguyên (q, r) (0 ≤ r < /b/) duy nhất sao cho **a** = **b** * **q** + **r**. Khi đó **q** gọi là ***thương nguyên***, **r** gọi là ***số dư*** của phép chia a cho b. Nếu **r** = 0 thì ta có phép chia hết.

Ví dụ:

Cho a = 13, b = 5, ta có 12 = 5*2 + 3. Ở đây thương là q = 2, số dư là r = 3.

2) Ước chung lớn nhất, bội chung nhỏ nhất

Số nguyên **d** được gọi là ***ước chung*** của các số nguyên $a_1, a_2, \ldots, a_n$, nếu nó là ***ước*** của tất cả các số đó.

Số nguyên **m** được gọi là ***bội chung*** của các số nguyên $a_1, a_2, \ldots, a_n$, nếu nó là ***bội*** của tất cả các số đó.

Một ước chung **d** >0 của các số nguyên $a_1, a_2, \ldots, a_n$, trong đó mọi ước chung của $a_1, a_2, \ldots, a_n$ đều là ước của **d**, thì **d** được gọi là ***ước chung lớn nhất*** (UCLN) của $a_1, a_2, \ldots, a_n$. Ký hiệu **d** = **gcd** $(a_1, a_2, \ldots, a_n)$ hay **d** = **UCLN**$(a_1, a_2, \ldots, a_n)$

Nếu **gcd**(a_1, a_2, …, a_n) = 1, thì các số a_1, a_2, …, a_n được gọi là ***nguyên tố cùng nhau***.

Một bội chung **m** >0 của các số nguyên a_1, a_2, …, a_n , trong đó mọi bội chung của a_1, a_2, …, a_n đều là bội của **m**, thì **m** được goi là ***bội chung nhỏ nhất*** (BCNN) của a_1, a_2, …, a_n . Ký hiệu **m** = **lcm**(a_1, a_2, …, a_n) hay **m** = **BCNN**(a_1, a_2, …, a_n)

Ví dụ:

Cho a =12, b =15, **gcd**(12,15) = 3, **lcm**(12,15) = 60.

Hai số 8 và 13 là ***nguyên tố cùng nhau***, vì **gcd**(8, 13) = **1**.

Ký hiệu:

$\mathbf{Z_n}$ = {0, 1, 2, .. . , n-1} là tập các số nguyên không âm < n.

Z_n^* = {e ∈ $\mathbf{Z_n}$, e là nguyên tố cùng nhau với n}. Tức là e # 0.

Ví dụ:

$\mathbf{Z}_7$ = {0, 1, 2, 3, 4, 5, 6, 7}. Khi đó số phần tử của $\mathbf{Z}_7$ là /$\mathbf{Z}_7$/= 8.

$\mathbf{Z}_7^*$ = {1, 2, 3, 4, 5, 6, 7}. Khi đó số phần tử của $\mathbf{Z}_7^*$ là /$\mathbf{Z}_7^*$/= 8.

2.1.1.2. Tính chất

1) **d** = gcd(a_1, a_2, …, a_n) khi và chỉ khi tồn tại các số x_1, x_2,…, x_n sao cho:

d = $a_1x_1+a_2x_2+\ldots+a_nx_n$

Đặc biệt: a_1, a_2, …, a_n nguyên tố cùng nhau ⇔ tồn tại các số x_1, x_2,…, x_n sao cho:**1** = $a_1x_1+a_2x_2+\ldots+a_nx_n$

2) **d** = gcd(a_1, a_2, …, a_n) ⇔ gcd(a_1/d, a_2/d,…, a_n/d) =1.

3) **m** = lcm(a_1, a_2, …, a_n) ⇔ gcd(m/a_1, m/a_2,…, m/a_n) =1.

4) gcd(**m** a_1, **m** a_2, …, **m** a_n) = **m** * gcd(a_1, a_2, …, a_n) (với m ≠ 0)

5) Nếu gcd(a, b) =1 thì lcm(a, b) = a * b

6) Nếu b>0, a = bq+r thì gcd(a,b) = gcd(b, r)

2.1.1.3. Thuật toán Euclide tìm ước chung lớn nhất

1) Bài toán

**** Dữ liệu vào*:** Cho hai số nguyên không âm a, b, a ≥ b.

**** Kết quả*:** **gcd**(a,b)

2) Thuật toán (*Mô phỏng bằng ngôn ngữ Pascal*)

Readln(a, b);

```
While  b>0 do
    begin
       r := a mod b;  a := b;  b := r;
    end;
Writeln(a);
```

3) Ví dụ: a=30, b=18; gcd(30,18) = gcd(18,12) = gcd(12,6) = gcd(6,0) = 6

a	b	r	a = b.q + r
3	1	1	30 = 18 *
1	1	6	18 = 12 *
1	6	0	12 = 6 * 2

2.1.1.4. Thuật toán Euclide mở rộng

1) Bài toán:

**** Dữ liệu vào*:** Cho hai số nguyên không âm a, b, a ≥ b.

**** Kết quả*:** **d = gcd** (a,b) và hai số **x**, **y** sao cho: a **x** + b **y** = **d.**

2) Thuật toán: (*Mô phỏng bằng ngôn ngữ Pascal*)

```
  Readln(a, b);
  IF  b=0 THEN
     Begin
  d := a; x := 1; y := 0;
  writeln(d, x, y);
     End
  ELSE
     Begin
  x2 := 1; x1 := 0; y2 := 0; y1 := 1;
  While  b>0  Do
          begin
                q := a div b;  r := a mod b;
                x := x2-q*x1; y := y2-q*y1;
                a := b; b := r; x2 := x1; x1 := x; y2 := y1; y1 := y;
```

```
        end;
    d := a; x := x2; y := y2;
    writeln(d, x, y);
End;
```

2.1.2. Quan hệ "Đồng dư"

2.1.2.1. Khái niệm

Cho các số nguyên a, b, **m** (m > 0) Ta nói rằng ***a*** và ***b "đồng dư" với nhau*** theo **modulo m**, nếu chia a và b cho **m**, ta nhận được cùng một số dư.

Ký hiệu: **a ≡ b** (mod **m**)

Ví dụ:

17 ≡ 5 (mod 3) vì chia 17 và 5 cho 3, được cùng số dư là 2.

Nhận xét: Các mệnh đề sau đây là tương đương:

1) a ≡ b (mod m)
2) m \ (a – b)
3) Tồn tại số nguyên **t** sao cho a = b + m **t**

Chứng minh:

1) ⇒ 2):

Nếu có 1), thì theo định nghĩa: a, b chia cho m, phải có cùng số dư, do đó:

$a = mq_a + r$; $b = mq_b + r$; Suy ra $(a – b) = m (q_a - q_b)$, tức là m \ (a - b)

2) ⇒ 3):

Nếu có 1), tức là m \ (a – b) Nghĩa là có t ∈ Z sao cho (a - b) = mt hay a = b + mt

3) ⇒ 1):

Nếu có 1), tức là tồn tại số nguyên **t** sao cho a = b + m **t**.

Lấy a chia cho m, giả sử thương là q_a và dư r, hay **a** = mq_a + r (0 ≤ r <m), do đó:

b + m **t** = a = mq_a + r hay **b** = m(q_a - **t**) + r (0 ≤ r < m) Điều đó chứng tỏ khi chia a và b cho m được cùng số dư r, hay a ≡ b (mod m)

2.1.2.2. Các tính chất của quan hệ "đồng dư"

1) Quan hệ "đồng dư" là quan hệ tương đương trong Z:

Với mọi số nguyên dương m ta có:

a ≡ a (mod m) với mọi a ∈ Z; *(tính chất phản xạ)*

a ≡ b (mod m) thì b ≡ a (mod m); *(tính chất đối xứng)*

a ≡ b (mod m) và b ≡ c (mod m) thì a ≡ c (mod m); *(tính chất bắc cầu)*

2) Tổng hay hiệu các "đồng dư":

(a+b) (mod n) ≡ [(a mod n) + (b mod n)] (mod n)

(a- b) (mod n) ≡ [(a mod n) - (b mod n)] (mod n)

Tổng quát:

Có thể cộng hoặc trừ từng vế nhiều đồng dư thức theo cùng một modulo **m**,

ta được một đồng dư thức theo cùng modulo **m**, tức là:

Nếu $a_i \equiv b_i \pmod m$, i = 1...k, thì $\sum_{i=1}^{k} t_i a_i = \sum_{i=1}^{k} t_i b_i \pmod m$ với $t_i = \pm 1$.

3) Tích các "đồng dư":

(a* b) (mod n) ≡ [(a mod n) * (b mod n)] (mod n)

Tổng quát:

Có thể nhân từng vế nhiều đồng dư thức theo cùng một modulo **m**,

ta được một đồng dư thức theo cùng modulo **m**, tức là:

Nếu $a_i \equiv b_i \pmod m$ với i=1..k, thì ta có: $\prod_{i=1}^{k} a_i = \prod_{i=1}^{k} b_i \pmod m$

Hệ quả:

* Có thể cộng hoặc trừ cùng một số vào hai vế của một đồng dư thức.

* Có thể chuyển vế các số hạng của đồng dư thức bằng cách đổi dấu các số hạng đó.

* Có thể cộng vào một vế của đồng dư thức một ***bội*** của modulo:

a ≡ b (mod m) → a+km ≡ b (mod m) với mọi k ∈ Z

* Có thể nhân hai vế của một đồng dư thức với cùng một số:

a ≡ b (mod m) → ac ≡ bc (mod m) với mọi c ∈ Z

* Có thể nâng lên ***lũy thừa*** bậc nguyên không âm cho 2 vế của một đồng dư thức:

$a \equiv b \pmod m \rightarrow a^n \equiv b^n \pmod m$ với mọi $n \in Z^+$

* Có thể ***chia*** 2 vế đồng dư thức cho một ước chung nguyên tố với modulo:

c\a, c\b, (c,m)=1, a ≡ b (mod m) ⇒ a/c ≡ b/c (mod m)

* Có thể ***nhân*** 2 vế đồng dư thức và modulo với cùng một số nguyên dương,

Nếu $a \equiv b \pmod m$, $c > 0 \Rightarrow ac \equiv bc \pmod{mc}$

* Có thể ***chia*** 2 vế đồng dư thức và modulo cho cùng một số nguyên dương là ước chung của chúng:

Nếu $c \setminus (a, b, m) \Rightarrow a/c \equiv b/c \pmod{m/c}$

* $a \equiv b \pmod m \Rightarrow a \equiv b \pmod k$ với $k \setminus m$

* $a \equiv b \pmod m \Rightarrow \gcd(a, m) = \gcd(b, m)$

Các lớp thặng dư

Quan hệ "đồng dư" theo modulo **m** trên tập Z (tập các số nguyên) là một quan hệ tương đương (vì có tính chất phản xạ, đối xứng, bắc cầu), do đó nó tạo ra trên tập Z một phân hoạch gồm các lớp tương đương: hai số nguyên thuộc cùng một lớp tương đương khi và chỉ khi chúng có cùng một số dư khi chia cho **m**.

Mỗi lớp tương đương đại diện bởi một số duy nhất trong tập $\mathbf{Z_m} = \{0, 1, \ldots, m-1\}$ là số dư khi chia các số trong lớp cho **m**, ký hiệu một lớp được đại diện bởi số **a** là $\mathbf{[a]_m}$.

Như vậy $[a]_m = [b]_m \Leftrightarrow a \equiv b \pmod m$

Vì vậy ta có thể đồng nhất $\mathbf{Z_m}$ với tập các lớp tương đương theo modulo **m**.

$\mathbf{Z_m} = \{0, 1, 2, \ldots, \mathbf{m}-1\}$ được gọi là tập các "***thặng dư đầy đủ***" theo modulo **m**. Mọi số nguyên bất kỳ đều có thể tìm được trong $\mathbf{Z_m}$ một số đồng dư với mình theo modulo **m**. [30]

2.1.3. Số nguyên tố

2.1.2.1. Khái niệm

Số nguyên tố là số tự nhiên lớn hơn 1 và chỉ có hai ước là 1 và chính nó.

Ví dụ:

Các số 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37 là số nguyên tố.

Số 2 là số nguyên tố ***chẵn*** duy nhất.

Số nguyên tố có vai trò và ý nghĩa to lớn trong số học và lý thuyết mật mã.

Bài toán kiểm tra tính nguyên tố của một số nguyên dương n và phân tích một số n ra thừa số nguyên tố là các bài toán rất được quan tâm.

Ví dụ: ***10 số nguyên tố lớn đã được tìm thấy*** [33]

rank	Prime	Digits	Who	whe	refer
1	$2^{32582657}-1$	980835	G9	2006	Mers
2	$2^{30402457}-1$	915205	G9	2005	Mers
3	$2^{25964951}-1$	781623	G8	2005	Mers
4	$2^{24036583}-1$	723573	G7	2004	Mers
5	$2^{20996011}-1$	632043	G6	2003	Mers
6	$2^{13466917}-1$	405394	G5	2001	Mers
7	$19249\cdot2^{13018}$	391899	SB10	2007	
8	$27653\cdot2^{91674}$	275967	SB8	2005	
9	$28433\cdot2^{78304}$	235720	SB7	2004	
10	$33661\cdot2^{70312}$	211661	SB11	2007	

2.1.2.2. Định lý về số nguyên tố

1) Định lý: về số nguyên dương > 1.

Mọi số nguyên dương **n** > 1 đều có thể biểu diễn được ***duy nhất*** dưới dạng:

$n = P_1^{n_1}.P_2^{n_2}...P_k^{n_k}$, trong đó:

k, $\mathbf{n_i}$ (i =1,2,..,k) là các số tự nhiên, $\mathbf{P_i}$ là các số nguyên tố, từng đôi một khác nhau.

2) Định lý: Mersenne.

Cho $\mathbf{p} = \mathbf{2^k - 1}$, nếu **p** là số nguyên tố, thì **k** phải là số nguyên tố.

Chứng minh

Bằng phản chứng, giả sử **k** không là nguyên tố. Khi đó k = a.b với 1< a, b < k. Như vậy $\mathbf{p} = 2^k - 1 = 2^{ab} - 1 = (2^a)^b - 1 = (2^a - 1)E$

(Trong đó E là một biểu thức nguyên - áp dụng công thức nhị thức Niu-tơn)

Điều này mâu thuẫn giả thiết **p** là nguyên tố. Vậy giả sử là sai, hay k là số nguyên tố.

3) Hàm Euler:

Cho số nguyên dương n, ***số lượng*** các số nguyên dương bé hơn n và ***nguyên tố cùng nhau*** với n được ký hiệu ϕ**(n)** và gọi là hàm ***Euler***.

Nhận xét: Nếu **p** là số nguyên tố, thì ϕ**(p)** = **p-1**

Ví dụ:

Tập các số nguyên không âm nhỏ hơn 7 là $\mathbf{Z}_7$ = {0, 1, 2, 3, 4, 5, 6, 7}.

Do **7** là ***số nguyên tố***, nên Tập các số nguyên dương nhỏ hơn 7 và *nguyên tố cùng nhau* với 7 là $\mathbf{Z}_7^*$ ={1, 2, 3, 4, 5, 6, 7}. Khi đó /**Z**/ = ϕ**(p)** = **p-1** = 8 - 1 = 7**.**

Định lý: về Hàm Euler.

Nếu n là tích của hai số nguyên tố **n** = **p.q**, thì ϕ**(n)** = ϕ(p)ϕ(q) = (p-1)(q-1)

2.1.2.3. Phương pháp kiểm tra tính nguyên tố

Kiểm tra tính nguyên tố của một số nguyên dương là bài toán nảy sinh trong nhiều ứng dụng, đặc biệt là trong lý thuyết mật mã. Năm 1975 Pratt đã chứng minh nó thuộc lớp NP và thuộc lớp co-NP $\cap$NP, đây là bài toán "khó".

1) Phương pháp cổ điển.

Ý tưởng:

Kiểm tra tính nguyên tố của một số nguyên dương **n** theo định nghĩa:

Thử lần lượt tìm các ***ước*** của **n**, từ 2 đến n / 2.

Nếu không tìm được ước nào thì kết luận **n** là nguyên tố.

Thuật toán:

```
KT := 0;
for  i := 2  to  sqrt(n) do
    if  (n mod i) = 0 then
        begin
            KT := 1; Break;
        end;
```

IF KT = 1 THEN Writeln (‘n nguyên tố ‘) ELSE Writeln (‘n không nguyên tố ‘);

Chú ý

Trong thuật toán trên, vòng lặp có số lần lặp là $n^{1/2}/2 = n$.

Nếu $n = 10^{150}$ thì thuật toán trên phải tính khoảng 10^{150} phép tính.

Nếu dùng 1 máy tính nhanh nhất hiện nay (2007): khoảng 500 000 tỷ = $5 * 10^{14}$ phép tính trong 1 giây, thì thời gian thực hiện là khoảng $5 * 10^{136}$ giây.

Mỗi ngày có khoảng 24 giờ * 60 phút * 60 giây $\approx 10^5$ giây.

Mỗi năm có khoảng 365 ngày * 10^5 giây $\approx 10^8$ giây.

Như vậy nếu $n = 10^{150}$ thì thuật toán trên phải tính khoảng $\mathbf{10^{128}}$ năm.

2) Phương pháp “xác suất“.

Trên cơ sở các định lý về số nguyên tố, hiện nay người ta có các phương pháp “xác suất“ để kiểm tra tính nguyên tố của một số nguyên dương n.

Ví dụ như các phương pháp: Solovay-Strassen, Lehmann-Peralta, Miller-Rabin.

***Định lý Ferma*:**

Nếu p là số nguyên tố, a là số nguyên, thì $\mathbf{a^p \equiv a} \pmod p$

Nếu p không chia hết a, thì $\mathbf{a^{p-1} \equiv 1} \pmod p$

Ví dụ: $4^7 \equiv 4 \pmod 7$; $4^{7-1} \equiv 1 \pmod 7$

***Định lý Euler*:**

Nếu gcd(a, m) = 1 thì $a^{\phi(m)} \equiv 1 \pmod m$

Trường hợp m là số nguyên tố, ta có định lý Ferma.

Ví dụ: m = 10, $\phi(m) = \phi(2)\phi(5) = 1 * 4 = 4$.

Ta có $7^4 \equiv 1 \pmod{10}$, $9^4 \equiv 1 \pmod{10}$, $21^4 \equiv 1 \pmod{10}$

Hệ quả 1: Nếu gcd(c, m) = 1 và $a \equiv b \pmod{\phi(m)}$ với a, b là các số tự nhiên, thì

$c^a \equiv c^b \pmod m$ và suy ra $c^a \bmod m = c^{a \bmod \phi(m)} \bmod m$.

Chứng minh: $a \equiv b \pmod{\phi(n)}$ nên $a = b + k\phi(m)$, $k \in Z$ và vì vậy

$c^a = c^{b+k\phi(m)} = c^b.(c^{\phi(m)})^k \equiv 1 \pmod m$, theo theo định lý Euler.

Nhận xét: Hệ quả trên giúp giảm nhẹ việc tính toán đồng dư của lũy thừa bậc cao.

Ví dụ: Ta thấy $\phi(15) = \phi(5)\phi(3) = 4.2 = 8$ và $1004 \equiv 4 \pmod 8$

Do đó $\mathbf{2^{1004}} \pmod{15} = 2^4 \pmod{15} = 16 \pmod{15} = 1$.

Hệ quả 2: Nếu các các số nguyên e, d thỏa mãn **e.d ≡ 1** (mod ϕ**(n)**), thì với mọi số **c** nguyên tố cùng nhau với m, ta có $(c^e)^d \equiv c \pmod m$

Chứng minh: Đặt a = ed và b = 1, từ hệ quả 1 ta có hệ quả 2.

Hệ quả này đóng vai trò then chốt trong việc thiết lập các hệ mã mũ (VD RSA)

2.1.2.4. Tính toán đồng dư của " lũy thừa" lớn

1) Trường hợp a > ϕ(m):

Trong trường hợp a > ϕ(m), khi ấy b < a. Người ta dùng Hệ quả 1 để tính "đồng dư" của " lũy thừa" lớn.

2) Trường hợp ϕ(m) > a:

Trong thực tế tính toán thường gặp m lớn, do đó ϕ(m) lớn, thậm chí > a, khi ấy người ta dùng kỹ thuật khác, ví dụ Phương pháp bình phương liên tiếp.

**** Phương pháp bình phương liên tiếp***.

Ví dụ: Tính $87^{43} \pmod{103}$

Khai triển số mũ 43 dưới dạng cơ số 2:

$43 = 32+8+2+1 = 2^5 + 2^3 + 2^1 + 2^0$ (*)

Tính liên tiếp các "đồng dư" bình phương như sau:

$87 \pmod{103} = 87$ (ứng với 2^0)

$87^2 \pmod{103} = 50$ (ứng với 2^1)

$87^4 \pmod{103} = 50^2 \pmod{103} = 28$

$87^8 \pmod{103} = 28^2 \pmod{103} = 63$ (ứng với 2^3)

$87^{16} \pmod{103} = 63^2 \pmod{103} = 55$

$87^{32} \pmod{103} = 55^2 \pmod{103} = 38$ (ứng với 2^5)

Theo khai triển (*), lấy tích của các lũy thừa bậc 2^5 , 2^3 , 2^1 , 2^0

(rút gọn theo modulo 130), thu được kết quả:

$87^{43} \pmod{103} = 38 * 63 * 50 * 87 \pmod{103} = 85$

**** Định lý về Số dư (ĐL Trung Quốc):***

Cho tập số nguyên tố cùng nhau từng đôi một $m_1, m_2, \ldots m_r$.

Với mỗi bộ số nguyên bất kỳ $a_1, a_2, \ldots a_r$, hệ phương trình đồng dư:

$x \equiv a_i \pmod{m_i}$, (i =1, 2, …, r), luôn có ***nghiệm duy nhất*** theo modulo **m**,

m = $m_1.m_2.\ldots m_r$. Nghiệm này có thể tính theo công thức:

x = $a_1 m_2 m_3 \ldots m_r b_1 + m_1 a_2 m_3 \ldots m_r b_2 + m_1 m_2 a_3 m_3 \ldots m_r b_3 + \ldots +$

$m_1 m_2 \ldots m_{r-1} a_r b_r \pmod{m_1.m_2.\ldots m_r}$,

trong đó $b_i = (m_1.m_2\ldots m_{i-1}m_{i+1}\ldots m_r)^{-1} \pmod{m_i}$, với mọi i =1, 2,…, r.

Nhận xét:

Định lý số dư Trung Quốc cho phép tính đồng dư theo modulo của một số lớn (tích của nhiều số nguyên tố cùng nhau), thông qua tính toán đồng dư theo modulo các số nhỏ (từng thừa số)

Ví dụ: Tìm nghiệm của hệ phương trình:

$$\begin{cases} x \equiv 3118 \pmod{5353} \\ x \equiv 139 \pmod{391} \\ x \equiv 239 \pmod{247} \end{cases}$$

Vì các số 5353, 391, 247 nguyên tố cùng nhau, nên theo định lý Trung Quốc về số dư hệ, có nghiệm duy nhất theo modulo **m** = 5353*391*247 = 516976681.

Để tìm **x** mod **m** ta tính:

$m_1 = m/5353 = 96577 \rightarrow y_1 = 96577^{-1} \bmod 5353 = 5329$

$m_2 = m/391 = 1322191 \rightarrow y_2 = 1322191^{-1} \bmod 391 = 16$

$m_3 = m/247 = 2093023 \rightarrow y_3 = 2093023^{-1} \bmod 247 = 238$

x = 31188.96577.5329 + 139.1322191.16 + 239.2093023.238 (mod m)

= 13824 **(mod** m**)**

2.2. Một số khái niệm trong Đại số

2.2.1. Cấu trúc Nhóm

1) Khái niệm Nhóm

Nhóm là một bộ (**G**, *), trong đó **G** $\neq \varnothing$, * là ***phép toán hai ngôi*** trên **G** thoả mãn ba tính chất sau:

+ Phép toán có tính kết hợp: (x*y)* z = x*(y*z) với mọi x, y, z $\in$ G.

+ Có phần tử phần tử ***trung lập*** **e** $\in$ G: x***e** = **e***x = x với mọi x $\in$ G.

+ Với mọi x$\in$ G, có phần tử nghịch đảo x'$\in$ G: x * x' = x' * x = e.

Cấp của nhóm **G** được hiểu là số phần tử của nhóm, ký hiệu là $|G|$.

Cấp của nhóm có thể là ∞ nếu **G** có vô hạn phần tử.

Nhóm Abel là nhóm (**G**, *), trong đó phép toán hai ngôi * có tính giao hoán.

Tính chất: Nếu a * b = a * c, thì b = c.

Nếu a * c = b * c, thì a = b.

Ví dụ:

* Tập hợp các số nguyên Z cùng với phép cộng (+) thông thường là nhóm giáo hoán, có phần tử đơn vị là số 0. Gọi là ***nhóm cộng*** các số nguyên.

* Tập Q^* các số hữu tỷ khác 0 (hay tập R^* các số thực khác 0), cùng với phép nhân (*) thông thường là nhóm giao hoán. Gọi là ***nhóm nhân*** các số hữu tỷ (số thực) khác 0.

* Tập các vectơ trong không gian với phép toán cộng vectơ là nhóm giao hoán.

2) Nhóm con của nhóm (**G**, *)

Nhóm con của G là tập $S \subset G$, $S \neq \phi$, và thỏa mãn các tính chất sau:

+ Phần tử trung lập e của G nằm trong S.

+ S khép kín đối với phép tính (*) trong G, tức là $x*y \in S$ với mọi $x, y \in S$.

+ S khép kín đối với phép lấy nghịch đảo trong G, tức $x^{-1} \in S$ với mọi $x \in S$.

2.2.2. Nhóm Cyclic

1) Khái niệm Nhóm Cyclic

Nhóm (**G**, *) được gọi là ***Nhóm Cyclic*** nếu nó được sinh ra bởi một trong các phần tử của nó. Tức là có phần tử $\mathbf{g} \in \mathbf{G}$ mà với mỗi $\mathbf{a} \in \mathbf{G}$, đều tồn tại số $\mathbf{n} \in N$ để

$\mathbf{g}^{\mathbf{n}} = \mathbf{g} * \mathbf{g} * \ldots * \mathbf{g} = \mathbf{a}$. (Chú ý g * g * … * g là $\mathbf{g} * \mathbf{g}$ với **n** lần)

Khi đó **g** được gọi là ***phần tử sinh*** hay ***phần tử nguyên thuỷ*** của nhóm **G**.

Nói cách khác: **G** được gọi là Nhóm Cyclic nếu tồn tại $\mathbf{g} \in \mathbf{G}$ sao cho mọi phần tử trong **G** đều là một ***luỹ thừa nguyên*** nào đó của **g**.

Ví dụ: Nhóm $(\mathbf{Z}^+, +)$ gồm các số nguyên dương là *Cyclic* với phần tử sinh $\mathbf{g} = \mathbf{1}$.

2) Cấp của Nhóm Cyclic:

Cho (**G**, *) là ***Nhóm Cyclic*** với phần tử sinh **g**. và phần tử trung lập **e.**

Nếu tồn tại số tự nhiên ***nhỏ nhất*** **n** mà $\mathbf{g}^{\mathbf{n}} = \mathbf{e}$, thì **G** sẽ chỉ gồm có **n** phần tử khác nhau: $\mathbf{e, g, g^2, g^3, \ldots, g^{n-1}}$. Khi đó **G** được gọi là *nhóm Cyclic* hữu hạn ***cấp*** **n**.

Nếu không tồn tại số tự nhiên **n** để $\mathbf{g}^{\mathbf{n}} = \mathbf{e}$, thì **G** có ***cấp*** ∞.

Ví dụ: ($\mathbf{Z}^+$, +) gồm các số nguyên dương là *Cyclic* với phần tử sinh **g** = **1**, **e** = 0.

Đó là Nhóm Cyclic vô hạn, vì không tồn tại số tự nhiên **n** để $\mathbf{g}^{\mathbf{n}} = \mathbf{e}$,

3) Cấp của một phần tử trong Nhóm Cyclic:

Phần tử $\alpha \in \mathbf{G}$ được gọi là có ***cấp d***, nếu **d** là số nguyên dương ***nhỏ nhất*** sao cho $\alpha^{\mathbf{d}} = \mathbf{e}$, trong đó **e** là phần tử trung lập của **G**.

Như vậy phần tử α có ***cấp 1***, nếu $\alpha = \mathbf{e}$.

2.2.3. Nhóm ($\mathbf{Z_n}^*$, phép nhân mod n)

1) Khái niệm Tập thặng dư thu gọn theo modulo

* Kí hiệu $\mathbf{Z_n} = \{0, 1, 2, \ldots, n-1\}$ là tập các số nguyên không âm < n.

$\mathbf{Z_n}$ và phép cộng (+) lập thành ***nhóm Cyclic*** có phần tử sinh là **1**, phần tử trung lập **e** = 0.

($\mathbf{Z_n}$, +) gọi là nhóm cộng, đó là nhóm hữu hạn có cấp n.

* Kí hiệu $\mathbf{Z_n}^* = \{\mathbf{x} \in \mathbf{Z_n}$, **x** là nguyên tố cùng nhau với **n**$\}$. Tức là **x** phải $\neq 0$.

$\mathbf{Z_n}^*$ được gọi là ***Tập thặng dư thu gọn theo mod n***, có số phần tử là $\phi(\mathbf{n})$

$\mathbf{Z_n}^*$ với *phép nhân mod n* lập thành một *nhóm* (nhóm nhân), pt trung lập **e** = 1.

Tổng quát ($\mathbf{Z_n}^*$, *phép nhân mod n*) không phải là nhóm Cyclic.

Nhóm nhân $\mathbf{Z_n}^*$ là Cyclic chỉ khi **n** có dạng: 2, 4, p^k, hay $2p^k$ với p là nguyên tố lẻ.

2) Một số kết quả đã được chứng minh

* ***Định lý Lagrange:*** Nếu **G** là nhóm cấp **n** và $\alpha \in \mathbf{G}$, thì ***Cấp*** của α *là* ***ước*** *của* **n**.

* ***Hệ quả:*** Giả sử $\alpha \in Z_n^*$ có ***Cấp*** m, thì m *là* ***ước*** *của* ϕ(n)

* ***Định lý:*** Nếu p là số nguyên tố thì Z_p^* là nhóm Cyclic.

Nếu $b \in Z_n^*$ thì $b^{\phi(n)} \equiv 1 \pmod n$ Nếu p là số nguyên tố thì $\phi(p) = p-1$.

Do đó với $b \in Z_p^*$ (tức b nguyên tố với p), thì $b^{\phi(p)} \equiv 1 \pmod n$, hay $b^{p-1} \equiv 1 \pmod n$

<u>Chú ý</u>

Theo định nghĩa, phần tử $\alpha \in Z_n^*$ có ***cấp*** **d** nếu **d** là số nguyên dương nhỏ nhất sao cho $\alpha^{\mathbf{d}} = \mathbf{e}$ trong Z_n^*. Như vậy trong Z_n^* ta hiểu là $\alpha^{\mathbf{d}} \equiv \mathbf{e} \pmod n$

Định lý**:** Nhóm con của một nhóm Cyclic là một nhóm Cyclic.

3) Phần tử nghịch đảo đối với phép nhân

**** Định nghĩa*:** Cho **a** $\in Z_n$, nếu tồn tại **b** $\in Z_n$ sao cho **a b** $\equiv$ 1 (mod **n**), ta nói **b** là ***phần tử nghịch đảo*** của **a** trong Z_n và ký hiệu $\mathbf{a}^{-1}$.

Một phần tử có phần tử nghịch đảo, gọi là khả nghịch.

**** Định lý***: UCLN (a, **n**) = 1 $\Leftrightarrow$ Phần tử a $\in Z_n$ có phần tử nghịch đảo.

Chứng minh:

Nếu $a\,a^{-1} \equiv 1 \pmod n$ thì $a\,a^{-1} = 1 + kn \leftrightarrow a\,a^{-1} - kn = 1 \rightarrow (a, n) = 1$.

Nếu (a, n) = 1, ta có $a\,a^{-1} + kn = 1 \rightarrow a\,a^{-1} = 1+kn$, do đó $a\,a^{-1} \equiv 1 \pmod n$

**** Hệ quả***: Mọi phần tử trong $\mathbf{Z_n^*}$ đều có phần tử nghịch đảo.

**** Tìm phần tử nghịch đảo bằng Thuật toán Euclid mở rộng.***

Input: **a** $\in Z_n$, **n**, Output: Phần tử nghịch đảo của **a**.

```
Procedure  Invert(a, n);
Begin
  g0:=n; g1:=a; u0:=1; u1:=0; v0:=0; v1:=1;
  i:=1;
  while  gi  ≠ 0  do
     begin
           y := gi-1 div gi;       gi+1 := gi+1 - y.gi;
           ui+1 := ui+1 - y.ui;    vi+1 := vi+1 - y.vi;
           i := i+1;
      end;
  t := vi+1;
  if  t > 0  then  a-1 := t  else  a-1 := t + n;
End;
```

Ví dụ: Tìm phần tử nghịch đảo của 3 trong $\mathbf{Z_7}$

Tức là phải giải phương trình 3 **x** $\equiv$ 1 (mod **7**), x sẽ là phần tử nghịch đảo của 3.

I	$\mathbf{g_i}$	$\mathbf{u_i}$	$\mathbf{v_i}$	**y**
1	7	1	0	
1	3	0	1	2
2	1	1	-2	3
3	0			

Vì **t** = v_2 = -2 < 0 do đó **x** = $\mathbf{a^{-1}}$:= **t** + **n** = -2 + 7 = 5.

Vậy 5 là phần tử nghịch đảo của 3 trong $\mathbf{Z_7}$

Chú ý

Định lý **(**Euler tổng quát): Nếu (a, n) = 1 thì $\mathbf{a^{\phi(n)}}$ mod n = 1

Hệ quả: Nếu **p** là số nguyên tố và (**a**, **p**) = 1, thì $\mathbf{a^{p-1}}$ (mod p) = 1

4) Khái niệm Logarit rời rạc

Cho **p** là số nguyên tố, **g** là phần tử nguyên thuỷ của $\mathbf{Z_p}$, $\beta \in Z_p^*$

"*Logarit rời rạc*" chính là việc giải phương trình **x** = $\mathbf{\log_g \beta}$ (mod **p**) với ẩn **x**.

Hay phải tìm số **x** duy nhất sao cho: $\mathbf{g^x \equiv \beta}$ (mod **p**)

5) Thặng dư bậc hai và ký hiệu Legendre.

**** Thặng dư bậc hai:***

Cho **p** là số nguyên tố lẻ, x là một số nguyên dương ≤ p-1.

x được gọi là "***thặng dư bậc hai***" mod **p**, nếu phương trình

$\mathbf{y^2 \equiv x}$ mod **p** có lời giải.

**** Kí hiệu Legendre:***

Cho p là số nguyên tố lẻ, và a là một số nguyên dương bất kỳ.

ký hiệu Legendre như sau:

a 0, nếu a ≡ 0 mod p

= 1, nếu a là thặng dư bậc hai mod p

b 1, trong các trường hợp còn lại.

2.3. Khái niệm độ phức tạp của thuật toán

2.3.1. Khái niệm Thuật toán

2.3.1.1. Khái niệm Bài toán

Bài toán được diễn đạt bằng hai phần:

Input: Các dữ liệu vào của bài toán.

Output: Các dữ liệu ra của bài toán (kết quả)

Không mất tính chất tổng quát, giả thiết các dữ liệu trong bài toán đều là số nguyên.

2.3.1.2. Khái niệm Thuật toán

"***Thuật toán***" được hiểu đơn giản là cách thức để giải một bài toán. Cũng có thể được hiểu bằng hai quan niệm: Trực giác hay Hình thức như sau:

1) Quan niệm trực giác về "***Thuật toán***".

Một cách trực giác, Thuật toán được hiểu là một dãy hữu hạn các qui tắc (Chỉ thị, mệnh lệnh) mô tả một quá trình tính toán, để từ dữ liệu đã cho (Input) ta nhận được kết quả (Output) của bài toán.

2) Quan niệm toán học về "***Thuật toán***".

Một cách hình thức, người ta quan niệm thuật toán là một máy Turing.

Thuật toán được chia thành hai loại: Đơn định và không đơn định.

Thuật toán đơn định (*Deterministic*):

Là thuật toán mà kết quả của mọi phép toán đều được xác định duy nhất.

Thuật toán không đơn định (*NonDeterministic*):

Là thuật toán có ít nhất một phép toán mà kết quả của nó là không duy nhất.

2.3.1.3. Hai mô hình tính toán

Hai quan niệm về thuật toán ứng với hai mô hình tính toán.

Ứng với hai mô hình tính toán có hai cách biểu diễn thuật toán.

1) Mô hình ứng dụng: Thuật toán được biểu diễn bằng ngôn ngữ tựa Algol.

+ Đơn vị nhớ: Một ô nhớ chứa trọn vẹn một dữ liệu.

+ Đơn vị thời gian: Thời gian để thực hiện một phép tính cơ bản trong số học hay logic như cộng, trừ, nhân, chia, ...

2) Mô hình lý thuyết: Thuật toán được biểu diễn bằng ngôn ngữ máy Turing.

+ Đơn vị nhớ: Một ô nhớ chứa một tín hiệu. Với mã nhị phân thì đơn vị nhớ là 1 bit.

+ Đơn vị thời gian: Thời gian để thực hiện một bước chuyển hình trạng

2.3.2. Khái niệm Độ phức tạp của thuật toán

1) Chi phí của thuật toán *(Tính theo một bộ dữ liệu vào):*

Chi phí phải trả cho một quá trình tính toán gồm chi phí về thời gian và bộ nhớ.

Chi phí thời gian của một quá trình tính toán là thời gian cần thiết để thực hiện một quá trình tính toán. Với thuật toán tựa Algol: Chi phí thời gian là số các phép tính cơ bản thực hiện trong quá trình tính toán.

Chi phí bộ nhớ của một quá trình tính toán là số ô nhớ cần thiết để thực hiện một quá trình tính toán.

Gọi A là một thuật toán, e là dữ liệu vào của bài toán đã được mã hoá bằng cách nào đó. Thuật toán A tính trên dữ liệu vào e phải trả một giá nhất định. Ta ký hiệu:

$\mathbf{t_A}(e)$ là giá thời gian và $\mathbf{l_A}(e)$ là giá bộ nhớ.

2) Độ phức tạp về bộ nhớ (*Trong trường hợp xấu nhất*):

$\mathbf{L_A(n) = max\{ l_A(e), với\ |e| \le n\}}$, **n** là "kích thước" đầu vào của thuật toán.

3) Độ phức tạp thời gian (*Trong trường hợp xấu nhất*):

$\mathbf{T_A(n) = max\{ t_A(e), với\ |e| \le n\}}$.

4) Độ phức tạp tiệm cận*:** Độ phức tạp PT(n) được gọi là ***tiệm cận tới hàm f(n), ký hiệu O(f(n)) nếu $\exists$ các số n_0 , c mà $PT(n) \le c.f(n)$, $\forall n \ge n_0$.

5) Độ phức tạp đa thức:

Độ phức tạp PT(n) được gọi ***đa thức***, nếu nó ***tiệm cận tới đa thức p(n)***

6) Thuật toán đa thức: Thuật toán được gọi là ***đa thức***, nếu độ phức tạp về thời gian (trong trường hợp xấu nhất) của nó là ***đa thức***.

Nói cách khác:

+ Thuật toán ***thời gian đa thức*** là thuật toán có độ phức tạp thời gian $\mathbf{O(n^t)}$,
trong đó **t** là hằng số.

+ Thuật toán ***thời gian hàm mũ*** là thuật toán có độ phức tạp thời gian $\mathbf{O(t^{f(n)})}$,
trong đó **t** là hằng số và **f(n)** là đa thức của n.

* ***Thời gian chạy của các lớp thuật toán khác nhau:***

Độ phức tạp	Số phép tính($n=10^6$)	Thời gian(10^6ptính/s)
O(1)	1	1micro giây
O(n)	10^6	1 giây
$O(n^2)$	10^{12}	11,6 ngày
$O(n^3)$	10^{18}	32 000 năm
$O(2^n)$	$10^{301\,030}$	$10^{301\,006}$ tuổi của vũ trụ

Chú ý:- Có người cho rằng ngày nay máy tính với tốc độ rất lớn, không cần quan tâm nhiều tới thuật toán nhanh, chúng tôi xin dẫn một ví dụ đã được kiểm chứng.

- Bài toán xử lý n đối tượng, có ba thuật toán với 3 mức phức tạp khác nhau sẽ chịu

3 hậu quả như sau: ***Sau 1 giờ:***

Thuật toán A có độ phức tạp O(n) : 3,6 triệu đối tượng.

Thuật toán B có độ phức tạp O(n log n) : 0,2 triệu đối tượng.

Thuật toán C có độ phức tạp $O(2^n)$: 21 đối tượng.

2.3.3. Phân lớp bài toán theo độ phức tạp

2.3.3.1. Các khái niệm.

1) Khái niệm *"Dẫn về được".*

Bài toán **B** được gọi là "***Dẫn về được***" bài toán A một cách ***đa thức***, ký hiệu: **B ∝ A,** nếu có thuật toán đơn định đa thức để giải bài toán A, thì cũng có thuật toán đơn định đa thức để giải bài toán **B**.

Nghĩa là: Bài toán A "khó hơn" bài toán **B**, hay **B** "dễ" hơn **A**, **B** được diễn đạt bằng ngôn ngữ của bài toán **A,** hay có thể hiểu **B** là trường hợp riêng của **A**.

Vậy nếu giải được bài toán **A** thì cũng sẽ giải được bài toán **B**.

Quan hệ ∝ có tính chất bắc cầu: NếuC ∝ B và B ∝ A thì C ∝ A.

2) Khái niệm *"Khó tương đương".*

Bài toán A gọi là "khó tương đương" bài toán B, ký hiệu **A ~ B**,

nếu : A ∝ B và B ∝ A

2.3.3.2. Các lớp bài toán

1) Lớp bài toán P, NP**.**

Ký hiệu:

P là lớp bài toán giải được bằng thuật toán đơn định, đa thức (Polynomial)

NP là lớp bài toán giải được bằng thuật toán không đơn định, đa thức.

Theo định nghĩa ta có P ⊂ NP.

Hiện nay người ta chưa biết được **P ≠ NP ?**

2) Lớp Bài toán NP- Hard.

Bài toán **A** được gọi là ***NP - hard*** (NP- khó) nếu ∀ **L** ∈ NP đều là **L ∝ A**.

Lớp bài toán NP - hard bao gồm tất cả những bài toán NP - hard.

Bài toán NP – hard có thể nằm ***trong*** hoặc ***ngoài*** lớp NP.

3) Lớp bài toán NP- Complete.

(1) *Bài toán NP- Complete.*

Bài toán A được gọi là NP - Complete (NP- đầy đủ) nếu A là ***NP – Hard*** và A$\in$ **NP**.

Bài toán NP – Complete là bài toán NP - hard nằm trong lớp NP.

Lớp bài toán NP - Complete bao gồm tất cả những bài toán NP - Complete.

Lớp NP – Complete là có thực, vì Cook và Karp đã chỉ ra BT đầu tiên thuộc lớp này. Đó là bài toán "thỏa được": SATISFYABILITY.

(2) *Chứng minh bài toán là NP – Hard.*

Cách 1: Theo định nghĩa

Bài toán **A** được gọi là ***NP - hard*** (NP- khó) nếu $\forall$ **L** $\in$ NP đều là **L** $\propto$ **A**.

Chứng minh theo định nghĩa gặp nhiều khó khăn vì phải chứng minh: Mọi bài toán trong NP đều "***dễ hơn***" A.

Theo cách 1, năm 1971 Cook và Karp đã chỉ ra bài toán đầu tiên thuộc lớp NP - Hard, đó là bài toán "thoả được": SATISFYABILITY.

Cách 2

Để chứng minh bài toán **A** là NP – hard, trong thực tế người ta thường dựa vào bài toán **B** nào đó đã được biết là NP - Hard và chứng minh rằng **B** $\propto$ **A**.

Theo tính chất bắc cầu của quan hệ ***"dẫn về"***, A thoả mãn định nghĩa NP -hard. Theo cách hiểu trực quan: **B** đã ***"khó"*** thì A càng ***"khó"***.

2.3.3.3. Phân lớp các bài toán.

1) Cho một bài toán, có 2 khả năng xảy ra: Đã có lời giải hoặc chưa có lời giải.

2) Cho một bài toán đã có lời giải, có 2 khả năng xảy ra:

Giải được bằng thuật toán hay không giải được bằng thuật toán.

3) Cho một bài toán giải được bằng thuật toán, cũng chia thành hai loại:

"Thực tế giải được" và "Thực tế khó giải".

- Bài toán "thực tế giải được" hiểu là nó ***có thể*** giải được bởi thuật toán, xử lý trong thời gian đủ nhanh, thực tế cho phép, đó là thuật toán có độ phức tạp thời gian là "***đa thức***". Bài toán này thuộc loại "***dễ giải***".

- Bài toán "thực tế khó giải" hiểu là nó ***chỉ có thể*** giải được bởi thuật toán, xử lý trong nhiều thời gian, thực tế khó chấp nhận, đó là thuật toán có độ phức tạp thời gian là trên đa thức ("***hàm mũ***") Bài toán này thuộc loại "***khó giải***".

2.3.4. Hàm một phía và hàm cửa sập một phía.

1) Hàm f(x) được gọi là ***hàm một phía*** nếu tính "***xuôi***" y = f(x) thì "***dễ***",

nhưng tính "**ngược**" $x = f^{-1}(y)$ lại rất "**khó**".

Ví dụ:

Hàm $f(x) = \mathbf{g}^{x} \pmod p$, với p là số nguyên tố lớn, (**g** là phần tử nguyên thuỷ mod p) là hàm một phía.

2) Hàm f(x) được gọi là ***hàm cửa sập một phía*** nếu tính y = f(x) thì "dễ", tính $x = f^{-1}(y)$ lại rất "**khó**". Tuy nhiên có ***cửa sập z*** để tính $x = f^{-1}(y)$ là "dễ".

Ví dụ:

Hàm $f(x) = \mathbf{x}^{\mathbf{a}} \pmod n$ (với n là tích của hai số nguyên tố lớn n = p*q) là hàm một phía. Nếu chỉ biết a và n thì tính $x = f^{-1}(y)$ rất "**khó**", nhưng nếu biết ***cửa sập*** p và q, thì tính được $f^{-1}(y)$ là khá "***dễ***".

Chương 3. MÃ HÓA DỮ LIỆU

3.1. Tổng quan về mã hoá dữ liệu

3.1.1. Khái niệm Mã hóa dữ liệu

Để bảo đảm **An toàn thông tin** (**ATTT**) lưu trữ trong máy tính (giữ gìn thông tin cố định) hay bảo đảm An toàn thông tin trên đường truyền tin (trên mạng máy tính), người ta phải "***Che Giấu***" các thông tin này.

"***Che***" thông tin (dữ liệu) hay "***Mã hóa*** " thông tin là ***thay đổi hình dạng*** thông tin gốc, và người khác "**khó**" nhận ra.

"***Giấu***" thông tin (dữ liệu) là ***cất giấu*** thông tin trong bản tin khác, và người khác cũng "**khó**" nhận ra.

Trong chương này chúng ta bàn về "***Mã hóa*** " thông tin.

1) Hệ mã hóa:

Việc mã hoá phải theo quy tắc nhất định, quy tắc đó gọi là ***Hệ mã hóa***.

Hệ mã hóa được định nghĩa là bộ năm (**P**, **C**, **K**, **E**, **D**), trong đó:

P là tập hữu hạn các ***bản rõ*** có thể. **C** là tập hữu hạn các ***bản mã*** có thể.

K là tập hữu hạn các ***khoá*** có thể.

E là tập các hàm lập mã. **D** là tập các hàm giải mã.

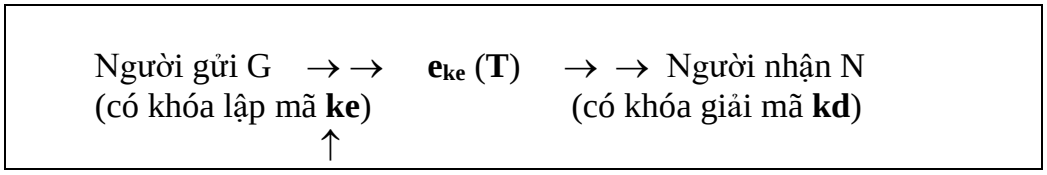
Với khóa lập mã $\mathbf{ke} \in \mathbf{K}$, có hàm lập mã $\mathbf{e_{ke}} \in \mathbf{E}$, $\mathbf{e_{ke}}: \mathbf{P} \rightarrow \mathbf{C}$,

Với khóa giải mã $\mathbf{kd} \in \mathbf{K}$, có hàm giải mã $\mathbf{d_{kd}} \in \mathbf{D}$, $\mathbf{d_{kd}}: \mathbf{C} \rightarrow \mathbf{P}$,

sao cho $\mathbf{d_{kd}(e_{ke}(x)) = x}$, $\forall\ \mathbf{x} \in \mathbf{P}$.

Ơ đây **x** được gọi là ***bản rõ***, $\mathbf{e_{ke}(x)}$ được gọi là ***bản mã***.

2) Mã hóa và Giải mã:



Tin tặc có thể trộm bản mã $\mathbf{e_{ke}}$ (**T**)

Người gửi G muốn gửi bản tin **T** cho người nhận N. Để bảo đảm bí mật, G mã hoá bản tin bằng khóa lập mã **ke**, nhận được bản mã $\mathbf{e_{ke}}$ (**T**), sau đó gửi cho N.

Tin tặc có thể trộm bản mã $\mathbf{e_{ke}}$ (**T**), nhưng cũng "***khó***" hiểu được bản tin gốc **T** nếu không có khoá giải mã **kd**.

Người N nhận được bản mã, họ dùng khoá giải mã **kd**, để giải mã $\mathbf{e_{ke}}$ (**T**), sẽ nhận được bản tin gốc $\mathbf{T = d_{kd}(e_{ke}(T))}$

3.1.2. Phân loại hệ mã hóa

Có nhiều mã hoá tùy theo cách phân loại, sau đây xin giới thiệu một số cách.

***Cách 1: Phân loại mã hoá theo* đặc trưng của khoá.**

Mã hoá khoá riêng, Mã hoá khoá công khai

Hiện có 2 loại mã hóa chính: mã hóa khóa đối xứng và mã hóa khoá công khai.

Hệ mã hóa khóa đối xứng có khóa lập mã và khóa giải mã "giống nhau", theo nghĩa biết được khóa này thì "***dễ***" tính được khóa kia. Vì vậy phải giữ bí mật cả 2 khóa.

Hệ mã hóa khóa công khai có khóa lập mã khác khóa giải mã (**ke** ≠ **kd**), biết được khóa này cũng "***khó***" tính được khóa kia. Vì vậy chỉ cần bí mật khóa giải mã, còn công khai khóa lập mã.

***Cách 2: Phân loại mã hoá theo* đặc trưng xử lý bản rõ.**

Mã hoá khối, Mã hoá dòng

***Cách 3: Phân loại mã hoá theo* ứng dụng đặc trưng.**

Mã hoá đồng cấu.

3.1.2.1. Hệ mã hóa khóa đối xứng

Mã hóa khóa đối xứng là Hệ mã hóa mà biết được khóa lập mã thì có thể "***dễ***" tính được khóa giải mã và ngược lại. Đặc biệt một số Hệ mã hóa có khoá lập mã và khoá giải mã trùng nhau (**ke** = **kd**), như Hệ mã hóa "dịch chuyển" hay DES.

Hệ mã hóa khóa đối xứng còn gọi là ***Hệ mã hóa khoá bí mật***, hay ***khóa riêng***, vì phải giữ bí mật cả 2 khóa. Trước khi dùng Hệ mã hóa khóa đối xứng, người gửi và người

nhận phải thoả thuận thuật toán mã hóa và ***khoá chung*** (lập mã hay giải mã), khoá phải được giữ bí mật. Độ an toàn của Hệ mã hóa loại này ***phụ thuộc vào khoá***.

Ví dụ:

+ *Hệ mã hóa cổ điển* là Mã hóa khóa đối xứng: dễ hiểu, dễ thực thi, nhưng có độ an toàn không cao. Vì giới hạn tính toán chỉ trong phạm vi bảng chữ cái, sử dụng trong bản tin cần mã, ví dụ là Z_{26} nếu dùng các chữ cái tiếng Anh. Với hệ mã hóa cổ điển, nếu biết khoá lập mã hay thuật toán lập mã, có thể "dễ" xác định được bản rõ, vì "dễ" tìm được khoá giải mã.

+ ***Hệ mã hóa* DES** (1973) là Mã hóa khóa đối xứng ***hiện đại***, có độ an toàn cao.

a) Đặc điểm của Hệ mã hóa khóa đối xứng.

Ưu điểm:

Hệ mã hóa khóa đối xứng mã hóa và giải mã ***nhanh hơn*** Hệ mã hóa khóa công khai.

Hạn chế:

1) Mã hóa khóa đối xứng chưa thật an toàn với lý do sau:

Người mã hoá và người giải mã phải có "***chung***" **một khoá**. Khóa phải được giữ bí mật tuyệt đối, vì biết khoá này "***dễ***" xác định được khoá kia và ngược lại.

2) Vấn đề thỏa thuận khoá và quản lý khóa chung là khó khăn và phức tạp. Người gửi và người nhận phải luôn thống nhất với nhau về khoá. Việc thay đổi khoá là rất khó và dễ bị lộ. Khóa chung phải được gửi cho nhau trên kênh an toàn.

Mặt khác khi hai người (lập mã, giải mã) cùng biết "chung" một bí mật, thì càng khó giữ được bí mật !

b) Nơi sử dụng Hệ mã hóa khóa đối xứng.

Hệ mã hóa khóa đối xứng thường được sử dụng trong môi trường mà khoá chung có thể dễ dàng trao chuyển bí mật, chẳng hạn trong cùng một mạng nội bộ.

Hệ mã hóa khóa đối xứng thường dùng để mã hóa những bản tin lớn, vì tốc độ mã hóa và giải mã nhanh hơn Hệ mã hóa khóa công khai.

3.1.2.2. Hệ mã hóa khóa công khai

Hệ mã hóa khóa phi đối xứng là Hệ mã hóa có khóa lập mã và khóa giải mã khác nhau (**ke** ≠ **kd**), biết được khóa này cũng "**khó**" tính được khóa kia.

Hệ mã hóa này còn được gọi là **Hệ mã hoá khóa công khai**, vì:

Khoá lập mã cho ***công khai***, gọi là ***khoá công khai (Public key)***

Khóa giải mã giữ bí mật, còn gọi là ***khóa riêng (Private key)*** hay ***khóa bí mật***.

Một người bất kỳ có thể dùng khoá công khai để mã hoá bản tin, nhưng chỉ người nào có đúng khoá giải mã thì mới có khả năng đọc được bản rõ.

Hệ mã hóa khoá công khai hay ***Hệ mã hóa phi đối xứng*** do Diffie và Hellman phát minh vào những năm 1970.

a) Đặc điểm của Hệ mã khoá công khai.

Ưu điểm:

1) Hệ mã hóa khóa công khai có ưu điểm chủ yếu sau:

Thuật toán được viết một lần, công khai cho nhiều lần dùng, cho nhiều người dùng, họ chỉ cần giữ bí mật khóa riêng của mình.

2) Khi biết các tham số ban đầu của hệ mã hóa, việc tính ra cặp khoá công khai và bí mật phải là "dễ", tức là trong thời gian đa thức.

Người gửi có bản rõ P và khoá công khai, thì "dễ" tạo ra bản mã C.

Người nhận có bản mã C và khoá bí mật, thì "dễ" giải được thành bản rõ P.

3) Người mã hoá dùng khóa công khai, người giải mã giữ khóa bí mật. Khả năng lộ khóa bí mật khó hơn vì chỉ có một người giữ gìn.

Nếu thám mã biết khoá công khai, cố gắng tìm khoá bí mật, thì chúng phải đương đầu với bài toán "khó".

4) Nếu thám mã biết khoá công khai và bản mã C, thì việc tìm ra bản rõ P cũng là bài toán "khó", số phép thử là vô cùng lớn, không khả thi.

Hạn chế:

Hệ mã hóa khóa công khai: mã hóa và giải mã ***chậm hơn*** hệ mã hóa khóa đối xứng.

b) Nơi sử dụng Hệ mã hóa khoá công khai.

Hệ mã hóa khóa công khai thường được sử dụng chủ yếu trên các mạng công khai như Internet, khi mà việc trao chuyển khoá bí mật tương đối khó khăn.

Đặc trưng nổi bật của hệ mã hoá công khai là khoá công khai (public key) và bản mã (ciphertext) đều có thể gửi đi trên một kênh truyền tin ***không an toàn***.

Có biết cả khóa công khai và bản mã, thì thám mã cũng không dễ khám phá được bản rõ.

Nhưng vì có tốc độ mã hóa và giải mã ***chậm***, nên hệ mã hóa khóa công khai chỉ dùng để mã hóa những bản tin ngắn, ví dụ như mã hóa khóa bí mật gửi đi.

Hệ mã hóa khóa công khai thường được sử dụng cho cặp người dùng thỏa thuận khóa bí mật của Hệ mã hóa khóa riêng.

3.2. Hệ mã hoá đối xứng – cổ điển

Khái niệm

Hệ mã hóa đối xứng đã được dùng từ rất sớm, nên còn gọi là ***Hệ mã hóa dối xứng - cổ điển*** (gọi ngắn gọn là ***Hệ mã hóa dối xứng cổ điển***)

Bản mã hay bản rõ là dãy các ký tự Latin.

Lập mã: thực hiện theo các bước sau:

1/ Nhập bản rõ ký tự: RÕ_CHỮ. 2/ Chuyển RÕ_CHỮ ==> RÕ_SỐ.

3/ Chuyển RÕ_SỐ ==> MÃ_SỐ. 4/ Chuyển MÃ_SỐ ==> MÃ_CHỮ.

Giải mã: thực hiện theo các bước sau:

1/ Nhập bản mã ký tự: MÃ_CHỮ. 2/ Chuyển MÃ_CHỮ ==> MÃ_SỐ.

3/ Chuyển MÃ_SỐ ==> RÕ_SỐ. 4/ Chuyển RÕ_SỐ ==> RÕ_CHỮ.

Để chuyển từ CHỮ sang SỐ hay ngược lại từ SỐ trở về CHỮ, người ta theo một qui ước nào đó, ví dụ chữ cái thay bằng số theo **modulo 26** như sau:

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	X	Y	
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	23	24	25	26

Để thực hiện mã hóa hay giải mã với các "***số***", người ta dùng các phép toán số học theo **modulo 26**.

Các hệ mã hóa cổ điển

Mã hóa cổ điển gồm nhiều hệ, ví dụ:

Hệ mã hóa dịch chuyển: Khóa có 1 "chìa". (Thể hiện bằng 1 giá trị)

Hệ mã Affine: Khóa có 2 "chìa". (Thể hiện bằng 2 giá trị)

Hệ mã hóa thay thế: Khóa có 26 "chìa". (Thể hiện bằng 16 giá trị)

Hệ mã hóa VIGENERE: Khóa có m "chìa". (Thể hiện bằng m giá trị)

Hệ mã hóa HILL: Khóa có ma trận "chìa" (chùm chìa khóa)

3.2.1. Hệ mã hóa: Dịch chuyển

Sơ đồ

Đặt $\mathbf{P} = \mathbf{C} = \mathbf{K} = \mathbf{Z_{26}}$. ***Bản mã*** **y** và ***bản rõ*** $\mathbf{x} \in \mathbf{Z_{26}}$.

Với khóa $\mathbf{k} \in \mathbf{K}$, ta định nghĩa:

Hàm Mã hóa: $\mathbf{y} = \mathbf{e_k}(\mathbf{x}) = (\mathbf{x} + \mathbf{k}) \bmod 26$

Hàm Giải mã: $\mathbf{x} = \mathbf{d_k}(\mathbf{y}) = (\mathbf{y} - \mathbf{k}) \bmod 26$

Ví dụ

* Bản rõ chữ: **T O I N A Y T H A V I R U S**

* Chọn khóa **k** = **3**.

* Bản rõ số: **19 14 8 26 13 0 24 26 19 7 0 26 21 8 17 20 18**

* Với phép mã hóa $\mathbf{y} = \mathbf{e_k}(\mathbf{x}) = (\mathbf{x} + \mathbf{k}) \bmod 26 = (\mathbf{x} + \mathbf{3}) \bmod 26$, ta nhận được:

* Bản mã số: **22 17 11 3 16 3 1 3 22 10 3 3 24 11 20 23 21**

* Bản mã chữ: **W R L D Q D B D W K D D Y L U X V**

• Với phép giải mã $\mathbf{x} = \mathbf{d_k}(\mathbf{y}) = (\mathbf{y} - \mathbf{k}) \bmod 26 = (\mathbf{y} - \mathbf{3}) \bmod 26$, ta nhận lại được bản rõ số, sau đó là bản rõ chữ.

Độ an toàn ***Độ an toàn của mã dịch chuyển: Rất thấp.***

Tập khóa **K** chỉ có 26 khóa, nên việc phá khóa (thám mã) có thể thực hiện dễ dàng bằng cách thử kiểm tra từng khóa: **k** = 1, 2, 3, ..., 26.

3.2.2. Hệ mã hóa: Thay thế (Hoán vị toàn cục)

Sơ đồ

Đặt $\mathbf{P} = \mathbf{C} = \mathbf{Z_{26}}$. ***Bản mã*** **y** và ***bản rõ*** $\mathbf{x} \in \mathbf{Z_{26}}$.

Tập khóa **K** là tập mọi hoán vị trên $\mathbf{Z_{26}}$.

Với khóa $\mathbf{k} = \pi \in \mathbf{K}$, tức là 1 hoán vị trên $\mathbf{Z_{26}}$, ta định nghĩa:

Mã hóa: $\mathbf{y} = \mathbf{e_\pi}(\mathbf{x}) = \pi(\mathbf{x})$

Giải mã: $\mathbf{x} = \mathbf{d_\pi}(\mathbf{y}) = \pi^{-1}(\mathbf{y})$

Ví dụ

* Bản rõ chữ: **T O I N A Y T H A V I R U S**

***** Chọn khóa $\mathbf{k} = \pi$ là hoán vị:

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	X	Y	
Y	X	V	U	T	S	R	Q	P	O	N	M	L	K	J	I	H	G	F	E	D	C	B	A	Z

* Mã hóa theo công thức **y = e_π (x) = π (x)**:

* Bản mã chữ: **E J P Z K Y V Z E Q Y Z C P G D F**

* Giải mã theo công thức **x = d_π (y) = π^{-1} (y),** ta nhận lại được bản rõ chữ.

Độ an toàn Độ an toàn của mã thay thế: ***Thuộc loại cao.***

Tập khóa K có 26 ! khóa (> 4. 10^{26}), nên việc phá khóa (thám mã) có thể thực hiện bằng cách duyệt tuần tự **26 !** hoán vị của 26 chữ cái.

Để kiểm tra tất cả **26 !** khóa, tốn rất nhiều thời gian !

Hiện nay với hệ mã này, người ta có phương pháp thám mã khác nhanh hơn.

3.2.3. Hệ mã hóa: AFFINE

Sơ đồ

Đặt **P** = **C** = $\mathbf{Z_{26}}$. ***Bản mã*** **y** và ***bản rõ*** **x** ∈ $\mathbf{Z_{26}}$.

Tập khóa **K** = {(**a**, **b**), với **a, b** ∈ $\mathbf{Z_{26}}$, UCLN(**a**, 26) = 1}

Với khóa **k** = (**a**, **b**) ∈ **K**, ta định nghĩa:

Phép Mã hóa **y = e_k (x) = (a x + b)** mod 26

Phép Giải mã **x = d_k (y) = a^{-1} (y - b)** mod 26

Ví dụ

* Bản rõ chữ: **CHIEUNAYOVUONHOA**

* Chọn khóa **k** = (**a**, **b**) = (**3**, 6)

* Bản rõ số: **x** = 2 7 8 4 20 13 **0** 24 14 21 20 14 13 7 14 **0**

Mã hóa theo công thức **y = e_k (x) = (a x + b)** mod 26 = (**3 x + 6**) mod 26

* Bản mã số: **y** = 12 1 4 18 14 19 **6** 0 22 17 14 22 19 1 22 **6**

* Bản mã chữ: **MBESOTGAWROWTBWG**

Giải mã theo công thức **x = d_k (y) = a^{-1} (y – b)** mod 26

= **3^{-1} (y – 6)** mod 26 = 9 * (**y – 6**) mod 26.

Độ an toàn Độ an toàn của Hệ mã hóa Affine***: Rất thấp.***

+ Điều kiện UCLN(**a**, 26) = 1 để bảo đảm **a** có phần tử nghịch đảo $\mathbf{a}^{-1} \bmod 26$, tức là thuật toán giải mã $\mathbf{d_K}$ luôn thực hiện được.

+ Số lượng $\mathbf{a} \in \mathbf{Z}_{26}$ nguyên tố với 26 là $\phi(26) = \mathbf{12}$, đó là

1, 3, 5, 7 ,9, 11, 15, 17, 19, 21, 23, 25

Các số nghịch đảo theo (mod 26) tương ứng: 1, 9, 21, 15, 3, 19, 7, 23, 11, 5, 17, 25

+ Số lượng $\mathbf{b} \in \mathbf{Z}_{26}$ là **26** .

+ Số các khoá (**a, b**) có thể là **12** * **26** = **312.** Rất ít !

Như vậy việc dò tìm khóa mật khá dễ dàng.

3.2.4. Hệ mã hóa : VIGENERE

Sơ đồ

Đặt $\mathbf{P} = \mathbf{C} = \mathbf{K} = (\mathbf{Z_{26}})^{\mathbf{m}}$, **m** là số nguyên dương, các phép toán thực hiện trong $\mathbf{Z_{26}}$.

Bản mã **Y** và ***bản rõ*** $\mathbf{X} \in (\mathbf{Z_{26}})^{\mathbf{m}}$. Khoá $\mathbf{k} = (\mathbf{k_1}, \mathbf{k_2}, \ldots, \mathbf{k_m})$ gồm **m** phần tử.

Mã hóa $\mathbf{Y} = (\mathbf{y_1}, \mathbf{y_2}, \ldots, \mathbf{y_m}) = \mathbf{e_k}(\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_m}) = (\mathbf{x_1} + \mathbf{k_1}, \mathbf{x_2} + \mathbf{k_2}, \ldots, \mathbf{x_m} + \mathbf{k_m}) \bmod \mathbf{m}$.

Giải mã $\mathbf{X} = (\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_m}) = \mathbf{d_k}(\mathbf{y_1}, \mathbf{y_2}, \ldots, \mathbf{y_m}) = (\mathbf{y_1} - \mathbf{k_1}, \mathbf{y_2} - \mathbf{k_2}, \ldots, \mathbf{y_m} - \mathbf{k_m}) \bmod \mathbf{m}$.

Ví dụ

* Bản rõ chữ: **THISISACRYPTOSYSTEM**

Chọn khoá: **k** = "**KWORD**" = {**10, 22, 14, 17, 3**} với độ dài **m**=5.

* Bản rõ số: S**X** = 19 7 8 18 8 18 0 2 17 24 15 19 14 18 24 18 19 4 12

* Mã hóa:

Chia bản rõ S**X** thành các đoạn, mỗi đoạn gồm **m** =5 số.

Với mỗi đoạn, áp dụng công thức mã hóa, ta nhận được bản mã số.

19 10	7 22	8 14	18 17	8 3	18 10	0 22	2 14	17 17	24 3
3	3	22	9	11	2	22	16	8	1

15 10	19 22	14 14	18 17	24 3	18 10	19 22	4 14	12 17
25	15	2	9	1	2	15	18	3

* Bản mã số: S**Y** = **3 3 22 9 11 2 22 16 8 1 25 15 2 9 1 2 15 18 3**

* Bản mã chữ: **DDWJL CWQIB ZPCJB CPSD**

Độ an toàn Độ an toàn của mã VIGENERE***: Tương đối cao.***

Nếu khoá gồm **m** ký tự khác nhau, mỗi ký tự có thể được ánh xạ vào 1 trong **m** ký tự có thể, do đó hệ mật này được gọi là hệ ***thay thế đa biểu***.

Như vậy số khoá (độ dài **m**) có thể có trong mật Vigenere là $\mathbf{26}^{\mathbf{m}}$.

Nếu dùng phương pháp "tấn công vét cạn", thám mã phải kiểm tra $\mathbf{26}^{\mathbf{m}}$ khóa.

Hiện nay với hệ mã này, người ta có phương pháp thám mã khác nhanh hơn.

3.2.5. Hệ mã hóa: Hoán vị cục bộ.

Sơ đồ

Đặt $\mathbf{P} = \mathbf{C} = \mathbf{Z_{26}}^{\mathbf{m}}$, **m** là số nguyên dương. ***Bản mã*** **Y** và ***bản rõ*** $\mathbf{X} \in (\mathbf{Z_{26}})^{\mathbf{m}}$.

Tập khóa *K* là tập tất cả các hoán vị của {1, 2, …., **m**}.

Với mỗi khoá $\mathbf{k} = \pi \in K$, $\mathbf{k} = (\mathbf{k_1}, \mathbf{k_2}, \ldots, \mathbf{k_m})$ gồm **m** phần tử, ta định nghĩa:

* Mã hóa $\mathbf{Y} = (\mathbf{y_1}, \mathbf{y_2}, \ldots, \mathbf{y_m}) = \mathbf{e_k}\,(\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_m}) = (\mathbf{x_{k(1)}}, \mathbf{x_{k(2)}}, \ldots, \mathbf{x_{k(m)}})$

* Giải mã $\mathbf{X} = (\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_m}) = \mathbf{d_k}\,(\mathbf{y_1}, \mathbf{y_2}, \ldots, \mathbf{y_m}) = (\mathbf{y_{k(1)}}^{-1}, \mathbf{y_{k(2)}}^{-1}, \ldots, \mathbf{y_{k(m)}}^{-1})$

Trong đó $\mathbf{k}^{-1} = \pi^{-1}$ là hoán vị ngược của π.

Ví dụ

* Bản rõ chữ **CX = SHESEL ISSEAS HELLSB YTHESE ASHO**

Đặt $\mathbf{P} = \mathbf{C} = \mathbf{Z_{26}}^{\mathbf{m}}$, trong đó **m** = **6**.

Chọn khoá **k** là một hoán vị π của (1, 2, 3, 4, 5, 6):

1	2	3	4	5	6
3	5	1	6	4	2

Hoán vị ngược là π^{-1} là :

1	2	3	4	5	6
3	6	1	5	2	4

* Mã hóa: Tách bản rõ thành từng nhóm 6 kí tự:

SHESEL | ISSEAS | HELLSB | YTHESE | ASHO

Với mỗi nhóm 6 ký tự, sắp xếp lại các chữ theo hoán vị **π**, ta nhận được:

EESLSH | SALSES | LSHBLE | HSYEET | HRAE

* Bản mã chữ: **CY** = **EESLSHSALSES LSHBLEHSYEETHRAE**

* Dùng hoán vị ngược π^{-1}, ta sẽ thu được bản rõ **CX**.

Độ an toàn

Nếu dùng phương pháp "tấn công vét cạn", thám mã phải kiểm tra số khóa có thể là:

1 ! + 2! + 3 ! + … + m ! trong đó m ≤ 26.

Hiện nay với hệ mã này, người ta có phương pháp thám mã khác nhanh hơn.

3.2.6. Hệ mã hóa: HILL.

Sơ đồ Lester S. Hill đưa ra năm 1929.

Đặt **P** = **C** = $\mathbf{Z_{26}}^{\mathbf{m}}$, **m** là số nguyên dương. ***Bản mã*** **Y** và ***bản rõ*** **X** ∈ $(\mathbf{Z_{26}})^{\mathbf{m}}$.

Tập khóa K = {**K** ∈ $\mathbf{Z}_{26}^{\mathbf{m*m}}$ / **det** (**K**, 26) = **1**}. (**K** phải có $\mathbf{K}^{-1}$)

Mỗi khóa **K** là một "***Chùm chìa khóa***" (một Ma trận "Các chìa khóa")

Với mỗi **K** ∈ K , định nghĩa:

* Hàm lập mã: $\mathbf{Y} = (y_1, y_2, \ldots, y_m) = \mathbf{e}_k (x_1, x_2, \ldots, x_m) = (x_1, x_2, \ldots, x_m) * \mathbf{K}$

* Hàm giải mã: $\mathbf{X} = (x_1, x_2, \ldots, x_m) = \mathbf{d}_k (y_1, y_2, \ldots, y_m) = (y_1, y_2, \ldots, y_m) * \mathbf{K}^{-1}$

Ví dụ

* Bản rõ chữ: **TUDO**

Chọn **m** = 2, khóa **K** = $\begin{pmatrix} 11 & 8 \\ 3 & 7 \end{pmatrix}$, bảo đảm **det** (**K**, 26) = **1,** tính $\mathbf{K}^{-1} = \begin{matrix} 7 & 18 \\ 23 & 11 \end{matrix}$

* Bản rõ số: **19 20 | 13 14**

x_1 x_2 | x_1 x_2

Với mỗi bộ rõ số (x_1 , x_2), theo hàm lập mã (y_1 , y_2) = (x_1 , x_2) * **K**, ta tính được:

$y_1 = 11 * x_1 + 3 * x_2$, $y_2 = 8 * x_1 + 7 * x_2$

* Bản mã số: **9 6 | 23 18**

* Bản mã chữ: **FGXS**

Độ an toàn

Nếu dùng phương pháp "tấn công vét cạn", thám mã phải kiểm tra số khóa có thể với **m** lần lượt là **2, 3, 4, …** trong đó **m** lớn nhất là bằng độ dài bản rõ.

3.3. Hệ mã hoá đối xứng DES

3.3.1. Hệ mã hoá DES

3.3.1.1. Giới thiệu

Hiện nay có nhiều hệ mã hóa đối xứng loại mới, mục này trình bày Chuẩn mã hóa dữ liệu **DES** (Data Encryption Standard). 15/05/ 1973, Uỷ ban tiêu chuẩn quốc gia Mỹ (NBS) (được sự thẩm định của Cục an ninh QG (NAS) đã công bố một khuyến nghị về hệ mã hoá chuẩn.

- Hệ mã hoá phải có độ an toàn cao.
- Hệ mã hoá phải được định nghĩa đầy đủ và dễ hiểu.
- Độ an toàn của Hệ mã hoá phải phải nằm ở Khoá, không nằm ở thuật toán.
- Hệ mã hoá phải sẵn sàng cho mọi người dùng ở các lĩnh vực khác nhau.
- Hệ mã hoá phải xuất khẩu được.

DES được IBM phát triển, là một cải biên của hệ mật LUCIPHER DES, nó được công bố lần đầu tiên vào ngày 17/03/1975. Sau nhiều cuộc tranh luận công khai, cuối cùng DES được công nhận như một chuẩn liên bang vào ngày 23/11/1976 và được công bố vào ngày 15/01/1977.

Năm 1980, "Cách dùng DES " được công bố. Từ đó chu kỳ 5 năm DES được xem xét lại một lần bởi Uỷ ban tiêu chuẩn quốc gia Mỹ, lần gần đây nhất là 2004.

3.3.1.2. Qui trình mã hóa theo DES.

Giai đoạn 1 : Bản Rõ chữ =====➔ Bản Rõ số (Dạng nhị phân)

Chia thành

Giai đoạn 2 : Bản Rõ số =====➔ Các đoạn 64 bit Rõ số

Giai đoạn 3 : 64 bit Rõ số =====➔ 64 bit Mã số

Kết nối

Giai đoạn 4 : Các đoạn 64 bit Mã số =====➔ Bản Mã số (Dạng nhị phân)

Giai đoạn 5 : Bản Mã số =====➔ Bản Mã chữ

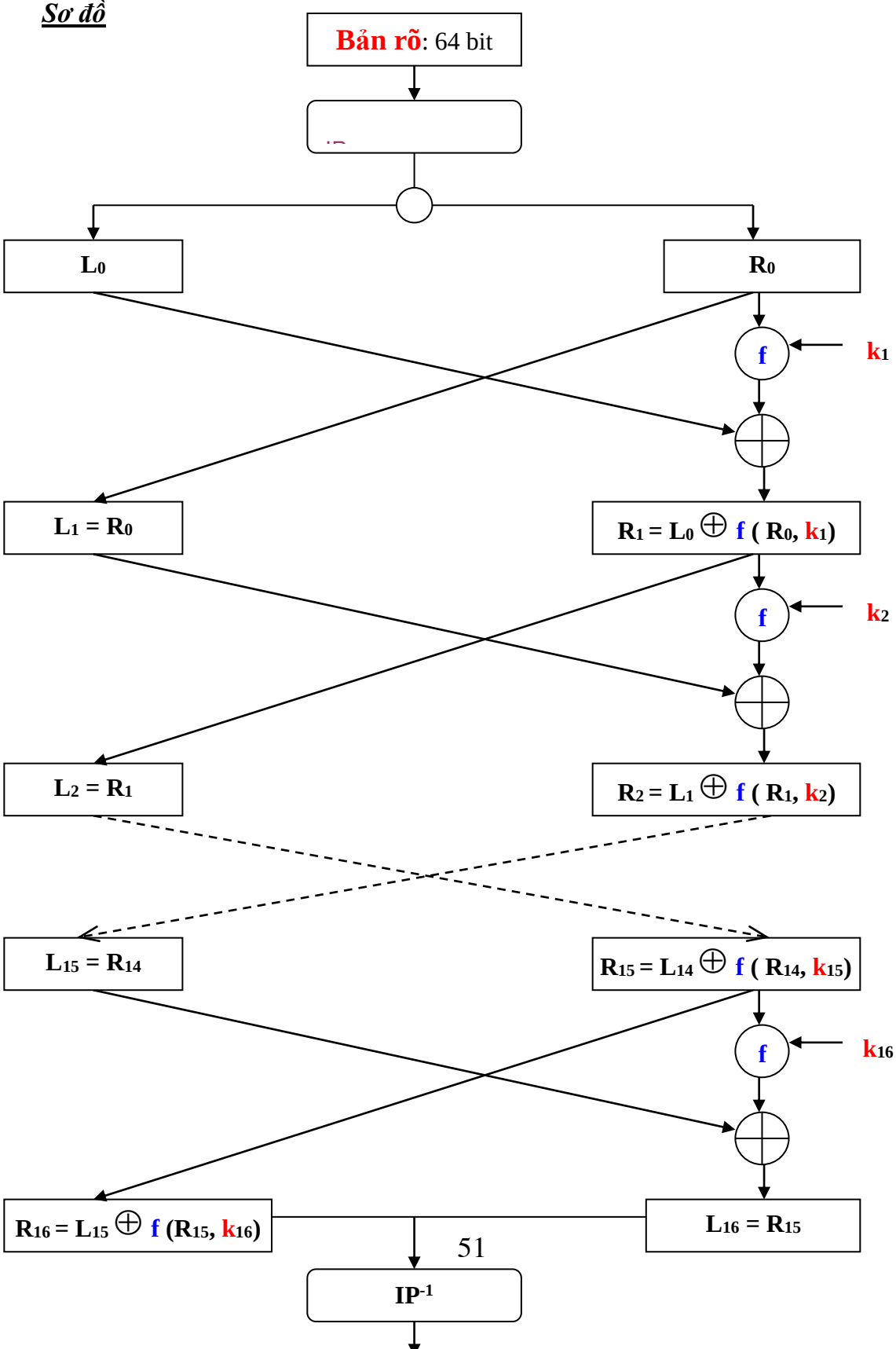
3.3.2. Lập mã và Giải mã DES

3.3.2.1. Qui trình lập mã DES

Thuật toán DES tập trung thực hiện ***Giai đoạn 3*** .của qui trình mã hóa.

Đó là chuyển đổi bản rõ số với ***64*** bit thành bản mã với ***64*** bit.

Sơ đồ



3.3.2.2. Thực hiện mã hóa DES theo Sơ đồ

* Bản rõ là xâu **x** , Bản mã là xâu **y**, Khoá là xâu **K**, đều có độ dài 64 bit.

* Thuật toán mã hóa DES thực hiện qua *3 bước* chính như sau:

Bước 1: Bản rõ **x** được hoán vị theo phép hoán vị **IP**, thành **IP** (**x**)

IP (**x**) = $\mathbf{L_0 R_0}$, trong đó $\mathbf{L_0}$ là 32 bit đầu (Left), $\mathbf{R_0}$ là 32 bit cuối (Right)

(**IP** (**x**) tách thành $\mathbf{L_0 R_0}$)

Bước 2: Thực hiện **16** vòng mã hoá với những phép toán giống nhau.

Dữ liệu được kết hợp với khoá thông qua hàm **f** :

$$\mathbf{L_i = R_{i-1}}, \quad \mathbf{R_i = L_{i-1} \oplus f(R_{i-1}, k_i)},$$ trong đó:

$\oplus$ là phép toán ***hoặc loại trừ*** của hai xâu bit (cộng theo modulo 2)

$\mathbf{k_1, k_2, ..., k_{16}}$ là các ***khoá con*** (48 bit) được tính từ khóa gốc **K**.

Bước 3: Thực hiện phép hoán vị ngược $\mathbf{IP^{-1}}$ cho xâu $\mathbf{R_{16}L_{16}}$, thu được bản mã **y**.

$\mathbf{y = IP^{-1}(R_{16}, L_{16})}$ (Lưu ý thứ tự bit $\mathbf{R_{16}}$ và $\mathbf{L_{16}}$)

* Bảng hoán vị ban đầu **IP**:

+ bit 1 của **IP**(x) là bit **58** của x.
+ bit 2 của **IP**(x) là bit **50** của x.

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

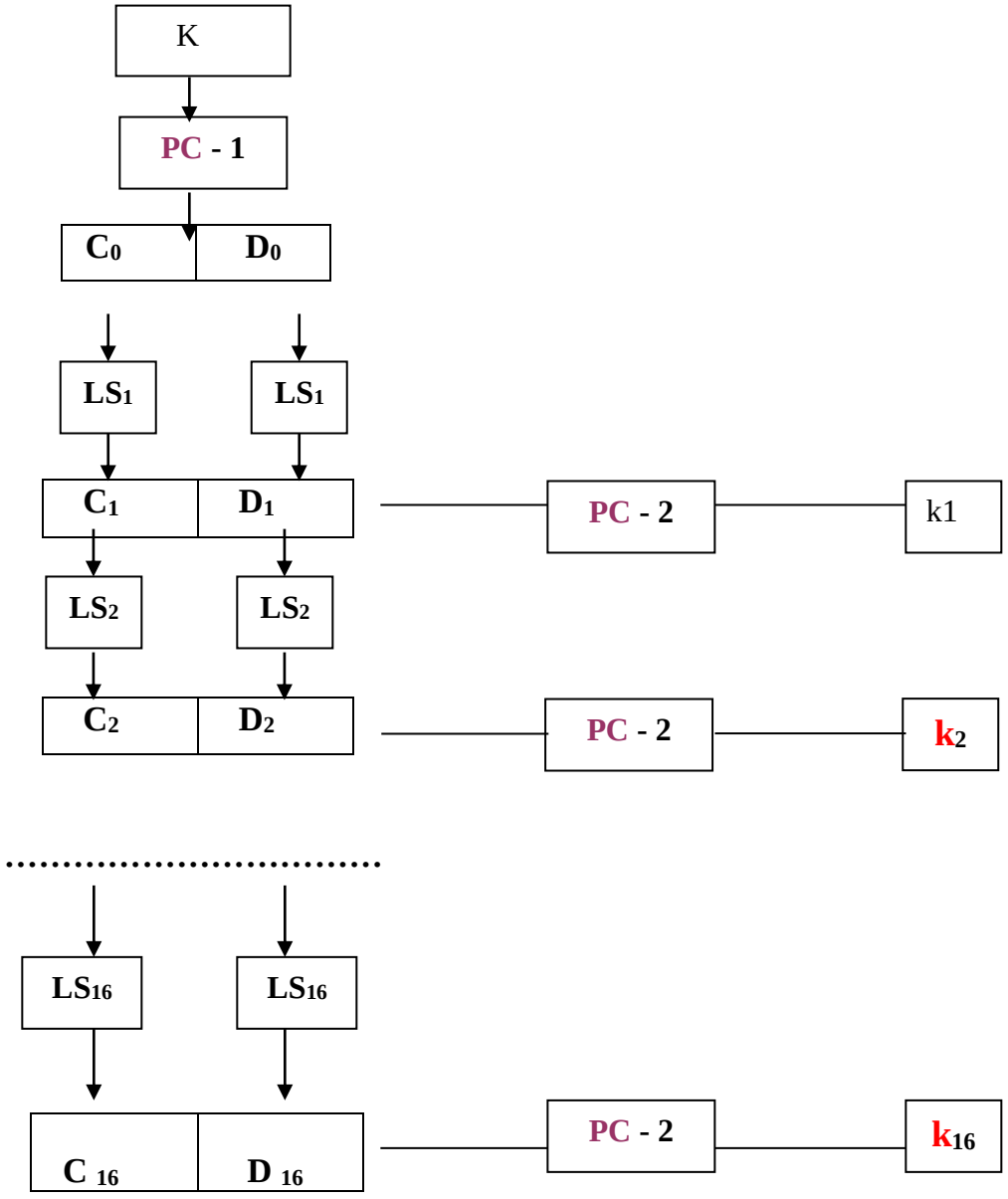
* Bảng hoán vị cuối cùng $\mathbf{IP^{-1}}$:

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

3.3.2.3. Tính các khóa con $k_1, k_2, \ldots, k_{16}$ ***từ khóa gốc*** **K**.

Sơ đồ



* ***Tính*** khoỏ k_i (**48** bit):

1) Khoá **K** là xâu dài 64 bit, trong đó 56 bit là khoá và 8 bit để kiểm tra tính chẵn lẻ nhằm phát hiện sai, các bit này không tham gia vào quá trình tính toán.

Các bit kiểm tra tính chẵn lẻ nằm ở vị trí 8, 16, 24,…, 64 được xác định, sao cho mỗi byte chứa ***một số lẻ*** các số **1**. Bởi vậy mỗi sai sót đơn lẻ được xác định trong mỗi nhóm 8 bit.

2) Tính khoá $\mathbf{k_i}$ như sau:

+ Với khoá **K** độ dài 64 bit, ta loại bỏ các bit kiểm tra tính chẵn lẻ, hoán vị 56 bit còn lại theo phép hoán vị **PC-1**:

$\mathbf{PC\text{-}1\ (K)} = \mathbf{C_0\, D_0}$

Trong đó $\mathbf{C_0}$ là 28 bit đầu, $\mathbf{D_0}$ là 28 bit cuối cùng của **PC-1(K)**

+ Với i = 1, 2, ... , 16, ta tính: $\mathbf{C_i} = \mathbf{LS_i}\,(\,\mathbf{C_{i-1}}\,)$, $\mathbf{D_i} = \mathbf{LS_i}\,(\,\mathbf{D_{i-1}}\,)$

Trong đó $\mathbf{LS_i}$ là phép chuyển dịch vòng sang trái:

Dịch **1** ***vị trí*** nếu i = 1, 2, 9, 16. Dịch **2** ***vị trí*** với những giá trị **i** khác.

+ Với i = 1, 2, ... , 16, khóa $\mathbf{k_i}$ được tính theo phép hoán vị **PC-2** từ $\mathbf{C_i\, D_i}$:

$\mathbf{k_i} = \mathbf{PC\text{-}2}\,(\mathbf{C_i\, D_i})$ (**48** bit)

* Phép hoán vị **PC - 1**:

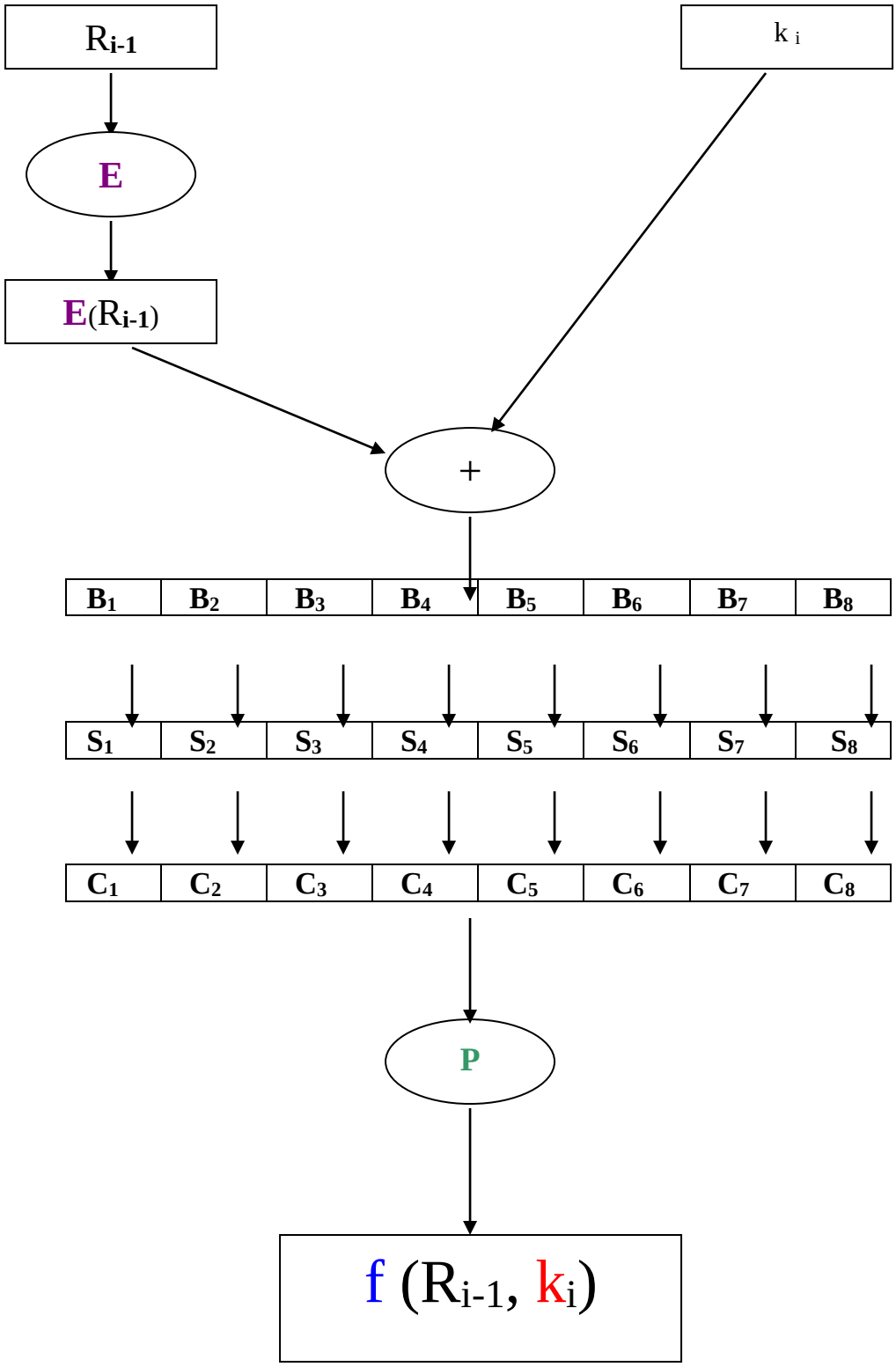
```
57 49 41 33 25 17  9
 1 58 50 42 34 26 18
10  2 59 51 43 35 27
19 11  3 60 52 44 36
63 55 47 39 31 23 15
 7 62 54 46 38 30 22
14  6 61 53 45 37 29
21 13  5 28 20 12  4
```

* Phép hoán vị **PC - 2**:

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

3.3.2.4. Tính hàm $f(R_{i-1}, k_i)$

Sơ đồ



**** Tính hàm* f (R_{i-1}, k_i)**

Để cho đơn giản, ta không ghi chỉ số **i-1**, **i**, và mô tả cách tính **f (R , k)**:

1) Mở rộng xâu **R** (32 bit) thành xâu 48 bit, theo hàm mở rộng **E**:

E: **R** (32 bit) **--- >** **E(R)** (48 bit)

E(R) gồm 32 bit của cũ của **R** và 16 bit của **R** xuất hiện lần thứ 2.

2) Tính **E(R)** $\oplus$ **k**, trong đó **E(R)** (48 bit) và **k** (48 bit)

Kết quả gồm 8 xâu B_j, mỗi xâu B_j có 6 bit (8*6 = 48):

B = B_1 B_2 B_3 B_4 B_5 B_6 B_7 B_8.

3) Tính $C_j = S_j(B_j)$, j = 1,… , 8. Dùng 8 bảng $S_1, S_2, \ldots, S_8$.

S_j là bảng cố định với **r** * **c** số nguyên từ 0 -> 15, ($0 \le r \le 3$, $0 \le c \le 15$)

S_j thể hiện việc thay thế mỗi B_j thành C_j (C_j là xâu 4 bit) theo qui tắc sau:

* Giả sử $B_j = b_1 b_2 b_3 b_4 b_5 b_6$. (6 bit)

+ $b_1 b_6$ xác định biểu diễn nhị phân của hàng **r** trong S_j ($0 \le r \le 3$)

+ $b_2 b_3 b_4 b_5$ xác định biểu diễn nhị phân của cột **c** trong S_j ($0 \le c \le 15$)

Xâu C_j (4 bit) được định nghĩa là biểu diễn nhị phân của phần tử $S_j(r, c)$

4) Thực hiện 8 lần bước 3), ta nhận được xâu **C** = C_1 C_2 … C_8 (32 bit)

Sau hoán vị **P**, cho kết quả **P** (**C**), đó chính là **f** (**R**, **k**)

* Phép 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

* Phép 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

**** Các bảng* S_1 , S_2 , … , S_8:**

S_1

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

S_2

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

S_3

```
13 18 | 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
    |-----------------------------------------
 0 0 |10 0 9 14 6 3 15 5 1 13 12 7 11 4 2 8
 0 1 |13 7 0 9 3 4 6 10 2 8 5 14 12 11 15 1
 1 0 |13 6 4 9 8 15 3 0 11 1 2 12 5 10 14 7
 1 1 | 1 10 13 0 6 9 8 7 4 15 14 3 11 5 2 12
```

S_4

```
19 24 | 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
    |-----------------------------------------
 0 0 | 7 13 14 3 0 6 9 10 1 2 8 5 11 12 4 15
 0 1 |13 8 11 5 6 15 0 3 4 7 2 12 1 10 14 9
 1 0 |10 6 9 0 12 11 7 13 15 1 3 14 5 2 8 4
 1 1 | 3 15 0 6 10 1 13 8 9 4 5 11 12 7 2 14
```

S_5

```
25 30 | 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
    |-----------------------------------------
 0 0 | 2 12 4 1 7 10 11 6 8 5 3 15 13 0 14 9
 0 1 |14 11 2 12 4 7 13 1 5 0 15 10 3 9 8 6
 1 0 | 4 2 1 11 10 13 7 8 15 9 12 5 6 3 0 14
 1 1 |11 8 12 7 1 14 2 13 6 15 0 9 10 4 5 3
```

S_6

```
31 36 | 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
    |-----------------------------------------
 0 0 |12 1 10 15 9 2 6 8 0 13 3 4 14 7 5 11
 0 1 |10 15 4 2 7 12 9 5 6 1 13 14 0 11 3 8
 1 0 | 9 14 15 5 2 8 12 3 7 0 4 10 1 13 11 6
 1 1 | 4 3 2 12 9 5 15 10 11 14 1 7 6 0 8 13
```

S_7

```
37 42 |  0  1 2  3  4 5  6  7  8  9 10 11 12 13 14 15
    |-------------------------------------------------
 0  0 |  4 11 2 14 15 0  8 13  3 12 9  7  5 10  6  1
 0  1 | 13 0 11  7  4 9  1 10 14  3 5 12  2 15  8 6
1  0 | 1  4 11 13 12  3  7 14 10 15 6  8  0  5  9  2
1  1 |  6 11 13  8  1 4 10  7  9  5 0 15 14  2  3 12
```

S_8

```
43 48 | 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
   |-----------------------------------------
 0 0 |13 2 8 4 6 15 11 1 10 9 3 14 5 0 12 7
 0 1 | 1 15 13 8 10 3 7 4 12 5 6 11 0 14 9 2
 1 0 | 7 11 4 1 9 12 14 2 0 6 10 13 15 3 5 8
 1 1 | 2 1 14 7 4 10 8 13 15 12 9 0 3 5 6 11
```

**** Qui định lập bảng*** **S_j**:

- Mỗi hàng của bảng **S** phải là một hoán vị của 0, 1, …,15.

- Không có bảng S nào là hàm tuyến tính hay Apphin của các đầu vào của nó.

- Thay đổi 1 bit vào ở một bảng S, sẽ gây ra sự thay đổi ít nhất 2 bit ra của nó.

- Nếu 2 xâu vào của một bảng S giống nhau ở 2 bit đầu và 2 bit cuối, thì 2 xâu ra phải khác nhau ít nhất tại 2 bit.

- Nếu 2 xâu vào của một bảng S khác nhau ở 2 bit đầu và giống nhau ở 2 bit cuối, thì 2 xâu ra phải khác nhau.

- Với mỗi bảng S, nếu cố định 1 bit vào xét giá trị của 1 bit ra nào đó, thì số các xâu vào tạo ra giá trị 0 ở bit ra đó cũng phải xấp xỉ bằng số các xâu vào tạo ra giá trị 1 ở bit ra đó.

3.3.2.5. Qui trình giải mã DES

Qui trình giải mã của DES tương tự như qui trình lập mã, nhưng theo dùng các khóa thứ tự ngược lại: **k_{16} , k_{15}, … , k_1** .

Xuất phát (đầu vào) từ bản mã **y**, kết quả (đầu ra) là bản ró **x**.

3.3.2.6. Ví dụ

Bản rõ **X** = **0123456789ABCDEF** = 0000

0001 0010 0011 0100 0101 0110 0111 1000 1001 1010 1011 1**1**00 1101 1**1**10 1111

50 **58**

Bước 1: Bản rõ **x** được hoán vị theo phép hoán vị **IP**, thành **IP** (**x**)

IP (**x**) = **L_0** **R_0** , trong đó **L_0** là 32 bit đầu (Left), **R_0** là 32 bit cuối (Right)

(**IP** (**x**) tách thành **L_0** **R_0**)

L_0 = **11**00 1100 0000 0000 1100 1001 1111 1111 (32 bit)

R_0 = 1111 0000 1010 1010 1111 0000 1010 1010 (32 bit)

Ví dụ: theo hoán vị **IP**, bit **1** của $\mathbf{L}_0$ là bit **58** của **x**, bit **2** của $\mathbf{L}_0$ là bit **50** của **x**.

Bước 2: Thực hiện **16** vòng mã hoá với những phép toán giống nhau.

Dữ liệu được kết hợp với khoá thông qua hàm **f** :

$$\mathbf{L}_i = \mathbf{R}_{i-1}, \quad \mathbf{R}_i = \mathbf{L}_{i-1} \oplus \mathbf{f}(\mathbf{R}_{i-1}, \mathbf{k}_i),$$ trong đó:

$\mathbf{k_1}, \mathbf{k_2}, ..., \mathbf{k_{16}}$ là các ***khoá con*** (48 bit) được tính từ khóa gốc **K**.

a) Tính khóa con $\mathbf{k_1}$ (48 bit) từ khóa gốc **K** = **133457799BBCDFF1** (64 bit)=

0001 0011 0011 0100 0101 0111 0111 1001 1001 1011 1011 1100 1101 1111 1111 0001

* Hoán vị **PC-1**: $\mathbf{K} \rightarrow \mathbf{C}_0\mathbf{D}_0$ (Từ **K** qua PC-1, nhận được $\mathbf{C}_0\mathbf{D}_0$)

$\mathbf{C}_0$ = 1111000 0110011 0010101 0101111 (28 bit)

$\mathbf{D}_0$ = 0101010 1011001 1001111 0001111 (28 bit)

$\mathbf{C_1} = LS_1(\mathbf{C}_0)$ = 1110000 1100110 0101010 1011111 (28 bit)

$\mathbf{D_1} = LS_1(\mathbf{D}_0)$ = 1010101 0110011 0011110 0011110 (28 bit)

* Hoán vi **PC-2**: $\mathbf{C_1}\mathbf{D_1} \rightarrow \mathbf{k_1}$ (48 bit)

$\mathbf{k_1}$ = **000110 110000 001011 101111 111111 000111 000001 110010**

b) Tính hàm $\mathbf{f}(\mathbf{R}_0, \mathbf{k}_1)$

+ Theo bước 1: $\mathbf{R}_0$ = 1111 0000 1010 1010 1111 0000 1010 1010 (32 bit)

1) Mở rộng xâu $\mathbf{R}_0$ (32 bit) thành xâu $\mathbf{E}(\mathbf{R}_0)$ (48 bit), theo hàm mở rộng **E**:

+ Hoán vị **E**: $\mathbf{R}_0 \rightarrow \mathbf{E}(\mathbf{R}_0)$:

$\mathbf{E}(\mathbf{R}_0)$ = 011110 100001 010101 010101 011110 100001 010101 010101 (48 bit)

+ Theo a):

$\mathbf{k_1}$ = 000110 110000 001011 101111 111111 000111 000001 110010 (48 bit)

2) Tính $\mathbf{E}(\mathbf{R}_0) \oplus \mathbf{k_1} = \mathbf{B_1}\ \mathbf{B_2}\ \mathbf{B_3}\ \mathbf{B_4}\ \mathbf{B_5}\ \mathbf{B_6}\ \mathbf{B_7}\ \mathbf{B_8}$ (48 bit)

011000 010001 100010 110010 100001 100110 010100 100111

3) Tính $\mathbf{C_1} = \mathbf{S_1}(\mathbf{B_1})$, dùng bảng $\mathbf{S_1}$.

$\mathbf{S_1}$ thể hiện việc thay thế $\mathbf{B_1}$ (6 bit) thành $\mathbf{C_1}$ (4 bit) theo qui tắc sau:

$\mathbf{B_1} = \mathbf{b_1}\ \mathbf{b_2}\ \mathbf{b_3}\ \mathbf{b_4}\ \mathbf{b_5}\ \mathbf{b_6}$ = 011000

+ $\mathbf{b_1}\mathbf{b_6}$ = $(\mathbf{00})_2 = (\mathbf{00})_{10}$ = Hàng **0** trong $\mathbf{S_1}$.

+ $\mathbf{b_2}\ \mathbf{b_3}\ \mathbf{b_4}\ \mathbf{b_5} = (\mathbf{1100})_2 = (\mathbf{12})_{10}$ = Cột **12** trong $\mathbf{S_1}$.

Xâu $\mathbf{C_1}$ (4 bit) được định nghĩa là biểu diễn nhị phân của phần tử $\mathbf{S}_1(\mathbf{0, 12})$

$\mathbf{C_1} = \mathbf{S}_1(\mathbf{0, 12}) = (\mathbf{5})_{10} = (\mathbf{0101})_2$

+ Tương tự ta tính được $\mathbf{C_j}$, j = 2, 3, …, 8.

4) Thực hiện 8 lần 3), ta nhận được xâu $\mathbf{C} = \mathbf{C_1}\,\mathbf{C_2} \ldots \mathbf{C_8}$ (32 bit)

C = 0101 1100 1000 0010 1011 0101 1001 0111

Sau hoán vị **P**, cho kết quả **P** (**C**), đó chính là $\mathbf{f}(\mathbf{R}_0, \mathbf{k}_1)$

$\mathbf{f}(\mathbf{R}_0, \mathbf{k}_1) = \mathbf{P}(\mathbf{C})$ = 0010 0011 0100 1010 1010 1001 1011 1011

Bước 3: Kết quả là bản mã **85E813540F0AB405**

3.3.3. Độ an toàn của Hệ mã hóa DES

1) Độ an toàn của Hệ mã hóa DES có liên quan đến các bảng $\mathbf{S_j}$:

Ngoại trừ các bảng **S**, mọi tính toán trong DES đều tuyến tính, tức là việc tính phép hoặc loại trừ của hai đầu ra cũng giống như phép hoặc loại trừ của hai đầu vào, rồi tính toán đầu ra.

Các bảng S chứa đựng nhiều thành phần phi tuyến của hệ mật, là yếu tố quan trọng nhất đối với độ mật của hệ thống.

Khi mới xây dựng hệ mật DES, thì tiêu chuẩn xây dựng các hộp S không được biết đầy đủ. Và có thể các hộp S này có thể chứa các "***cửa sập***" được giấu kín. Và đó cũng là một điểm đảm bảo tính bảo mật của hệ DES.

2) Hạn chế của DES chính là kích thước không gian khoá:

Số khóa có thể là $\mathbf{2^{56}}$, không gian này là nhỏ để đảm bảo an toàn thực sự. Nhiều thiết bị chuyên dụng đã được đề xuất nhằm phục vụ cho phép tấn công với bản rõ đã biết. Phép tấn công này chủ yếu thực hiện theo phương pháp "vét cạn".

Tức là với bản rõ **x** và bản mã **y** tương ứng (64 bit), mỗi khoá có thể đều được kiểm tra cho tới khi tìm được một khoá **K** thoả mãn $\mathbf{e_K}(\mathbf{x}) = \mathbf{y}$.

3.4. Hệ mã hoá khoá công khai

3.4.1. Hệ mã hóa RSA.

Sơ đồ (Rivest, Shamir, Adleman đề xuất năm 1977)

****Tạo cặp khóa*** (***bí mật, công khai***) ***(***a, b***)*** :

Chọn bí mật số nguyên tố lớn **p**, **q**, tính $\mathbf{n} = \mathbf{p} * \mathbf{q}$, công khai **n**, đặt $\mathbf{P} = \mathbf{C} = \mathbf{Z_n}$

Tính bí mật $\phi(\mathbf{n}) = (\mathbf{p-1})(\mathbf{q-1})$ Chọn khóa công khai $\mathbf{b} < \phi(\mathbf{n})$, nguyên tố với $\phi(\mathbf{n})$

Khóa bí mật **a** là phần tử nghịch đảo của **b** theo mod ϕ(n): **a*****b** ≡ 1 (mod ϕ(n)

Tập cặp khóa (bí mật, công khai) **K** = {(**a**, **b**)/ a, b ∈ Z_n , **a*****b** ≡ 1 (mod ϕ(n))}.

Với ***Bản rõ*** **x** ∈ **P** và ***Bản mã*** **y** ∈ **C**, định nghĩa:

* ***Hàm Mã hoá***: **y** = $\mathbf{e_k}$ (**x**) = $\mathbf{x}^{\mathbf{b}}$ mod **n**

* ***Hàm Giải mã***: **x** = $\mathbf{d_k}$ (**y**) = $\mathbf{y}^{\mathbf{a}}$ mod **n**

Ví dụ

* Bản rõ chữ: **R E N A I S S A N C E**

****Sinh khóa***:

Chọn bí mật số nguyên tố **p**= 53, **q**= 61, tính **n** = **p** * **q** = **3233**, công khai **n**.

Đặt **P** = **C** = $\mathbf{Z_n}$, tính bí mật ϕ**(n)** = (**p-1**) (**q-1**) = 52 * 60 = 3120.

+ Chọn khóa công khai **b** là nguyên tố với ϕ**(n)**, tức là ƯCLN(**b**, ϕ**(n)**) = 1,

ví dụ chọn **b** = 71.

+ Khóa bí mật **a** là phần tử nghịch đảo của **b** theo mod ϕ(n): **a*****b** ≡ 1 (mod ϕ(n))

Từ **a*****b** ≡ 1 (mod ϕ(n)), ta nhận được khóa bí mật **a** = **791**.

* Bản rõ số:

R E N A I S S A N C E (Dấu cách)

17 04 13 00 08 18 18 00 13 02 04 26

$\mathbf{m_1}$ $\mathbf{m_2}$ $\mathbf{m_3}$ $\mathbf{m_4}$ $\mathbf{m_5}$ $\mathbf{m_6}$

* Theo phép lập mã: $\mathbf{c_i}$ = $\mathbf{m_i}^{\mathbf{b}}$ mod **n** = $\mathbf{m_i}^{\mathbf{71}}$ mod **3233**, ta nhận được:

* Bản mã số:

$\mathbf{c_1}$ $\mathbf{c_2}$ $\mathbf{c_3}$ $\mathbf{c_4}$ $\mathbf{c_5}$ $\mathbf{c_6}$

3106 0100 0931 2691 1984 2927

* Theo phép giải mã: $\mathbf{m_i}$ = $\mathbf{c_i}^{\mathbf{a}}$ mod **n** = $\mathbf{c_i}^{\mathbf{791}}$ mod 3233, ta nhận lại bản rõ.

Độ an toàn

1) Hệ mã hóa RSA là tất định, tức là với một bản rõ **x** và một khóa bí mật **a**, thì chỉ có một bản mã **y**.

2) Hệ mật RSA an toàn, khi giữ được bí mật khoá giải mã **a**, p, q, ϕ(n)

Nếu biết được p và q, thì thám mã dễ dàng tính được ϕ(n) = (q-1)*(p-1)

Nếu biết được ϕ(n), thì thám mã sẽ tính được **a** theo thuật toán Euclide mở rộng.

Nhưng phân tích n thành tích của p và q là bài toán "khó".

Độ an toàn của Hệ mật RSA dựa vào khả năng giải bài toán phân tích số nguyên dương n thành tích của 2 số nguyên tố lớn p và q.

3.4.2. Hệ mã hóa Elgamal.

Sơ đồ (Elgamal đề xuất năm 1985)

****Tạo cặp khóa*** (***bí mật, công khai***) ***(*****a**, **h*****)*** :

Chọn số nguyên tố **p** sao cho bài toán logarit rời rạc trong $\mathbf{Z_p}$ là "khó" giải. Chọn phần tử nguyên thuỷ $\mathbf{g} \in \mathbf{Z_p}^*$. Đặt $\mathbf{P} = \mathbf{Z}_p^*$, $\mathbf{C} = \mathbf{Z}_p^* \times \mathbf{Z}_p^*$.

Chọn khóa bí mật là $\mathbf{a} \in \mathbf{Z_p}^*$. Tính khóa công khai $\mathbf{h} \equiv \mathbf{g}^{\mathbf{a}} \bmod \mathbf{p}$.

Định nghĩa tập khóa: $\mathcal{K} = \{(\mathbf{p}, \mathbf{g}, \mathbf{a}, \mathbf{h}): \mathbf{h} \equiv \mathbf{g}^{\mathbf{a}} \bmod \mathbf{p}\}$.

Các giá trị **p**, **g**, **h** được công khai, phải giữ bí mật **a**.

Với ***Bản rõ*** $\mathbf{x} \in \mathbf{P}$ và ***Bản mã*** $\mathbf{y} \in \mathbf{C}$, với khóa $\mathbf{k} \in \mathcal{K}$ định nghĩa:

* ***Lập mã***: Chọn ngẫu nhiên bí mật $\mathbf{r} \in \mathbf{Z_{p-1}}$, bản mã là $\mathbf{y} = \mathbf{e_k}(\mathbf{x}, \mathbf{r}) = (\mathbf{y_1}, \mathbf{y_2})$

Trong đó $\mathbf{y_1} = \mathbf{g}^{\mathbf{r}} \bmod \mathbf{p}$ và $\mathbf{y_2} = \mathbf{x} * \mathbf{h}^{\mathbf{r}} \bmod \mathbf{p}$

* ***Giải mã***: $\mathbf{d_k}(\mathbf{y_1}, \mathbf{y_2}) = \mathbf{y_2}(\mathbf{y_1}^{\mathbf{a}})^{-1} \bmod \mathbf{p}$.

Ví dụ * Bản rõ **x** = **1299**.

Chọn **p** = 2579, **g** = **2**, **a** = 765. Tính khóa công khai $\mathbf{h} = \mathbf{2}^{765} \bmod 2579 = \mathbf{949}$.

* Lập mã: Chọn ngẫu nhiên **r** = **853**. Bản mã là **y** = (**435, 2369**), trong đó

$\mathbf{y_1} = 2^{852} \bmod 2579 = \mathbf{435}$ và $\mathbf{y_2} = 1299 * 949^{\mathbf{853}} \bmod 2579 = \mathbf{2396}$

* Giải mã: $\mathbf{x} = \mathbf{y_2}(\mathbf{y_1}^{\mathbf{a}})^{-1} \bmod \mathbf{p} = 2369 * (435^{765})^{-1} \bmod 2579 = \mathbf{1299}$.

Độ an toàn

1) Hệ mã hóa Elgamal là không tất định, tức là với một bản rõ **x** và 1 khóa bí mật **a**, thì có thể có nhiều hơn một bản mã **y**, vì trong công thức lập mã còn có thành phần ngẫu nhiên **r**.

2) Độ an toàn của Hệ mật Elgamal dựa vào khả năng giải bài toán logarit rời rạc trong $\mathbf{Z_p}$. Theo giả thiết trong sơ đồ, thì bài toán này phải là "khó" giải.

Cụ thể như sau: Theo công thức lập mã: $\mathbf{y} = \mathbf{e_k}(\mathbf{x}, \mathbf{r}) = (\mathbf{y_1}, \mathbf{y_2})$, trong đó $\mathbf{y_1} = \mathbf{g}^{\mathbf{r}} \bmod \mathbf{p}$ và $\mathbf{y_2} = \mathbf{x} * \mathbf{h}^{\mathbf{r}} \bmod \mathbf{p}$

Như vậy muốn xác định bản rõ **x** từ công thức $\mathbf{y_2}$, thám mã phải biết được **r**.

Giá trị này có thể tính được từ công thức y_1 , nhưng lại gặp bài toán logarit rời rạc.

BÀI TẬP CHƯƠNG 3. MÃ HOÁ DỮ LIỆU.

Để hiểu cách thức mã hóa và giải mã đối với từng hệ mã hóa cụ thể, bài tập chương 3 tập trung vào việc lập chương trình mã hóa và giải mã cho các hệ mã hóa.

Bài tập

Viết chương trình Mã hóa dữ liệu theo các Hệ mã hoá sau:

1/ Hệ mã hoá Dịch chuyển.

2/ Hệ mã hoá Thay thế.

3/ Hệ mã hoá Hoán vị.

4/ Hệ mã hoá Affine.

5/ Hệ mã hoá Vigenere.

6/ Hệ mã hoá Hill.

7/ Hệ mã hoá RSA.

8/ Hệ mã hoá Elgamal.

9/ Hệ mã hoá Rabin.

10/ **Hệ mã hoá chuẩn DES.**

+ Sơ đồ mã hóa, giải mã.

+ Tính khóa K_i.

+ Tính Hàm f(R_i , K_i)

Mẫu Chương trình

Mỗi chương trình mã hóa phải thực hiện các công việc theo thực đơn sau:

Thực đơn chính.

L. Lập mã.

G. Giải mã.

K. Kết thúc.

L. Thực đơn Lập mã.

1. Nhập bản tin (Xâu ký tự): RÕ_CHỮ.
2. Chuyển RÕ_CHỮ =====> RÕ_SỐ.
3. Chuyển RÕ_SỐ =====> MÃ_SỐ.
4. Chuyển MÃ_SỐ =====> MÃ_CHỮ.

0. Về thực đơn chính.

G. Thực đơn Giải mã.

1. Nhập bản tin (Xâu ký tự): MÃ_CHỮ.
2. Chuyển MÃ_CHỮ =====> MÃ_SỐ.
3. Chuyển MÃ_SỐ =====> RÕ_SỐ.
4. Chuyển RÕ_SỐ =====> RÕ_CHỮ.
0. Về thực đơn chính.

Chương 4. CHỮ KÝ SỐ

4.1. Tổng quan về chữ ký số

4.1.1. Khái niệm "Chữ ký số"

4.1.1.1. Giới thiệu

Để chứng thực nguồn gốc hay hiệu lực của một tài liệu (ví dụ: đơn xin học, giấy báo nhập học, ...), lâu nay người ta dùng chữ ký "***tay***", ghi vào phía dưới của mỗi tài liệu. Như vậy người ký phải ***trực tiếp*** "***ký tay***" vào tài liệu.

Ngày nay các tài liệu được số hóa, người ta cũng có nhu cầu chứng thực nguồn gốc hay hiệu lực của các tài liệu này. Rõ ràng không thể "ký tay" vào tài liệu, vì chúng không được in ấn trên giấy. Tài liệu "số" (hay tài liệu "điện tử") là một xâu các bit (0 hay 1), xâu bít có thể rất dài (nếu in trên giấy có thể hàng nghìn trang) "Chữ ký" để chứng thực một xâu bít tài liệu cũng không thể là một xâu bit nhỏ đặt phía dưới xâu bit tài liệu. Một "chữ ký" như vậy chắc chắn sẽ bị kẻ gian sao chép để đặt dưới một tài liệu khác bất hợp pháp.

Những năm 80 của thế kỷ 20, các nhà khoa học đã phát minh ra "***chữ ký số***" để chứng thực một "***tài liệu số***". Đó chính là "**bản mã**" của xâu bít tài liệu.

Người ta tạo ra "***chữ ký số***" (chữ ký điện tử) trên "***tài liệu số***" giống như tạo ra "**bản mã**" của tài liệu với "khóa lập mã".

Như vậy "***ký số***" trên "***tài liệu số***" là "***ký***" trên từng bit tài liệu. Kẻ gian khó thể giả mạo "chữ ký số" nếu nó không biết "khóa lập mã".

Để kiểm tra một "***chữ ký số***" thuộc về một "***tài liệu số***", người ta giải mã "***chữ ký số***" bằng "khóa giải mã"., và so sánh với tài liệu gốc.

Ngoài ý nghĩa để chứng thực nguồn gốc hay hiệu lực của các tài liệu số hóa, Mặt mạnh của "***chữ ký số***" hơn "chữ ký tay" là ở chỗ người ta có thể "***ký***" vào tài liệu từ rất xa trên mạng công khai. Hơn thế nữa, có thể "***ký***" bằng các thiết bị cầm tay (VD điện

thoại di động) tại khắp mọi nơi (Ubikytous) và di động (Mobile), miễn là kết nối được vào mạng. Đỡ tốn bao thời gian, sức lực, chi phí, …

"Ký số" thực hiện trên từng bit tài liệu, nên độ dài của "***chữ ký số***" ít nhất cũng bằng độ dài của tài liệu. Do đó thay vì ký trên tài liệu dài, người ta thường dùng "***hàm băm***" để tạo "***đại diện***" cho tài liệu, sau đó mớI "Ký số" lên "***đại diện***" này.

4.1.1.2. Sơ đồ chữ ký số

Sơ đồ chữ ký là bộ năm (**P**, **A**, **K**, **S**, **V**), trong đó:

P là tập hữu hạn các văn bản có thể.

A là tập hữu hạn các chữ ký có thể.

K là tập hữu hạn các khoá có thể.

S là tập các thuật toán ký.

V là tập các thuật toán kiểm thử.

Với mỗi khóa $\mathbf{k} \in \mathbf{K}$, có thuật toán ký $\mathbf{Sig}_k \in \mathbf{S}$, $\mathbf{Sig}_k : \mathbf{P} \rightarrow \mathbf{A}$, có thuật toán kiểm tra chữ ký $\mathbf{Ver}_k \in \mathbf{V}$, $\mathbf{Ver}_k : \mathbf{P} \times \mathbf{A} \rightarrow$ {đúng, sai}, thoả mãn điều kiện sau với mọi $\mathbf{x} \in \mathbf{P}$, $\mathbf{y} \in \mathbf{A}$:

$$\mathbf{Ver}_k(\mathbf{x}, \mathbf{y}) = \begin{cases} \text{Đúng, nếu } \mathbf{y} = \mathbf{Sig}_k(\mathbf{x}) \\ \text{Sai, nếu } \mathbf{y} \neq \mathbf{Sig}_k(\mathbf{x}) \end{cases}$$

Chú ý

Người ta thường dùng hệ mã hóa khóa công khai để lập "***Sơ đồ chữ ký số"***.

Ở đây khóa bí mật **a** dùng làm khóa "***ký***", khóa công khai **b** dùng làm khóa kiểm tra "***chữ ký***".

Ngược lại với việc mã hóa, dùng khóa công khai **b** để lập mã., dùng khóa bí mật **a** để giải mã.

Điều này là hoàn toàn tự nhiên, vì "***ký***" cần giữ bí mật nên phải dùng khóa bí mật **a** để "***ký***". Còn "***chữ ký***" là công khai cho mọi người biết, nên họ dùng khóa công khai **b** để kiểm tra.

4.1.2. Phân loại "Chữ ký số".

Có nhiều loại chữ ký tùy theo cách phân loại, sau đây xin giới thiệu một số cách.

***Cách 1: Phân loại chữ ký theo* đặc trưng kiểm tra chữ ký.**

1) **Chữ ký khôi phục thông điệp**:

Là loại chữ ký, trong đó người gửi chỉ cần gửi "***chữ ký***", người nhận có thể khôi phục lại được thông điệp, đã được "***ký***" bởi "***chữ ký***" này.

Ví dụ: Chữ ký RSA là chữ ký khôi phục thông điệp, sẽ trình bày trong mục sau.

2) **Chữ ký đi kèm thông điệp**:

Là loại chữ ký, trong đó người gửi chỉ cần gửi "***chữ ký***", phải gửi kèm cả thông điệp đã được "***ký***" bởi "***chữ ký***" này. Ngược lại, người nhận sẽ không có được thông điệp gốc.

Ví dụ: Chữ ký Elgamal là chữ ký đi kèm thông điệp, sẽ trình bày trong mục sau.

***Cách 2: Phân loại chữ ký theo* mức an toàn.**

1) **Chữ ký "không thể phủ nhận"**:

Nhằm tránh việc nhân bản chữ ký để sử dụng nhiều lần, tốt nhất là người gửi tham gia trực tiếp vào việc kiểm thử chữ ký. Điều đó được thực hiện bằng một giao thức kiểm thử, dưới dạng một giao thức mời hỏi và trả lời.

Ví dụ: Chữ ký không phủ định (Chaum - van Antverpen), trình bày trong mục sau.

2) **Chữ ký "một lần"**:

Để bảo đảm an toàn, "Khóa ký" chỉ dùng 1 lần (one- time) trên 1 tài liệu.

Ví dụ: Chữ ký một lần Lamport. Chữ ký Fail - Stop (Van Heyst & Pedersen)

***Cách 3: Phân loại chữ ký theo* ứng dụng đặc trưng.**

Chữ ký "mù" (Blind Signature)

Chữ ký "nhóm" (Group Signature)

Chữ ký "bội" (Multy Signature)

Chữ ký "mù nhóm" (Blind Group Signature)

Chữ ký "mù bội" (Blind Multy Signature)

4.2. Chữ ký RSA

4.2.1. Sơ đồ chữ ký

Sơ đồ (Đề xuất năm 1978)

****Tạo cặp khóa*** (***bí mật, công khai***) (**a**, **b**) :

Chọn bí mật số nguyên tố lớn **p**, **q**, tính **n** = **p** * **q**, công khai **n**, đặt **P** = **C** = $\mathbf{Z_n}$

Tính bí mật $\phi(\mathbf{n}) = (\mathbf{p-1})(\mathbf{q-1})$ Chọn khóa công khai $\mathbf{b} < \phi(\mathbf{n})$, nguyên tố với $\phi\mathbf{(n)}$

Khóa bí mật **a** là phần tử nghịch đảo của **b** theo mod ϕ(n): $\mathbf{a*b} \equiv \mathbf{1} \pmod{\phi(n)}$

Tập cặp khóa (bí mật, công khai) **K** = {(**a**, **b**)/ a, b $\in Z_n$, $\mathbf{a*b} \equiv 1 \pmod{\phi(n)}$}.

* ***Ký số***: Chữ ký trên $\mathbf{x} \in \mathbf{P}$ là $\mathbf{y} = \mathbf{Sig}_k(\mathbf{x}) = \mathbf{x}^{\mathbf{a}} \pmod{\mathbf{n}}$, $\mathbf{y} \in \mathbf{A}$. (R1)

* ***Kiểm tra chữ ký***: $\mathbf{Ver_k}(\mathbf{x}, \mathbf{y})$ = đúng $\Leftrightarrow$ $\mathbf{x} \equiv \mathbf{y}^{\mathbf{b}} \pmod{\mathbf{n}}$ (R2)

Chú ý

- So sánh giữa sơ đồ chữ ký RSA và sơ đồ mã hóa RSA ta thấy có sự tương ứng.
- Việc ký chẳng qua là mã hoá, việc kiểm thử lại chính là việc giải mã:

Việc "ký số" vào **x** tương ứng với việc "mã hoá" tài liệu **x**.

Kiểm thử chữ ký chính là việc giải mã "chữ ký", để kiểm tra xem tài liệu đã giải mã có đúng là tài liệu trước khi ký không. Thuật toán và khóa kiểm thử "chữ ký" là công khai, ai cũng có thể kiểm thử chữ ký được.

Ví dụ Chữ ký trên **x** = **2**

****Tạo cặp khóa*** (***bí mật, công khai***) (**a**, **b**) :

Chọn bí mật số nguyên tố **p**=3, **q**=5, tính **n** = **p** * **q** = 3*5 = 15, công khai **n**.

Đặt **P** = **C** = $\mathbf{Z_n} = \mathbf{Z_n}$. Tính bí mật $\phi(\mathbf{n}) = (\mathbf{p-1})(\mathbf{q-1})$ = 3 * 4 = 8.

Chọn khóa công khai $\mathbf{b} = \mathbf{3} < \phi(\mathbf{n})$, nguyên tố với $\phi\mathbf{(n) = 8}$.

Khóa bí mật **a** = **3**, là phần tử nghịch đảo của **b** theo mod ϕ(n): $\mathbf{a*b} \equiv \mathbf{1} \pmod{\phi(n)}$

* ***Ký số***: Chữ ký trên $\mathbf{x} = \mathbf{2} \in \mathbf{P}$ là

$\mathbf{y} = \mathbf{Sig}_k(\mathbf{x}) = \mathbf{x}^{\mathbf{a}} \pmod{\mathbf{n}} = \mathbf{2}^{\mathbf{3}} \pmod{\mathbf{15}} = \mathbf{8}$, $\mathbf{y} \in \mathbf{A}$.

* ***Kiểm tra chữ ký***: $\mathbf{Ver_k}(\mathbf{x}, \mathbf{y})$ = đúng $\Leftrightarrow \mathbf{x} \equiv \mathbf{y}^{\mathbf{b}} \pmod{\mathbf{n}}$

$\Leftrightarrow \mathbf{2} \equiv \mathbf{8}^{\mathbf{b}} \pmod{\mathbf{15}}$

4.2.2. Độ an toàn của chữ ký RSA

1) Người gửi G gửi tài liệu **x** cùng chữ ký **y** đến người nhận N, có 2 cách xử lý:

a) Ký trước, Mã hóa sau:

G ký trước vào **x** bằng chữ ký **y** = $\mathbf{Sig_G}$ (**x**), sau đó mã hoá **x** và **y** nhận được

z = $\mathbf{e_G}$ (**x**, **y**) G gửi **z** cho N.

Nhận được **z**, N giải mã **z** để được **x**, **y**.

Tiếp theo kiểm tra chữ ký $\mathbf{Ver_N}$ (**x**, **y**) = true ?

b) Mã hóa trước, Ký sau:

G mã hoá trước **x** bằng **u** = $\mathbf{e_G}$ (**x**), sau đó ký vào **u** bằng chữ ký **v** = $\mathbf{Sig_G}$ (**u**)

G gửi (**u**, **v**) cho N.

Nhận được (**u**, **v**), N giải mã **u** được **x**.

Tiếp theo kiểm tra chữ ký $\mathbf{Ver_N}$ (**u**, **v**) = true ?

2) Giả sử H lấy trộm được thông tin trên đường truyền từ G đến N.

+ Trong trường hợp **a**, H lấy được **z**. Trong trường hợp **b**, H lấy được (**u**, **v**)

+ Để tấn công **x**, trong cả hai trường hợp, H đều phải giải mã thông tin lấy được.

+ Để tấn công vào chữ ký, thay bằng chữ ký (giả mạo), thì xảy ra điều gì ?

- Trường hợp **a**, để tấn công chữ ký **y**, H phải giải mã **z**, mới nhận được **y**.

- Trường hợp **b**, để tấn công chữ ký **v**, H đã sẵn có **v**, H chỉ việc thay **v** bằng **v'**.

H thay chữ ký **v** trên **u**, bằng chữ ký của H là **v'** = Sig_H(**u**), gửi (**u**, **v'**) đến N.

Khi nhận được **v'**, N kiểm thử thấy sai, gửi phản hồi lại G.

G có thể chứng minh chữ ký đó là giả mạo.

G gửi chữ ký đúng **v** cho N, nhưng quá trình truyền tin sẽ bị chậm lại.

+ Như vậy trong trường hợp **b**, H có thể giả mạo chữ ký mà không cần giải mã.

Vì thế có lời khuyên: ***Hãy ký trước, sau đó mã hoá cả chữ ký.***

4.3. Chữ ký ELGAMAL

4.3.1. Sơ đồ chữ ký Elgamal

Sơ đồ (Elgamal đề xuất năm 1985)

****Tạo cặp khóa* (*bí mật, công khai*) *(*a, h*)* :

Chọn số nguyên tố **p** sao cho bài toán logarit rời rạc trong $\mathbf{Z_p}$ là "khó" giải. Chọn phần tử nguyên thuỷ **g** $\in$ $\mathbf{Z_p}$* . Đặt **P** = $\mathbf{Z}_{\mathbf{p}}$*, **A** = $\mathbf{Z}_{\mathbf{p}}$* x $\mathbf{Z}_{\mathbf{p-1}}$.

Chọn khóa bí mật là **a** $\in$ $\mathbf{Z_p}$* . Tính khóa công khai **h** $\equiv$ $\mathbf{g}^{\mathbf{a}}$ mod **p**.

Định nghĩa tập khóa: $\mathcal{K}$ = {(**p**, **g**, **a**, **h**): $\mathbf{h} \equiv \mathbf{g}^{\mathbf{a}} \bmod \mathbf{p}$}.

Các giá trị **p**, **g**, **h** được công khai, phải giữ bí mật **a**.

* ***Ký số***: Dùng 2 khóa ký: khóa **a** và khóa ngẫu nhiên bí mật $\mathbf{r} \in \mathbf{Z}_{\mathbf{p-1}}^{*}$.

(Vì $\mathbf{r} \in \mathbf{Z}_{\mathbf{p-1}}^{*}$, nên nguyên tố cùng **p** -1, do đó tồn tại $\mathbf{r}^{-1} \bmod (\mathbf{p}-1)$)

Chữ ký trên $\mathbf{x} \in \mathbf{P}$ là $\mathbf{y} = \mathbf{Sig}_{\mathbf{k}}(\mathbf{x}, \mathbf{r}) = (\gamma, \delta)$, $\mathbf{y} \in \boldsymbol{A}$ (E1)

Trong đó $\gamma \in \mathbf{Z}_{\mathbf{p}}^{*}$, $\delta \in \mathbf{Z}_{\mathbf{p-1}}$:

$$\gamma = \mathbf{g}^{\mathbf{r}} \bmod \mathbf{p} \quad \text{và} \quad \delta = (\mathbf{x} - \mathbf{a} * \gamma) * \mathbf{r}^{-1} \bmod (\mathbf{p}-1)$$

* ***Kiểm tra chữ ký***:

$\mathbf{Ver}_{\mathbf{k}}(\mathbf{x}, \gamma, \delta)$ = đúng $\Leftrightarrow$ $\mathbf{h}^{\gamma} * \gamma^{\delta} \equiv \mathbf{g}^{\mathbf{x}} \bmod \mathbf{p}$. (E2)

Chú ý: Nếu chữ ký được tính đúng, kiểm thử sẽ thành công vì

$\mathbf{h}^{\gamma} * \gamma^{\delta} \equiv \mathbf{g}^{\mathbf{a}\gamma} * \mathbf{g}^{\mathbf{r} * \delta} \bmod p \equiv \mathbf{g}^{(\mathbf{a}\gamma + \mathbf{r} * \delta)} \bmod p \equiv \mathbf{g}^{\mathbf{x}} \bmod \mathbf{p}$.

Do $\delta = (\mathbf{x} - \mathbf{a} * \gamma) * \mathbf{r}^{-1} \bmod (p-1)$ nên $(\mathbf{a} * \gamma + \mathbf{r} * \delta) \equiv \mathbf{x} \bmod (p-1)$

<u>Ví dụ</u> Chữ ký Elgamal trên dữ liệu **x** = **112**.

****Tạo cặp khóa* (*bí mật, công khai*) (a, h)** :

Chọn số nguyên tố **p** = **463**. Đặt $\mathbf{P} = \mathbf{Z}_{\mathbf{p}}^{*}$, $\mathbf{A} = \mathbf{Z}_{\mathbf{p}}^{*} \times \mathbf{Z}_{\mathbf{p-1}}$.

Chọn phần tử nguyên thuỷ $\mathbf{g} = \mathbf{2} \in \mathbf{Z}_{\mathbf{p}}^{*}$.

Chọn khóa bí mật là $\mathbf{a} = 211 \in \mathbf{Z}_{\mathbf{p}}^{*}$.

Tính khóa công khai $\mathbf{h} \equiv \mathbf{g}^{\mathbf{a}} \bmod \mathbf{p} = \mathbf{2}^{221} \bmod 463 = \mathbf{249}$.

Định nghĩa tập khóa: $\mathcal{K}$ = {(**p**, **g**, **a**, **h**): $\mathbf{h} \equiv \mathbf{g}^{\mathbf{a}} \bmod \mathbf{p}$}.

Các giá trị **p**, **g**, **h** được công khai, phải giữ bí mật **a**.

* ***Ký số***: Chọn ngẫu nhiên bí mật $\mathbf{r} = 235 \in \mathbf{Z}_{\mathbf{p-1}}^{*}$. Khóa ký là (**a**, **r**)

Vì $\mathbf{r} \in \mathbf{Z}_{\mathbf{p-1}}^{*}$, nên nguyên tố cùng **p** -1, do đó tồn tại $\mathbf{r}^{-1} \bmod (\mathbf{p}-1)$ Cụ thể:

UCLN(**r**, **p**-1) = UCLN(235, 462) = 1, nên $\mathbf{r}^{-1} \bmod (p-1) = 235^{-1} \bmod 462 = 289$.

Chữ ký trên dữ liệu **x** = **112** là (γ, δ) = (**16**, **108**), trong đó:

$\gamma = \mathbf{g}^{\mathbf{r}} \bmod \mathbf{p} = \mathbf{2}^{235} \bmod 463 = \mathbf{16}$

$\delta = (\mathbf{x} - \mathbf{a} * \gamma) * \mathbf{r}^{-1} \bmod (\mathbf{p}-1) = (\mathbf{112} - 211 * 16) * 289 \bmod 462 = \mathbf{108}$

* ***Kiểm tra chữ ký***: $\mathbf{Ver}_{\mathbf{k}}(\mathbf{x}, \gamma, \delta)$ = đúng $\Leftrightarrow$ $\mathbf{h}^{\gamma} * \gamma^{\delta} \equiv \mathbf{g}^{\mathbf{x}} \bmod \mathbf{p}$.

$\mathbf{h}^{\gamma} * \gamma^{\delta}$ = $249^{16} * 16^{108} \bmod 463 = \mathbf{132}$

$\mathbf{g}^{x} \bmod \mathbf{p} = \mathbf{2}^{\mathbf{112}} \bmod 463 = \mathbf{132}$.

Hai giá trị đó bằng nhau, như vậy chữ ký là đúng.

4.3.2. Độ an toàn của chữ ký Elgamal

4.3.2.1. Vấn đề giả mạo chữ ký Elgamal

***1) Trường hợp 1*: Giả mạo chữ ký không cùng với tài liệu được ký**.

+ H cố gắng giả mạo chữ ký trên **x**, mà không biết khóa bí mật **a**.

Như vậy, H phải tính được γ và δ.

* Nếu chọn trước γ, H phải tính δ qua đẳng thức $\mathbf{h}^{\gamma} * \gamma^{\delta} \equiv \mathbf{g}^{x} \bmod \mathbf{p}$ (E2)

Tức là $\gamma^{\delta} \equiv \mathbf{g}^{x}\ \mathbf{h}^{-\gamma} \bmod p$ hay $\delta \equiv \log_{\gamma} \mathbf{g}^{x}\ \mathbf{h}^{-\gamma} \bmod p$.

* Nếu chọn trước δ, H phải tính γ qua phương trình: $\mathbf{h}^{\gamma} * \gamma^{\delta} \equiv \mathbf{g}^{x} \bmod \mathbf{p}$.

Hiện nay chưa có cách hữu hiệu 2 trường hợp trên, nhưng phỏng đoán là khó hơn bài toán logarit rời rạc.

Có thể có cách tính γ, δ đồng thời với (γ, δ) là chữ ký ? Chưa có trả lời rõ !

* Nếu chọn trước γ, δ, sau đó tính **x,** H phải đối đầu với bài toán logarit rời rạc.

Ta có $\mathbf{h}^{\gamma} * \gamma^{\delta} \equiv \mathbf{g}^{x} \bmod \mathbf{p}$ (E2)

Như vậy $\mathbf{x} \equiv \log_{\mathbf{g}} \mathbf{g}^{x} \equiv \log_{\mathbf{g}} \mathbf{h}^{\gamma} * \gamma^{\delta}$

***2) Trường hợp 2*: Giả mạo chữ ký cùng với tài liệu được ký**.

H có thể ký trên tài liệu ngẫu nhiên bằng cách chọn trước đồng thời **x**, γ, δ.

Cách 1

* Chọn **x**, γ, δ thoả mãn điều kiện kiểm thử như sau:

Chọn các số nguyên i, j sao cho $0 \le i, j \le p-2$, $(j, p-1) = 1$ và tính:

$\gamma = g^{i} h^{j} \bmod p$, $\delta = -\gamma\, j^{-1} \bmod (p-1)$, $\mathbf{x} = -\gamma\ i\, j^{-1} \bmod (p-1)$

Trong đó j^{-1} được tính theo mod (p -1) (nghĩa là j nguyên tố với p-1)

* Chứng minh (γ, δ) là chữ ký trên **x,** bằng cách kiểm tra điều kiện kiểm thử:

$\mathbf{h}^{\gamma}\gamma^{\delta} \equiv \mathbf{h}^{\gamma}(\mathbf{g}^{i}\ \mathbf{h}^{j})^{-\gamma \cdot j^{-1}} \bmod p \equiv \mathbf{h}^{\gamma}\ \mathbf{g}^{-i \cdot \gamma \cdot j^{-1}}\ \mathbf{h}^{-\gamma} \bmod p \equiv \mathbf{g}^{x} \bmod p$

Ví dụ

* Chọn các tham số của sơ đồ chữ ký Elgamal:

Số nguyên tố p = 463, phần tử sinh **g** = 2, Khóa bí mật a = 135.

Khóa công khai $\mathbf{h} = \mathbf{g}^{\mathbf{a}} \bmod p = 2^{135} \bmod 463 = 272$.

* Chọn $\mathbf{x}, \gamma, \delta$ thoả mãn điều kiện kiểm thử như sau:

Chọn i = 89, j = 125, $0 \le i, j \le p-2$, (j, p-1) = 1. Tính $j^{-1} \bmod (p-1) = 377$.

$\gamma = \mathbf{g}^{i} * \mathbf{h}^{j} \bmod p = 2^{89} * 272^{125} \bmod 463 = 218$

$\delta = -\gamma * j^{-1} \bmod (p-1) = -218 * 377 \bmod 462 = 50$

$\mathbf{x} = -\gamma * i * j^{-1} \bmod (p-1) = -218 * 89 * 377 \bmod 462 =$ **292**

* $(\gamma, \delta) = (218, 50)$ là chữ ký trên **x = 292,** vì thỏa mãn điều kiện kiểm thử:

$\mathbf{h}^{\gamma} * \gamma^{\delta} = 272^{218} * 218^{50} \equiv 322 \pmod{463}$

$\mathbf{g}^{x} = 2^{292} \equiv 322 \pmod{467}$

Cách 2

* Nếu (γ, δ) là chữ ký trên tài liệu **x** có từ trước, thì có thể giả mạo chữ ký trên tài liệu **x'** khác.

+ Chọn số nguyên k, i, j thỏa mãn $0 \le k, i, j \le p-2$, $(k\gamma - j\delta, p-1) = 1$ và tính:

$\lambda = \gamma^{k} g^{i} h^{j} \bmod p$, $\mu = \delta\lambda (\mathbf{k}\gamma - \mathbf{j}\delta)^{-1} \bmod (p-1)$,

$\mathbf{x'} = \lambda (\mathbf{k}\,\mathbf{x} + \mathbf{i}\,\delta) (\mathbf{k}\gamma - \mathbf{j}\delta)^{-1} \bmod (p-1)$

* (λ, μ) là chữ ký trên **x',** vì thỏa mãn điều kiện kiểm thử:

$\mathbf{h}^{\lambda} \lambda^{\mu} \equiv \mathbf{g}^{x'} \bmod p$.

Chú ý

Cả hai cách giả mạo nói trên đều cho chữ ký đúng trên tài liệu tương ứng, nhưng đó không phải là tài liệu được chọn theo ý của người giả mạo. Tài liệu đó đều được tính sau khi tính chữ ký, vì vậy giả mạo loại này trong thực tế cũng không có ý nghĩa nhiều.

4.3.2.2. Vấn đề Phá khóa theo sơ đồ Elgamal

Khoá bí mật **a** có thể bị phát hiện, nếu khóa ngẫu nhiên **r** bị lộ, hoặc dùng **r** cho hai lần ký khác nhau.

1) Trường hợp 1: Số ngẫu nhiên r bị lộ:

Nếu **r** bị lộ, thám mã sẽ tính được khoá mật $\mathbf{a} = (\mathbf{x} - \mathbf{r}\delta)\ \gamma^{-1} \bmod (p-1)$

2) Trường hợp 2: Dùng r cho hai lần ký khác nhau:

Giả sử dùng **r** cho 2 lần ký trên $\mathbf{x}_1$ và $\mathbf{x}_1$.

(γ, δ_1) là chữ ký trên $\mathbf{x}_1$, (γ, δ_2) là chữ ký trên $\mathbf{x}_2$,

Khi đó thám mã có thể tính **a** như sau:

$\beta^{\gamma} * \gamma^{\delta_1} \equiv \alpha^{x_1} (\text{mod } p)$. $\beta^{\gamma} * \gamma^{\delta_2} \equiv \alpha^{x_2} (\text{mod } p)$

Do đó ta có $\alpha^{x_1 - x_2} \equiv \gamma^{\delta_1 - \delta_2} (\text{mod } p)$

Đặt $\gamma = \alpha^{r}$, ta có $\alpha^{x_1 - x_2} \equiv \gamma^{k*(\delta_1 - \delta_2)} (\text{mod } p)$

tương đương với $x_1 - x_2 \equiv r\ (\delta_1 - \delta_2) \bmod (p-1)$ (1)

Đặt $d = (\delta_1 - \delta_2, p - 1)$ Khi đó $d \mid (p-1)$, $d \mid (\delta_1 - \delta_2) \Rightarrow d \mid (x_1 - x_2)$

$$x' = \frac{x_1 - x_2}{d}$$
$$\delta' = \frac{\delta_1 - \delta_2}{d}$$
$$p' = \frac{p-1}{d}$$

Khi đó đồng dư thức (1) trở thành: $x' \equiv \mathbf{r} * \delta' \pmod{p'}$

Vì $(\delta', p') = 1$ nên tính $\varepsilon = (\delta')^{-1} \bmod p'$ và tính $\mathbf{r} = x' * \varepsilon \bmod p'$

$\Rightarrow \mathbf{r} = x' * \varepsilon + i * p' \bmod (p-1)$, với i là giá trị nào đó, $0 \le i \le d-1$.

Thử với giá trị đó, ta tìm được **r** (điều kiện thử để xác định **r** là $\gamma = \alpha^{r} \bmod p$)

Tiếp theo sẽ tính được **a** như trường hợp 1)

4.4. Chữ ký DSS

4.4.1. Sơ đồ chữ ký DSS

4.4.1.1. Giới thiệu Chuẩn chữ ký số DSS

Chuẩn chữ ký số (DSS: Digital Signature Standard) được đề xuất năm 1991, là cải biên của sơ đồ chữ ký ElGamal, và được chấp nhận là chuẩn vào năm 1994 để dùng trong một số lĩnh vực giao dịch ở USA.

Thông thường tài liệu số được mã hoá và giải mã 1 lần. Nhưng chữ ký lại liên quan đến ***pháp luật***, ***chữ ký,*** có thể phải kiểm thử sau nhiều năm đã ký. Do đó ***chữ ký*** phải được bảo vệ cẩn thận.

Số nguyên tố **p** phải đủ lớn (chẳng hạn dài cỡ 512 bit) để bảo đảm an toàn, nhiều người đề nghị nó phải dài 1024 bit. Tuy nhiên, độ dài chữ ký theo sơ đồ Elgamal là gấp đôi số bit của p, do đó nếu p dài 512 bit thì độ dài chữ ký là 1024 bit.

Trong ứng dụng dùng thẻ thông minh (Smart card) lại mong muốn có chữ ký ngắn, nên giải pháp sửa đổi là một mặt dùng p với độ dài từ 512 bit đến 1024 bit (bội của 64), mặt khác trong chữ ký (γ, δ), các số γ, δ có độ dài biểu diễn ngắn, ví dụ 160 bit. Khi đó chữ ký là 320 bit.

Điều này được thực hiện bằng cách dùng nhóm con cyclic $\mathbf{Z_q}^*$ của $\mathbf{Z_p}^*$ thay cho $\mathbf{Z_p}^*$, do đó mọi tính toán được thực hiện trong $\mathbf{Z_p}^*$, nhưng thành phần chữ ký lại thuộc $\mathbf{Z_q}^*$.

+ Trong sơ đồ ký Elgamal, công thức tính δ được sửa đổi thành

$\delta = (x + \mathbf{a} * \gamma)\ \mathbf{r}^{-1} \bmod q.$

+ Điều kiện kiểm thử $\mathbf{h}^{\gamma}\ \gamma^{\delta} \equiv \mathbf{g}^{x} \bmod p$ được sửa đổi thành

$$\alpha^{x*\delta^{-1}} * \beta^{\gamma*\delta^{-1}} \equiv \gamma (\bmod p).$$

Chú ý nếu UCLN(x + **g** * γ, p-1) = 1 thì $\delta^{-1} \bmod p$ tồn tại.

4.4.1.2. Sơ đồ Chuẩn chữ ký số DSS

Sơ đồ

****Tạo cặp khóa* (*bí mật, công khai*) *(*a, h*)* :

+ Chọn số nguyên tố **p** sao cho bài toán logarit rời rạc trong $\mathbf{Z_p}$ là "khó" giải.

Chọn **q** là ước nguyên tố của p-1. Tức là p-1 = **t** * **q** hay p = **t** * q + 1.

(Số nguyên tố p cỡ 512 bit, q cỡ 160 bit)

+ Chọn $\mathbf{g} \in \mathbf{Z_p}^*$ là căn bậc **q** của **1** mod **p**, (**g** là phần tử sinh của $\mathbf{Z_p}^*$)

Tính $\alpha = \mathbf{g}^{t}$, chọn khóa bí mật $\mathbf{a} \in \mathbf{Z_p}^*$, tính khóa công khai $\mathbf{h} \equiv \alpha^{\mathbf{a}} \bmod \mathbf{p}$.

+ Đặt $\boldsymbol{P} = \mathbf{Z}_{\mathbf{q}}^*$, $\mathbf{A} = \mathbf{Z}_{\mathbf{q}}^* \times \mathbf{Z}_{\mathbf{q}}^*$, $\boldsymbol{K} = \{(\mathbf{p}, \mathbf{q}, \alpha, \mathbf{a}, \mathbf{h}) / \mathbf{a} \in \mathbf{Z_p}^*, \mathbf{h} \equiv \alpha^{\mathbf{a}} \bmod \mathbf{p}\}$.

+ Với mỗi khóa (**p**, **q**, α, **a**, **h**), k' = **a** bí mật, k" = (**p**, **q**, α, **h**) công khai.

*** *Ký số***: Dùng 2 khóa ký: khóa **a** và khóa ngẫu nhiên bí mật $\mathbf{r} \in \mathbf{Z}_{\mathbf{q}}^*$.

Chữ ký trên $\mathbf{x} \in \mathbf{Z_p}^*$ là $\mathbf{Sig}_{k'}(\mathbf{x}, \mathbf{r}) = (\gamma, \delta)$, trong đó

$\gamma = (\alpha^{\mathbf{r}} \bmod p) \bmod \mathbf{q}$, $\delta = ((\mathbf{x} + \mathbf{a} * \gamma) * \mathbf{r}^{-1} \bmod \mathbf{q}$.

(Chú ý $\mathbf{r} \in \mathbf{Z}_{\mathbf{q}}^*$, để bảo đảm tồn tại $\mathbf{r}^{-1} \bmod \mathbf{q}$)

*** *Kiểm tra chữ ký***: Với $\mathbf{e_1} = \mathbf{x} * \delta^{-1} \bmod \mathbf{q}$, $\mathbf{e_2} = \gamma * \delta^{-1} \bmod \mathbf{q}$.

$\mathbf{Ver}_{k''}(\mathbf{x}, \gamma, \delta)$ = đúng $\Leftrightarrow$ $(\alpha^{\mathbf{e1}} * \mathbf{h}^{\mathbf{e2}} \bmod \mathbf{p}) \bmod \mathbf{q} = \gamma$

Ví dụ

Tạo cặp khóa* (*bí mật, công khai***) ***(*****a**, **h*****)*** :

Chọn **p** = 7649, **q** = 239 là ước nguyên tố của p-1, **t** = 32.

Tức là p -1 = **t** * **q** hay **p** = **t** * q + 1 = 32*q + 1 = 32*239 + 1 = 7649.

Chọn **g** =3 $\in$ $\mathbf{Z}_{7649}$ là phần tử sinh. $\alpha = \mathbf{g}^{\mathbf{t}} \bmod p = 3^{32} \bmod 7649 = 7098$.

Chọn khóa mật **a** = 85, khóa công khai $\mathbf{h} = \alpha^{\mathbf{a}} \bmod p = 7098^{85} \bmod 7649 = 5387$.

**** Ký số***: Dùng 2 khóa ký: **a** và khóa ngẫu nhiên $\mathbf{r} = 58 \in \mathbf{Z}_q^*$, $\mathbf{r}^{-1} \bmod q = 136$.

+ Chữ ký trên **x** = 1246 là $\mathbf{Sig}_{k'}(\mathbf{x}, \mathbf{r}) = (\gamma, \delta) = (115, 87)$, trong đó

$\gamma = (\alpha^{\mathbf{r}} \bmod p) \bmod q = (7098^{58} \bmod 7649) \bmod 239 = 593 \bmod 239 = 115$.

$\delta = (\mathbf{x} + \mathbf{a} * \gamma) * \mathbf{r}^{-1} \bmod q = (1246 + 85 * 115) * 136 \bmod 239 = 87$.

**** Kiểm tra chữ ký***: $(\gamma, \delta) = (115, 87)$ là chữ ký đúng trên **x** = 1246.

$e_1 = x * \delta^{-1} \bmod q = 1246 * 11 \bmod q = 83$, $e_2 = \gamma * \delta^{-1} \bmod q = 115*11 \bmod q = 70$.

Điều kiện kiểm thử đúng ? $(\alpha^{e1} * \mathbf{h}^{e2} \bmod \mathbf{p}) \bmod \mathbf{q} = \gamma$, với $\delta^{-1} = 11$.

$(7098^{83} * 5387^{70} \bmod 7649) \bmod 239 = 593 \bmod 239 = 115 = \gamma$.

Chú ý

1) Liên quan tới các tính toán cụ thể trong sơ đồ:

+ Chú ý rằng phải có $\delta \neq 0 \pmod q$ để bảo đảm có $\delta^{-1} \bmod q$ trong điều kiện kiểm thử (tương đương UCLN(δ, p-1) = 1) Vì vậy nếu chọn **r** mà không được điều kiện trên, thì phải chọn **r** khác để có $\delta \neq 0 \pmod q$

Tuy nhiên khả năng $\delta \equiv 0 \bmod q$ là 2^{-160} , điều đó hầu như không xảy ra.

+ Một chú ý là là thay vì tính p trước rồi mới tính q, ta sẽ tính q trước rồi tìm p.

2) Liên quan chung tới DSS (1991):

+ Độ dài cố định của p là 512 bit. Nhiều người muốn p có thể thay đổi lớn hơn.

Vì thế NIST sửa đổi là p có độ dài thay đổi, là bội của 64: từ 512 đến 1024 bit.

+ Nếu dùng chữ ký RSA với thành phần kiểm thử chữ ký là nhỏ, thì việc kiểm thử nhanh hơn việc ký. Đối với DSS, ngược lại, việc ký nhanh hơn kiểm thử.

Điều này dẫn đến vấn đề:

Một tài liệu chỉ được ký một lần, nhưng nó lại được kiểm thử nhiều lần, nên người ta muốn thuật toán kiểm thử nhanh hơn.

Máy tính ký và kiểm thử như thế nào ? Nhiều ứng dụng dùng thẻ thông minh với khả năng có hạn, kết nối với 1 máy tính mạnh hơn, vì vậy nên xây dựng sơ đồ chữ ký ít liên quan đến thẻ.

Nhưng tình huống đặt ra là một thẻ thông minh có thể sinh ra chữ ký và cũng có thể kiểm thử chữ ký, do vậy rất khó kết luận ?

NIST trả lời rằng thời gian kiểm thử và sinh chữ ký, cái nào nhanh hơn không quan trọng, miễn là đủ nhanh.

4.5. Chữ ký không thể phủ định

4.5.1. Sơ đồ chữ ký

4.5.1.1. Giới thiệu chữ ký không thể phủ định

Trong phần trước ta đã trình bày một số sơ đồ chữ ký điện tử. Trong các sơ đồ đó, việc kiểm thử tính đúng đắn của chữ ký là do người nhận thực hiện. Nhằm tránh việc nhân bản chữ ký để sử dụng nhiều lần, tốt nhất là để người gửi tham gia trực tiếp vào việc kiểm thử chữ ký. Điều đó được thực hiện bằng một giao thức kiểm thử, dưới dạng một giao thức mời hỏi và trả lời.

Giả sử tài liệu cùng chữ ký từ G gửi đến N. Khi N yêu cầu G cùng kiểm thử chữ ký, thì một vấn đề nảy sinh là làm sao để ngăn cản G chối bỏ một chữ ký mà anh ta đã ký, G có thể tuyên bố rằng chữ ký đó là giả mạo ?

Để giải quyết tình huống trên, cần có thêm giao thức chối bỏ, bằng giao thức này, G có thể chứng minh một chữ ký là giả mạo. Nếu G từ chối tham gia vào giao thức đó, thì có thể xem rằng G không chứng minh được chữ ký đó là giả mạo.

Như vậy sơ đồ chữ ký không phủ định được gồm 3 phần: một thuật toán ký, một giao thức kiểm thử, và một giao thức chối bỏ.

4.5.1.2. Sơ đồ chữ ký không thể phủ định (Chaum - van Antverpen)

**** Chuẩn bị các tham số***:

Chọn số nguyên tố **p** sao cho bài toán log rời rạc trong $\mathbf{Z_p}$ là khó.

p = 2*q+1, q cũng là số nguyên tố.

Gọi **P** là nhóm nhân con của $\mathbf{Z_p}$* theo q (**P** gồm các thặng dư bậc hai theo mod p)

Chọn phần tử sinh **g** của nhóm **P** cấp q.

Đặt **P** = **A** = **P**, **K** = {(**p**, **g**, **a**, **h**): **a** ∈ $\mathbf{Z}_q^*$, **h** ≡ $\mathbf{g}^{\mathbf{a}}$ mod **p** }

1) Thuật toán ký: Dùng khoá bí mật k' = **a** để ký lên **x**:

Chữ ký là **y** = $\mathbf{Sig}_{k'}(\mathbf{x})$ = $\mathbf{x}^{\mathbf{a}}$ mod **p**.

2) Giao thức kiểm thử: Dùng khoá công khai k" = (**p**, **g**, **h**)

Với **x**, **y** ∈ **P**, người nhận N cùng người gửi G thực hiện giao thức kiểm thử:

1/ N chọn ngẫu nhiên $\mathbf{e_1}$, $\mathbf{e_2} \in \mathbf{Z_q}^*$

2/ N tính **c** = $\mathbf{y}^{\mathbf{e1}}$ $\mathbf{h}^{\mathbf{e2}}$ mod p, và gửi cho G.

3/ G tính $d = c^{a^{-1} \bmod q} \bmod p$ và gửi cho N.

4/ N chấp nhận **y** là chữ ký đúng, nếu **d** ≡ $\mathbf{x}^{\mathbf{e1}}$ $\mathbf{g}^{\mathbf{e2}}$ mod p

3) Giao thức chối bỏ:

1/ N chọn ngẫu nhiên $\mathbf{e_1}$, $\mathbf{e_2} \in \mathbf{Z_q}^*$

2/ N tính **c** = $\mathbf{y}^{\mathbf{e1}}$ $\mathbf{h}^{\mathbf{e2}}$ mod p, và gửi cho G.

3/ G tính $d = c^{a^{-1} \bmod q} \bmod p$ và gửi cho N.

4/ N thử điều kiện **d** ≠ $\mathbf{x}^{\mathbf{e1}}$ $\mathbf{g}^{\mathbf{e2}}$ (mod p)

5/ N chọn ngẫu nhiên $\mathbf{f_1}, \mathbf{f_2} \in \mathbf{Z_q}^*$.

6/ N tính $C = y^{f_1} * \beta^{f_2} \bmod p$ và gửi cho G.

7/ G tính $D = C^{a^{-1} \bmod q} \bmod p$ và gửi cho N.

8/ N thử điều kiện **D** ≠ $\mathbf{x}^{\mathbf{f1}}$ $\mathbf{g}^{\mathbf{f2}}$ (mod p)

9/ N kết luận **y** là chữ ký ***giả mạo*** nếu:

$(d * \alpha^{-e_2})^{f_1} \equiv (D * \alpha^{-f_2})^{e_1} (\bmod p)$ (thay α bằng ***g***)

Ví dụ Ký trên **x** = 229

**** Chuẩn bị các tham số***:

Chọn số nguyên tố **p** = 467 = 2 * q + 1, q = 233 cũng là số nguyên tố.

Chọn phần tử sinh của nhóm **P** là **g** = 4, (**P** là nhóm nhân con cấp q của $\mathbf{Z_p}^*$)

Đặt **P** = **A** = **P**, **K** = {(**p**, **g**, **a**, **h**): **a** ∈ $\mathbf{Z}_q^*$, **h** ≡ $\mathbf{g}^{\mathbf{a}}$ mod **p** }

Chọn khóa mật **a** = 121. Khóa công khai **h** ≡ $\mathbf{g}^{\mathbf{a}}$ mod **p** = 4^{121} mod 467 = 422.

1) Thuật toán ký: Dùng khoá bí mật k' = **a** để ký lên **x** = 229:

Chữ ký là **y** = **Sig** $_{k'}$ (**x**) = **x** ^{a} mod **p** = 229^{121} mod 467 = **9**.

2) Giao thức kiểm thử: Dùng khoá công khai k" = (**p**, **g**, **h**) = (467, 4, 422)

1/ N chọn ngẫu nhiên $\mathbf{e_1} = 48$, $\mathbf{e_2} = 213 \in \mathbf{Z_q^*}$

2/ N tính **c** = **y** ^{e1} **h** ^{e2} mod p = 116 và gửi cho G.

3/ G tính $d = c^{a^{-1} \bmod q} \bmod p$ = 235 và gửi cho N.

4/ N chấp nhận **y** là chữ ký đúng, nếu **d** ≡ **x** ^{e1} **g** ^{e2} mod p

N thử điều kiện **d** ≡ **x** ^{e1} **g** ^{e2} mod p.

Rõ ràng 235 ≡ $229^{48} * 4^{213}$ (mod 467)

N chấp nhận **y** = **9** đúng là chữ ký của G trên **x** = 229.

3) Giao thức chối bỏ:

Giả sử G gửi tài liệu **x** = 226 với chữ ký **y** = 183. Giao thức chối bỏ thực hiện:

1/ N chọn ngẫu nhiên $\mathbf{e_1} = 47$, $\mathbf{e_2} = 137 \in \mathbf{Z_q^*}$

2/ N tính **c** = **y** ^{e1} **h** ^{e2} mod p = 306, và gửi cho G.

3/ G tính $d = c^{a^{-1} \bmod q} \bmod p$ = 184, và gửi cho N.

4/ N thử điều kiện **d** ≠ **x** ^{e1} **g** ^{e2} (mod p)

Điều kiện trên không đúng vì $184 \neq 226^{47} * 4^{137} \equiv 145 \bmod 467$.

N lại tiếp tục thực hiện bước 5 của giao thức.

5/ N chọn ngẫu nhiên $\mathbf{f_1} = 225$, $\mathbf{f_2} = 19 \in \mathbf{Z_q^*}$.

6/ N tính $C = y^{f_1} * \beta^{f_2} \bmod p$ = 348, và gửi cho G.

7/ G tính $D = C^{a^{-1} \bmod q} \bmod p$ = 426, và gửi cho N.

8/ N thử điều kiện **D** ≠ **x** ^{f1} **g** ^{f2} (mod p)

D = 426 trong khi **x** ^{f1} **g** ^{f2} (mod p) = $226^{225} * 4^{19} \equiv 295 \bmod 467$.

Điều kiện 8 là đúng, nên N thực hiện bước 9:

9/ N kết luận **y** là chữ ký ***giả mạo*** nếu:

$(d * \alpha^{-e_2})^{f_1} \equiv (D * \alpha^{-f_2})^{e_1} (\bmod p)$ (thay α bằng ***g***)

N tính giá trị của 2 vế đồng dư ≡

$(d^* \alpha^{-e2})^{f1} \equiv (184 * 4^{-137})^{225} \equiv 79 \bmod 467$

$(D^* \alpha^{-f2})^{e1} \equiv (426 * 4^{-19})^{47} \equiv 79 \bmod 467$

Hai giá trị đó bằng nhau. Kết luận chữ ký **y** là ***giả mạo***.

4.6. Đại diện tài liệu và hàm băm

4.6.1. Vấn đề Đại diện tài liệu và Hàm băm

4.6.1.1. Một số vấn đề với "chữ ký số"

Vấn đề 1:

"Ký số" thực hiện trên từng bit tài liệu, nên độ dài của "***chữ ký số***" ít nhất cũng bằng độ dài của tài liệu. Một số chữ ký trên bản tin có kích thước gấp đôi bản tin gốc. Ví dụ khi dùng sơ đồ chữ ký DSS để ký vào bản tin có kích thước 160 bit, thì chữ ký số này sẽ có kích thước 320 bit.

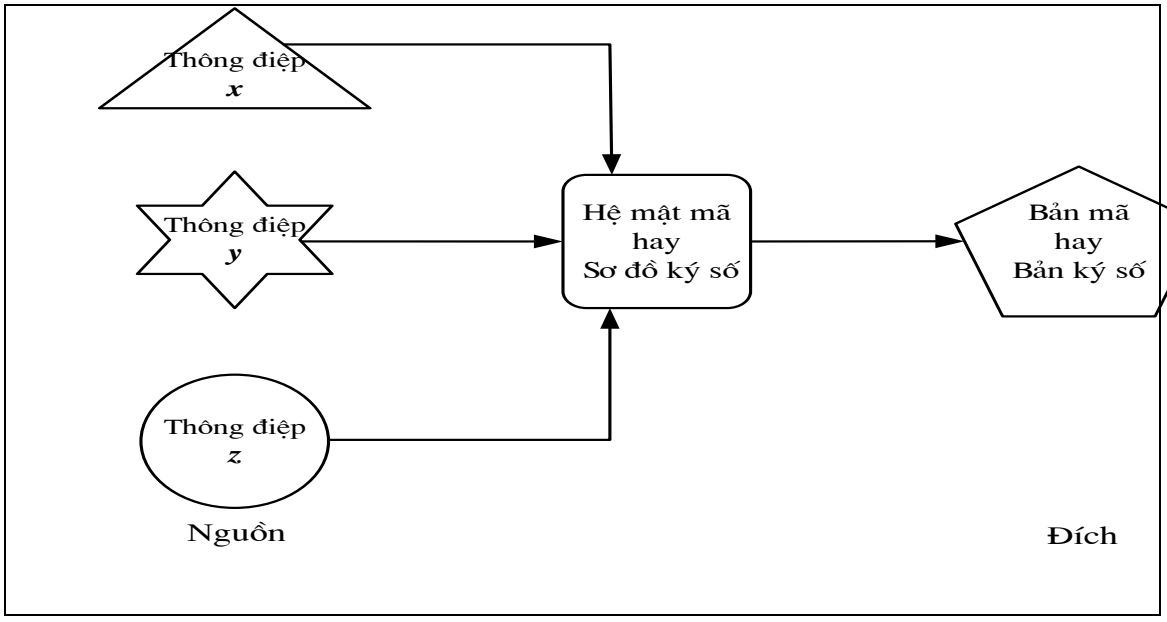
Trong khi đó trên thực tế, ta cần phải ký vào các bản tin có kích thước rất lớn, ví dụ vài chục MegaByte (tương ứng hàng ngàn trang tin trên giấy) Như vậy phải tốn nhiều bộ nhớ để lưu trữ "chữ ký", mặt khác tốn nhiều thời gian để truyền "chữ ký" trên mạng..

Vấn đề 2:

Với một số sơ đồ chữ ký "an toàn", thì tốc độ ký lại chậm vì chúng dùng nhiều phép tính số học phức tạp như số mũ modulo.

Vấn đề 3:

Thực tế có thể xảy ra trường hợp: Với nhiều bản tin đầu vào khác nhau, sử dụng hệ mã hóa hay sơ đồ ký số giống nhau (có thể khác nhau), nhưng lại cho ra bản mã hay chữ ký giống nhau (đó là ánh xạ nhiều – một), như hình dưới. Điều này sẽ dẫn đến phức tạp cho việc xác thực thông tin.



4.6.1.2. Giải quyết các vấn đề trên như thế nào ?

Cách 1:

Một cách đơn giản để giải quyết các vấn đề trên với thông điệp có kích thước lớn là "***chặt***" bản tin thành nhiều đoạn nhỏ (VD 160 bit), sau đó ký lên các đoạn đó độc lập nhau. Nhưng biện pháp này gặp các vấn đề trên.

Hơn thế nữa còn gặp vấn đề nghiêm trọng hơn. Đó là kết quả sau khi ký, nội dung của thông điệp có thể bị xáo trộn các đoạn với nhau, hoặc một số đoạn trong chúng có thể bị mất mát. Ta cần phải bảo vệ tính toàn vẹn của bản tin gốc.

Cách 2:

Thay vì phải ký trên tài liệu dài, người ta thường dùng "***hàm băm***" để tạo "***đại diện***" cho tài liệu, sau đó mới "Ký số" lên "***đại diện***" này.

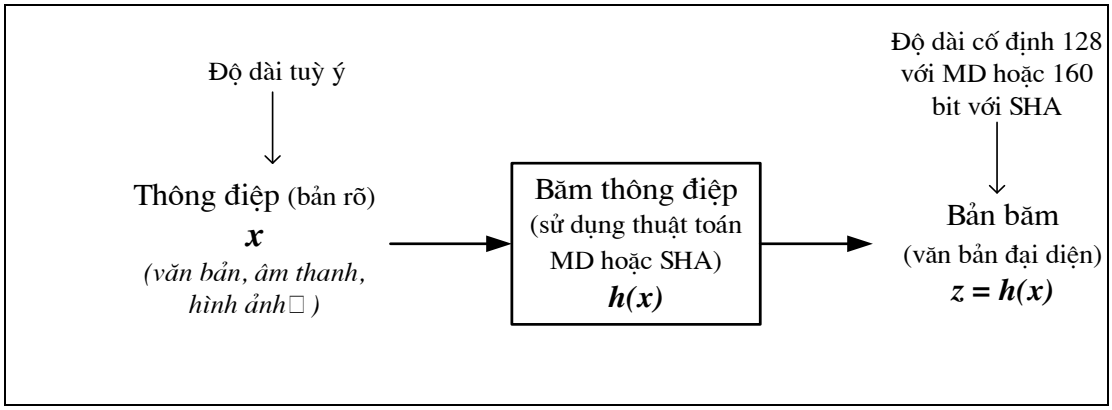
Các tài liệu (bản tin) có thể dưới dạng văn bản, hình ảnh, âm thanh, …và kích thước của chúng tùy ý (vài KB đến vài chục MB), qua các thuật toán băm: như MD4, MD5, SHA, các "***đại diện***" tương ứng của chúng có kích thước ***cố định***, ví dụ 128 bit với dòng MD, 160 bit với dòng SHA.

"***Đại diện***" của tài liệu chính là giá trị của "***hàm băm***" trên tài liệu, nó còn được gọi là "tóm lược" hay "bản thu gọn" của tài liệu.

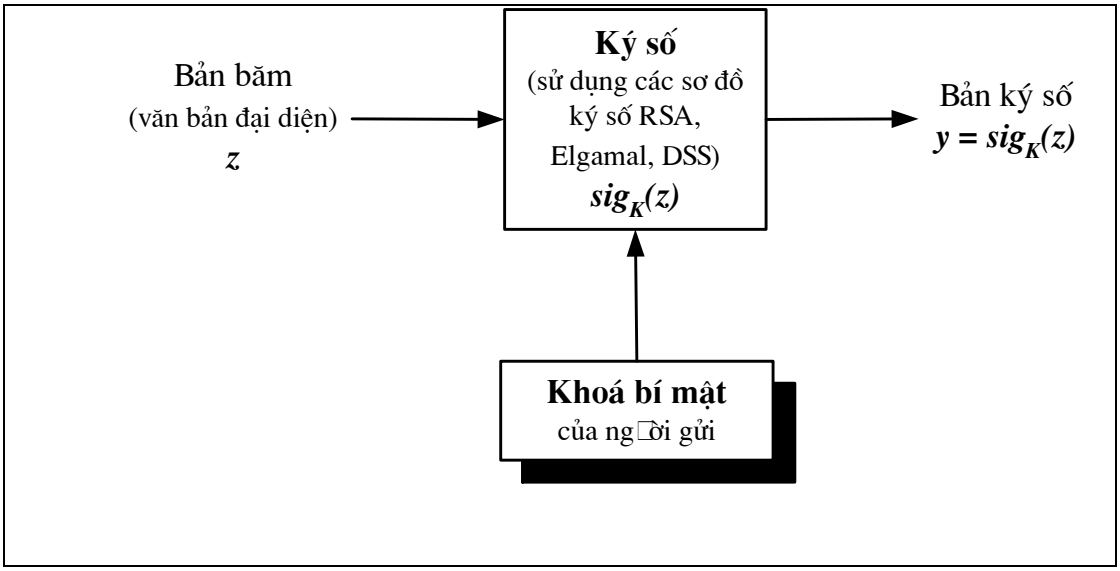
Với mỗi tài liệu (đầu vào), qua "***hàm băm***" chỉ có thể tính ra được một "***đại diện***"- giá trị băm tương ứng - duy nhất. "***Đại diện***" của tài liệu được xem là "***đặc thù***" của tài liệu (thông điệp), giống như dấu vân tay của mỗi người.

Trên thực tế, hai tài liệu khác nhau có hai "đại diện" khác nhau. Như vậy khi đã có "***đại diện***" duy nhất cho một tài liệu, thì việc "ký" vào tài liệu, được thay bằng "ký" vào "***đại diện***" của nó là hoàn toàn hợp lý. Đó là chưa kể việc tiết kiệm bao nhiêu thời gian cho việc "ký số", bộ nhớ lưu giữ "chữ ký", thời gian truyền "chữ ký" trên mạng, …

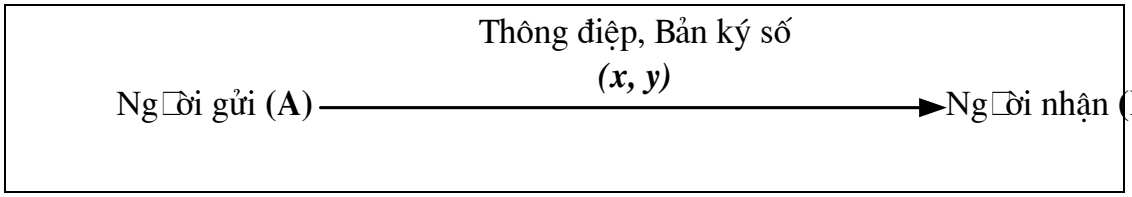
Cơ chế gửi tài liệu cùng "chữ ký" trên nó sử dụng hàm băm được mô tả theo các hình sau.



Băm thông điệp.



Ký trên bản băm thông điệp.



Truyền thông điệp và chữ ký.

4.6.2. Tổng quan về Hàm băm

4.6.2.1. Hàm băm (Hàm tạo đại diện tài liệu)

1) Khái niệm Hàm băm

Hàm băm là thuật toán không dùng khóa để ***mã hóa*** (ở đây dùng thuật ngữ "băm" thay cho "mã hóa"), nó có nhiệm vụ "lọc" (băm) tài liệu (bản tin) và cho kết quả là một giá trị "băm" có kích thước cố định, còn gọi là "***đại diện tài liệu***" hay "đại diện bản tin", "đại diện thông điệp".

Hàm băm là ***hàm một chiều***, theo nghĩa giá trị của hàm băm là ***duy nhất***, và từ giá trị băm này, "**khó thể**" **suy ngược** lại được nội dung hay độ dài ban đầu của tài liệu gốc.

2) Đặc tính của Hàm băm

Hàm băm **h** là hàm một chiều (One-way Hash) với các đặc tính sau:

1) Với tài liệu đầu vào (bản tin gốc) **x**, chỉ thu được giá trị băm duy nhất **z** = **h**(**x**)

2) Nếu dữ liệu trong bản tin **x** bị thay đổi hay bị xóa để thành bản tin **x**', thì giá trị băm **h**(**x**') ≠ **h**(**x**)

Cho dù chỉ là một sự thay đổi nhỏ, ví dụ chỉ thay đổi 1 bit dữ liệu của bản tin gốc **x**, thì giá trị băm **h**(**x**) của nó cũng vẫn thay đổi. Điều này có nghĩa là: hai thông điệp khác nhau, thì giá trị băm của chúng cũng khác nhau.

3) Nội dung của bản tin gốc "khó" thể suy ra từ giá trị hàm băm của nó. Nghĩa là: với thông điệp **x** thì "dễ" tính được **z** = **h**(**x**), nhưng lại "khó" tính ngược lại được **x** nếu chỉ biết giá trị băm **h**(**x**) (Kể cả khi biết hàm băm **h**).

3) Ứng dụng của hàm băm

1) Với bản tin dài x, thì chữ ký trên x cũng sẽ dài, như vậy tốn thời gian "ký", tốn bộ nhớ lưu giữ "chữ ký", tốn thời gian truyền "chữ ký" trên mạng.

Người ta dùng hàm băm **h** để tạo đại diện bản tin **z** = **h**(**x**), nó có độ dài ngắn (VD 128 bit) Sau đó ký trên **z**, như vậy chữ ký trên **z** sẽ nhỏ hơn rất nhiều so với chữ ký trên bản tin gốc x.

2) Hàm băm dùng để xác định tính toàn vẹn dữ liệu.

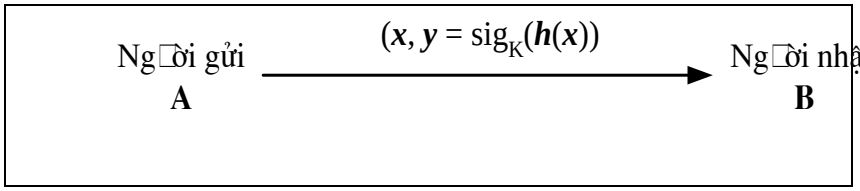
3) Hàm băm dùng để bảo mật một số dữ liệu đặc biệt, ví dụ bảo vệ mật khẩu, bảo vệ khóa mật mã, …

4.6.2.2. Các tính chất của Hàm băm

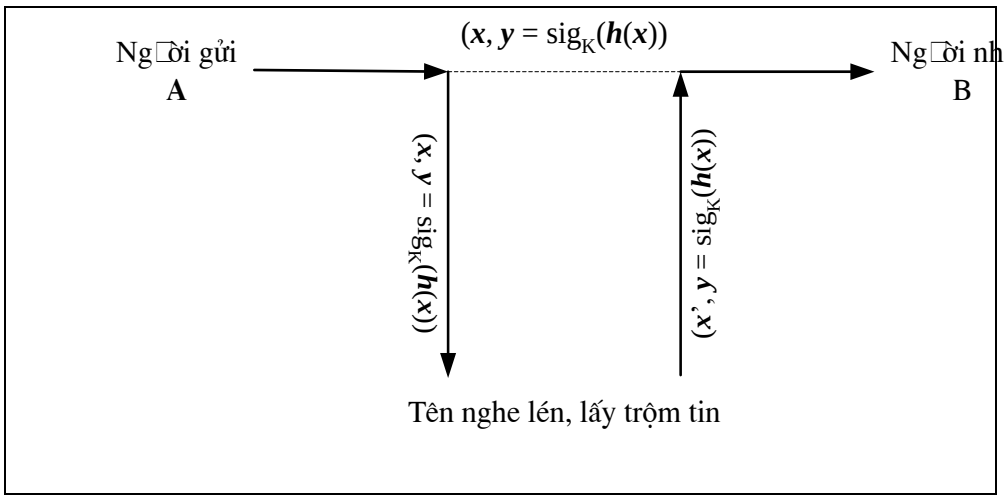
Tính chất 1: Hàm băm ***h*** là ***không va chạm* yếu**.

Ví dụ: Xét kiểu tấn công như sau: *Kiểu tấn công theo tính chất 1*.

* *Hình a*: *Cách đi đúng của thông tin:* thông tin được truyền đúng từ A đến B:



* Hình b: Thông tin bị lấy trộm và bị thay đổi trên đường truyền:



**** Kiểu tấn công theo tính chất 1:***

+ Người A gửi cho B bản tin (**x**, **y**) với **y** = sig_k (**h**(**x**)) B không nhận được (**x**, **y**) vì:

+ Trên đường truyền, tin bị lấy trộm. Tên trộm, bằng cách nào đó tìm được một bản tin **x**' ≠ **x** nhưng lại có **h**(**x**') = **h**(**x**) Hắn thay thế **x** bằng **x**', và chuyển tiếp (**x**', **y**) cho B.

+ Người B nhận được (**x**', **y**), và vẫn xác thực được thông tin đúng đắn. Do đó, để tránh kiểu tấn công như trên, hàm **h** phải thỏa mãn tính chất: **không va chạm yếu**.

**** Khái niệm:*** Hàm băm ***không va chạm* yếu**.

Hàm băm **h** được gọi là **không va chạm yếu**, nếu cho trước bức điện **x**, "khó" thể tính toán để tìm ra bức điện **x**' ≠ **x** mà **h**(**x**') = **h**(**x**)

Tính chất 2: Hàm băm **h** là ***không va chạm* mạnh**.

Ví dụ: Xét kiểu tấn công như sau: *Kiểu tấn công theo tính chất 2.*

+ Đầu tiên, tên giả mạo tìm được hai thông điệp khác nhau **x**' và **x** (**x**' ≠ **x**) mà có **h**(**x**') = **h**(**x**) (Ta coi bức thông điệp **x** là hợp lệ, còn **x**' là giả mạo)

+ Tiếp theo, hắn thuyết phục ông A ký vào bản tóm lược **h**(**x**) để nhận được **y**. Khi đó (**x**', **y**) là bức điện giả mạo nhưng hợp lệ vì **h**(**x**') = **h**(**x**)

Để tránh kiểu tấn công này, hàm **h** phải thỏa mãn tính chất: ***không va chạm* mạnh**.

**** Khái niệm:*** Hàm băm ***không va chạm* mạnh**.

Hàm băm **h** được gọi là ***không va chạm* mạnh** nếu "khó" thể tính toán để tìm ra hai bức thông điệp khác nhau **x**' và **x** (**x**' ≠ **x**) mà có **h**(**x**') = **h**(**x**)

Tính chất 3: Hàm băm **h** là ***hàm một chiều***.

Ví dụ: Xét kiểu tấn công như sau: *Kiểu tấn công theo tính chất 3.*

+ Người A gửi cho B thông tin (**x**, **z**, **y**) với **z** = **h**(**x**), **y** = sig_k (**z**)

+ Giả sử tên giả mạo tìm được bản tin **x'**, được tính ngược từ bản tóm lược

z = **h**(**x**)

+ Tên trộm thay thế bản tin **x** hợp lệ, bằng bản tin **x'** giả mạo, nhưng lại có

z = **h**(**x'**) Hắn ta ký số trên bản tóm lược **z** của **x'** bằng đúng chữ ký hợp lệ.

Nếu làm được như vậy, thì (**x'**, **z**, **y**) là bức điện giả mạo, nhưng hợp lệ.

Để tránh được kiểu tấn công này, hàm băm **h** cần thỏa mãn ***tính chất một chiều***.

**** Khái niệm:*** Hàm băm **một chiều**.

Hàm băm **h** được gọi là ***hàm một chiều*** nếu khi cho trước một bản tóm lược thông báo **z** thì "khó thể" tính toán để tìm ra thông điệp ban đầu **x** sao cho **h**(**x**) = **z**.

4.6.2.3. Các Hàm băm

Các hàm băm dòng MD (MD2, MD4, MD5) do Rivest đề xuất. Giá trị băm theo các thuật toán này có độ dài cố định là **128** bit. Hàm băm MD4 đưa ra vào năm 1990. Một năm sau phiên bản mạnh hơn là MD5 cũng được đề xuất.

Hàm băm an toàn SHA, phức tạp hơn nhiều, cũng dựa trên các phương pháp tương tự, được công bố trong Hồ sơ Liên bang năm 1992 và được chấp nhận làm tiêu chuẩn vào năm 1993 do Viện Tiêu Chuẩn và Công Nghệ Quốc Gia (NIST) Giá trị băm theo thuật toán này có độ dài cố định là **160** bit.

4.6.3. Hàm băm MD4

4.6.3.1. Khái niệm "Thông điệp đệm"

"***Thông điệp đệm***" (Messege Padding) là xâu bit có độ dài chia hết cho **512**.

"Thông điệp đệm" được lưu trong mảng **M** = M[0] M[1] … M[N-1].

Trong đó **M**[i] là xâu bit có độ dài 32 bit, gọi là *word*.

N ≡ 0 mod 16. (32 x 16 = 512)

M được xây dựng từ ***Bản tin gốc*** **a** bằng thuật toán:

1. **d** = 447 – (|**a**| mod 512) (= 512 nếu |**a**| mod 512 > 447)
2. Giả sử **l** là kí hiệu biểu diễn nhị phân của |**a**| mod 2^{64}, tl: | **l** | = **64**.
3. **M** = **a** || 1 || $0^{\mathbf{d}}$ || **l**

* Độ dài của xâu **a** || 1 || $0^{\mathbf{d}}$ qui ước là |**a**| + 1 + **d** = 448 mod 512.

* Độ dài của "Thông điệp đệm" **M** là

448 mod 512 +| **l** | = 448 mod 512 + **64** = 512 mod 512.

Chú ý: Vì **M** = **a** || 1 || $0^{\mathbf{d}}$ || **l** nên

d = |**M**| - (|**a**| + 1 + **l**) =

512 - (|**a**| + 1 + **64**) = 512 - (|**a**| +65) = 447 – (|**a**| mod 512)

Ví dụ

Xâu đầu vào là **a** = "ABC", xây dựng **M** như sau:

a: = "ABC" = "01000001 01000010 01000011". (Chú ý: 'A' =65)

* Độ dài tính theo bit của xâu ***a*****:** |***a***| = 24 bit

=> **d** = 447 – (|***a***| mod 512) = 423.

|***a***| + 1 + **d** = 24 + 1 + 423 = 448 mod 512.

* Biểu diễn nhị phân của độ dài xâu **a** là ***l***:

$\boldsymbol{l}$ = |***a***| mod 2^{64} = 24 mod 2^{64} = 24 = 16 + 8 = $2^4 + 2^3$ = ($\underbrace{00.....00}_{59 so}$ 11000)$_2$

=> Độ dài của l là |***l***| = |$\underbrace{00.....00}_{59 so}$ 11000| = 59 + 5 = 64.

M = **a** || **1** || 0^{d} || ***l***

=> M = 01000001 01000010 01000011 || **1** || $\underbrace{00.....00}_{423so}$ || $\underbrace{00.....00}_{59so}$ **11000**

M = M[0] M[1] … M[N-1], N ≡ 0 mod 16

M[0] = 01000001 01000010 01000011 10000000

M[1] = M[2] = ….. = M[13] = M[14] = $\underbrace{00.....00}_{32so}$

M[15] = 00000000 00000000 00000000 00011000

Trong việc xây dựng M, ta gắn số **1** đơn lẻ vào sau **a**, sau đó thêm tiếp các số 0 vào đủ để độ dài của M đồng dư với 448 modulo 512. Cuối cùng nối thêm 64 bit (chính là |***l***|) chứa biểu diễn nhị phân về độ dài ban đầu của ***x*** (được rút gọn theo modulo 2^{64} nếu cần)

Xâu kết quả M có độ dài chia hết cho 512. Vì thế khi chặt M thành các *word* 32 bit, số *word* nhận được là N sẽ chia hết cho 16.

Mục đích việc tạo ra mảng M – "thông điệp đệm" – là để các hàm băm xử lý trên từng khối (block) 512 bit, tức là 16 *word*, cùng một lúc.

4.6.3.2. Thuật toán băm MD5

INPUT : Thông điệp là một xâu ***a*** có độ dài ***b*** bit.

OUTPUT : Bản băm, đại diện cho thông điệp gốc, độ dài cố định **128** bit.

a) Tóm tắt thuật toán

Bước 1: **Khởi tạo các thanh ghi**

Có 4 thanh ghi để tính toán nhằm đưa ra các đoạn mã: A, B, C, D. Bản tóm lược của thông điệp được xây dựng như sự kết nối của các thanh ghi. Mỗi thanh ghi có độ dài 32 bit. Các thanh ghi này được khởi tạo giá trị hecxa.

word A := **67 45 23 01** *word* B := **ef cd ab 89**

word C := **98 ba dc fe** *word* D := **10 32 54 76**

Bước 2:

Xử lý thông điệp ***a*** trong 16 khối *word*, có nghĩa là xử lý cùng một lúc 16 *word* = 16 * 32 bit = 512 bit.

Chia mảng M thành các khối 512 bit, đưa từng khối 512 bit vào mảng T[j]. Mỗi lần xử lý một khối 512 bit. Lặp lại N/16 lần.

b) Thuật toán MD4

1/ A := **67 45 23 01** B := **ef cd ab 89**

C := **98 ba dc fe** D := **10 32 54 76**

2/ **FOR** i := 0 **TO** N/16 - 1 **DO**

for j := 0 **to** 15 **do** **T**[j] = M[16 i + j];

AA := A; BB := B;

CC := C; DD := D;

Mỗi lần xử lý 16 từ, mỗi từ 32 bit, tl: 512 bit.

3/ **Vòng 1**

Vòng 2

Vòng 3

4/ A = A + AA; B = B + BB; C = C + CC; D = D + DD;

Gán giá trị cho 4 biến AA, BB, CC, DD bằng giá trị 4 thanh ghi A, B, C, D tương ứng.

c) Các phép tính và các hàm dùng trong Thuật toán MD4

**** Các phép toán logic được sử dụng trong ba vòng.***

$X \wedge Y$ là phép toán AND theo bit giữa X và Y

$X \vee Y$ là phép toán OR theo bit giữa X và Y

$X \oplus Y$ là phép toán XOR theo bit giữa X và Y

$\neg X$ chỉ phép bù của X

$X + Y$ là phép cộng theo modulo 2^{32}

$X <<< s$ là phép dịch vòng trái X đi s vị trí ($0 \leq s \leq 31$)

**** Ba hàm F, G, H dùng tương ứng trong vòng 1, 2, 3.***

Mỗi hàm này là một hàm boolean tính theo bit.

$$F(X, Y, Z) = (X \wedge Y) \vee ((\neg X) \wedge Z)$$

$$G(X, Y, Z) = (X \wedge Y) \vee (X \wedge Z) \vee (Y \wedge Z)$$

$$H(X, Y, Z) = X \oplus Y \oplus Z$$

Ba vòng trong MD4 là hoàn toàn khác nhau. Mỗi vòng (3.1, 3.2, 3.3) gồm một trong 16 *word* trong T được xử lý. Các phép toán được thực hiện trong ba vòng tạo ra các giá trị mới trong bốn *thanh ghi*. Cuối cùng, bốn *thanh ghi* được cập nhật ở 3.4 bằng cách cộng ngược các giá trị lưu trước đó ở 2.3. Phép cộng này được xác định là cộng các số nguyên dương, được rút gọn theo modulo 2^{32}.

d) Ba vòng "băm".

Vòng 1

1. A = (A + F(B, C, D) + T[0]) <<< 3
2. D = (D + F(A, B, C) + T[1]) <<< 7
3. C = (C + F(D, A, B) + T[2]) <<< 11
4. B = (B + F(C, D, A) + T[3]) <<< 19
5. A = (A + F(B, C, D) + T[4]) <<< 3
6. D = (D + F(A, B, C) + T[5]) <<< 7
7. C = (C + F(D, A, B) + T[6]) <<< 11
8. B = (B + F(C, D, A) + T[7]) <<< 19
9. A = (A + F(B, C, D) + T[8]) <<< 3
10. D = (D + F(A, B, C) + T[9]) <<< 7
11. C = (C + F(D, A, B) + T[10]) <<< 11
12. B = (B + F(C, D, A) + T[11]) <<< 19
13. A = (A + F(B, C, D) + T[12]) <<< 3
14. D = (D + F(A, B, C) + T[13]) <<< 7
15. C = (C + F(D, A, B) + T[14]) <<< 11
16. B = (B + F(C, D, A) + T[15]) <<< 19

Kết quả của VD **a** *sau khi được xử lý qua vòng 1.*

1. 64B3DA82	5. 3D5E5934	9. 59798D5E	13. 7551AAC6
2. 34D8EB03	6. 489D5140	10. D206302D	14. 789B984F
3. B7BCB118	7. CCD14D6C	11. 753D6134	15. F55A1F31
4. 6D91B115	8. 454D0E92	12. F52AED08	16. ABA71E22

Vòng 2

1. A = (A + G(B, C, D) + T[0] + 5A827999) <<< 3
2. D = (D + G(A, B, C) + T[4] + 5A827999) <<< 5
3. C = (C + G(D, A, B) + T[8] + 5A827999) <<< 9
4. B = (B + G(C, D, A) + T[12] + 5A827999) <<< 13
5. A = (A + G(B, C, D) + T[1] + 5A827999) <<< 3
6. D = (D + G(A, B, C) + T[5] + 5A827999) <<< 5
7. C = (C + G(D, A, B) + T[9] + 5A827999) <<< 9
8. B = (B + G(C, D, A) + T[13] + 5A827999) <<< 13
9. A = (A + G(B, C, D) + T[2] + 5A827999) <<< 3
10. D = (D + G(A, B, C) + T[6] + 5A827999) <<< 5
11. C = (C + G(D, A, B) + T[10] + 5A827999) <<< 9
12. B = (B + G(C, D, A) + T[14] + 5A827999) <<< 13
13. A = (A + G(B, C, D) + T[3] + 5A827999) <<< 3
14. D = (D + G(A, B, C) + T[7] + 5A827999) <<< 5
15. C = (C + G(D, A, B) + T[11] + 5A827999) <<< 9
16. B = (B + G(C, D, A) + T[15] + 5A827999) <<< 13

Giá trị 5A827999 là một hằng số ở dạng hecxa có độ dài 32 bit

Kết quả của VD **a** *sau khi được xử lý qua vòng 2.*

1. 558C2E28	5. 558C2E28	9. 31E9FE4A	13. B60A11E6
2. 5A0E08F9	6. 5A0E08F9	10. 6F68E462	14. 2DED6D8E
3. F6A9B390	7. F6A9B390	11. D745F88A	15. A2870B31
4. 7876BC8F	8. 7876BC8F	12. 7050BC10	16. 4384D178

Vòng 3

1. A = (A + H(B, C, D) + T[0] + 6ED9EBA1) <<< 3
2. D = (D + H(A, B, C) + T[8] + 6ED9EBA1) <<< 9
3. C = (C + H(D, A, B) + T[4] + 6ED9EBA1) <<< 11
4. B = (B + H(C, D, A) + T[12] + 6ED9EBA1) <<< 15
5. A = (A + H(B, C, D) + T[2] + 6ED9EBA1) <<< 3
6. D = (D + H(A, B, C) + T[10] + 6ED9EBA1) <<< 9
7. C = (C + H(D, A, B) + T[6] + 6ED9EBA1) <<< 11
8. B = (B + H(C, D, A) + T[14] + 6ED9EBA1) <<< 15
9. A = (A + H(B, C, D) + T[1] + 6ED9EBA1) <<< 3
10. D = (D + H(A, B, C) + T[9] + 6ED9EBA1) <<< 9
11. C = (C + H(D, A, B) + T[5] + 6ED9EBA1) <<< 11
12. B = (B + H(C, D, A) + T[13] + 6ED9EBA1) <<< 15
13. A = (A + H(B, C, D) + T[3] + 6ED9EBA1) <<< 3
14. D = (D + H(A, B, C) + T[11] + 6ED9EBA1) <<< 9
15. C = (C + H(D, A, B) + T[7] + 6ED9EBA1) <<< 11
16. B = (B + H(C, D, A) + T[15] + 6ED9EBA1) <<< 15

Giá trị 6ED9EBA1 là một hằng số ở dạng hecxa có độ dài 32 bit

Kết quả của VD **a** *sau khi được xử lý qua vòng 3.*

1. 98A7C489	5. F3031C80	9. C02E826B	13. 03477E5E
2. E70B031C	6. 7D7A371B	10. F38DC78B	14. 77509F0A
3. A96B2FFA	7. 1C2487DE	11. E3C7F63B	15. FB3D792D
4. 58BE9F94	8. F7767709	12. 812AB00F	16. 23D73C06

e) Kết quả "băm"

Kết quả ra là đoạn mã có độ dài 128 bit, được thu gọn từ thông điệp **a** có độ dài ***b*** bit. Đoạn mã này thu được từ 4 thanh ghi A, B, C, D: bắt đầu từ byte thấp của thanh ghi A cho đến byte cao của thanh ghi D.

Với VD **a** = "ABC", kết quả cuối cùng là Đại diện văn bản:

A = 6A8CA15F

B = 671E4A93

C = 93F85626

D = 3409907C

Chú ý: A = A + AA = 03477E5E

67452301

= 6A8CA15F

BÀI TẬP CHƯƠNG 4. CHỮ KÝ SỐ.

Để hiểu cách thức Ký số và Kiểm thử chữ ký đối với từng sơ đồ ký số cụ thể, bài tập chương 4 tập trung vào việc lập chương trình Ký số và Kiểm thử chữ ký.

Bài tập

Viết chương trình thực hiện Ký số sau:

1/ Sơ đồ chữ ký số RSA.

2/ Sơ đồ chữ ký số Elgamal.

3/ Sơ đồ chuẩn chữ ký số DSS.

4/ Sơ đồ chữ ký không thể phủ nhận.

Mẫu Chương trình

* Mỗi chương trình ký số phải thực hiện các công việc theo thực đơn sau:

Thực đơn chính.

S. Ký số.

V. Kiểm thử chữ ký.

K. Kết thúc.

Chương 5. PHƯƠNG PHÁP ẨN GIẤU THÔNG TIN

5.1. Tổng quan về ẩn giấu thông tin

5.1.1. Khái niệm "Ẩn - giấu tin"

"***Ẩn- giấu tin***", tiếng Hy lạp là "**Steganography**", tiếng Anh: "**Covered Writing**".

"***Ẩn- giấu tin***" được hiểu là ***nhúng mẩu tin mật*** vào một ***vật mang tin*** khác, sao cho mắt thường khó phát hiện ra mẩu tin mật đó, mặt khác khó nhận biết được vật mang tin đã được giấu một tin mật.

Trong lịch sử, có nhiều câu truyện về "giấu tin" phục vụ mục đích quân sự:

Giấu tin bằng cách dùng "mực không mầu" để viết tin mật. Để xem tin mật, người nhận dùng thủ thuật cho hiện mầu.

Người ta "khắc" bản đồ kho báu lên đầu các thuỷ thủ, để tóc mọc che kín đi.

Quân Hy lạp đã thông báo cho nhau về âm mưu của kẻ địch, bằng cách "khắc tin" dưới lớp sáp của viên thuốc.

Trung hoa thời trung cổ, người ta ghi các hình tượng vào các vị trí nhất định trong một bức thư rồi gửi nó đi.

Khi có chữ viết, người ta giấu tin mật vào một bài thơ thông thường, bằng cách dùng các ký tự đầu tiên của mỗi từ. trong bài thơ này.

Công nghệ thông tin đã tạo ra những môi trường "Ẩn giấu tin" mới vô cùng tiện lợi và phong phú. Không chỉ "Ẩn giấu tin" trong văn bản, người ta còn có thể "Ẩn giấu tin" trong hình ảnh, âm thanh. Không chỉ "Ẩn giấu tin" trong các gói "dữ liệu", người ta có thể "Ẩn giấu tin" trong các phần mềm trên đường truyền tin. Cũng có thể "Ẩn giấu tin" ngay trong các khoảng trống hay các phân vùng ẩn của môi trường lưu trữ như đĩa cứng, đĩa mềm, USB. Đó chính là các phương pháp "***Ẩn- giấu tin số***".

Ngày nay "***Ẩn-giấu tin số***" không chỉ dùng cho mục đích quân sự, nghệ thuật "***Ẩn giấu tin số***" còn để phục vụ các mục đích tích cực như bảo vệ bản quyền các "tài liệu

số", ví dụ như "tranh ảnh số", "bản nhạc số", công trình khoa học hay bài văn bài thơ đã được "số hóa", …

Các tài liệu trên được "giấu" một định danh ghi bản quyền, phương pháp "Ẩn giấu tin số" này được gọi là "***Thuỷ ấn số***" (Watermaking)

Sự kiện ngày 11/09/2001 khiến cho các liên lạc bí mật qua mạng máy tính được quan tâm hơn nữa. Người ta đồn rằng Osma Bin Laden và quân khủng bố đã liên lạc với nhau bằng cách giấu thông tin vào ảnh trên Internet.

Năm 1990, các kết quả nghiên cứu đầu tiên về "giấu tin" với sự trợ giúp của máy tính, đã đặt nền móng cho sự phát triển phương pháp bí mật: "***Ẩn giấu tin số***".

Sự phát triển của Internet kéo theo sự phát triển của nhiều lĩnh vực khác. Internet đã tạo ra môi trường kinh doanh mới: Kinh doanh trên mạng máy tính. Ví dụ thương mại điện tử.

Kinh doanh trên mạng mang lại rất nhiều lợi ích, tuy nhiên cũng phát sinh những mặt tiêu cực như vi phạm bản quyền, giả mạo thương hiệu, … Trước tình thế đó, các phương pháp "***Ẩn giấu tin số***" đã khẳng định được chỗ đứng của mình. Chúng được áp dụng để "ghi dấu ấn" vào "sản phẩm số" các thông tin như chữ ký, nhãn thương hiệu, … để minh chứng cho sự hợp pháp của "sản phẩm số" đó, góp phần bảo vệ bản quyền "sản phẩm số".

Để phân biệt với "***Ẩn giấu tin***" theo nghĩa thông thường nhằm "che giấu" thông tin mật, ta tạm gọi loại "***Ẩn giấu tin***" để bảo vệ bản quyền "sản phẩm số" là "***Đánh giấu tài liệu số***", "***Đánh giấu sản phẩm số***" hay "***Ẩn tin***".

5.1.2. Các thành phần của Hệ "Ẩn - Giấu tin".

Các thành phần chính của một hệ **"Ẩn - Giấu tin"** trong ảnh gồm có:

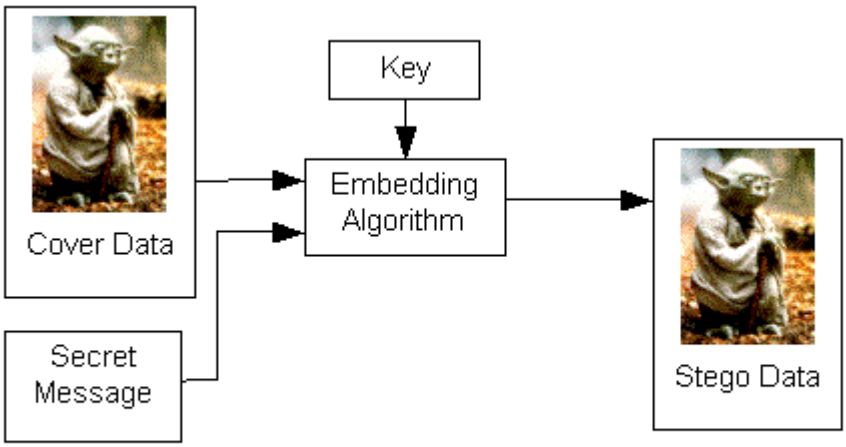
+ Mẩu tin mật: có thể là văn bản, hình ảnh, âm thanh (audio, video,...),
trong quá trình giấu tin, chúng được chuyển thành chuỗi các bit.

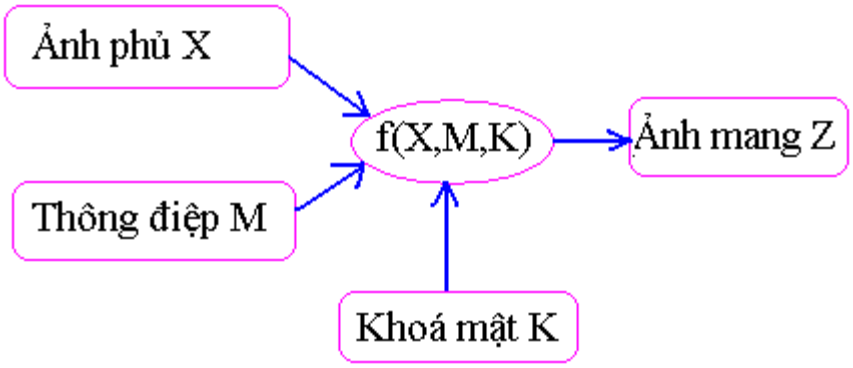
+ Môi trường sẽ chứa tin mật: Thường là ảnh, nên gọi là **Ảnh phủ**, **Ảnh gốc**.

+ Khoá K: Khoá viết mật, tham gia vào quá trình giấu tin để tăng tính bảo mật.

+ Môi trường đã chứa tin mật:

Thường là ảnh, nên gọi là **Ảnh mang**, là ảnh sau khi đã nhúng tin mật vào.

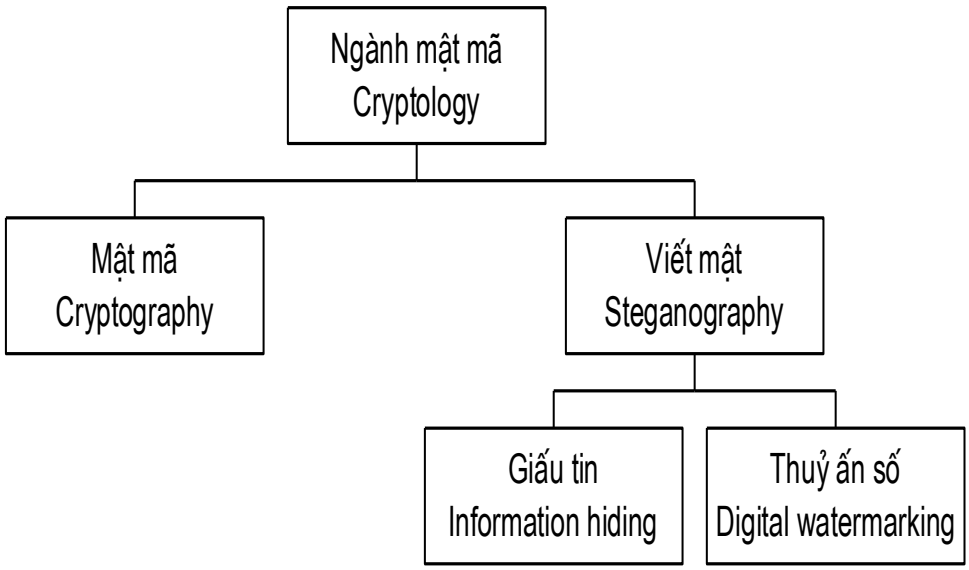




Hình 1: Sơ đồ "Ẩn - giấu tin" trong ảnh

5.1.3. Ẩn - Giấu tin và Mật mã.

Có thể xem "***Ẩn - giấu tin***" là một nhánh của ngành mật mã với mục tiêu là nghiên cứu các phương pháp "***che giấu*** " thông tin mật.



Hình 2: Các lĩnh vực nghiên cứu của Mật mã học

"Ẩn - giấu tin" và "Mã hóa" tuy cùng có mục đính là để đối phương "khó" phát hiện ra tin cần giấu, tuy nhiên nó khác với mã hóa ở chỗ:

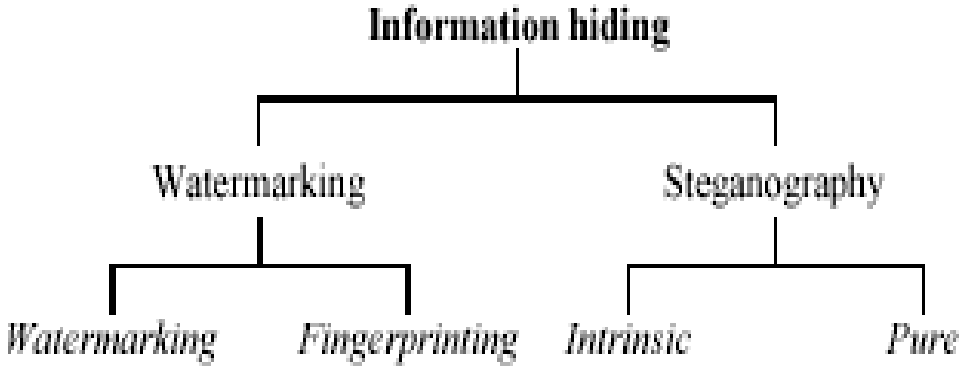
+ **"Mã hóa"** là giấu đi "***ý nghĩa***" của thông tin.

+ **"Ẩn - Giấu tin"** là giấu đi "***sự hiện diện***" của thông tin.

Về bản chất, **"Ẩn - giấu tin"** gần với "***Nén tin***" hơn.

5.1.4. Phân loại "Ẩn - Giấu tin"

Trong lĩnh vực bảo mật thông tin, "Ẩn - Giấu tin" bao gồm các vấn đề sau:



Hình 3: Các vấn đề nghiên cứu trong "Ẩn - giấu tin" ***(Information Hiding)***

1) Giấu tin (**Steganography**) là kỹ thuật ***nhúng*** "**mẩu tin mật**" (Mẩu tin cần giữ bí mật) vào "**môi trường giấu tin**" (môi trường phủ),

+ "***Giấu tin có xử lý***" (**Intrinsic Steganography**) là một dạng "***Giấu tin*** ", trong đó để tăng bảo mật, có thể phải dùng ***khoá viết mật*** (Stego-key) Để giải mã, người ta cũng phải có ***khoá viết mật*** đó.

Khoá viết mật không phải dùng để mã hóa mẩu tin, nó có thể là khoá dùng để ***sinh ra "hàm băm"***, phục vụ "**rải tin mật**" vào môi trường giấu tin.

+ "***Giấu tin đơn thuần***" (**Pure Steganography**) là một dạng "***Giấu tin*** ", trong đó không dùng khoá viết mật để giấu tin, tức là chỉ giấu tin đơn thuần vào môi trường giấu tin.

2) "Thuỷ ấn số" (**Watermarking**) là kỹ thuật ***nhúng*** "**giấu ấn số**" (tin giấu)

vào một "**tài liệu số**" ("sản phẩm số"), nhằm chứng thực (đánh dấu, xác thực) nguồn gốc hay chủ sở hữu của "**tài liệu số**" này.

Ví dụ "**giấu ấn số**" dùng để xác nhận bản quyền một "tác phẩm.số".

Tạm gọi ***"Thuỷ ấn số"*** là ***"Ẩn tin"***, để phân biệt với ***"Giấu tin"***.

+ "***Dấu vân tay***" (**Fingerprinting**) là một dạng "***Thuỷ ấn số*****"**, trong đó "**giấu ấn**" (tin giấu) là ***một định danh duy nhất*** (ví dụ định danh người dùng)

3) So sánh *"Ẩn tin"* (Watermarking) và ***"Giấu tin"*** **(Steganography)**

Về mặt hình thức, ***"Ẩn tin"*** giống ***"Giấu tin"*** ở chỗ đều tìm cách nhúng thông tin vào một môi trường.

Về mặt nội dung, "***Ẩn tin***" (thuỷ ấn) có một số điểm khác so với *"**Giấu tin**"*:

* ***Về mục tiêu***:

+ Mục tiêu của "***Ẩn tin***" là ***nhúng "mẩu tin"*** thường là biểu tượng, chữ ký, dấu nhỏ đặc trưng vào môi trường phủ, nhằm phục vụ việc chứng thực bản quyền tài liệu.

Như vậy "***mẩu tin***" **c**ần nhúng (để làm biểu tượng xác thực) không nhất thiết phải là bí mật, nhiều khi cần lộ ra cho mọi người biết để mà "dè chừng" !

+ "***Ẩn tin***" có thể vô hình hoặc hữu trên vật mang tin.

"***Ẩn tin***" tìm cách biến "**tin giấu**" thành ***một thuộc tính*** của vật mang tin.

Mục đích của "***Ẩn tin***" là ***bảo vệ môi trường giấu tin***.

+ Mục tiêu của *"**Giấu tin**"*, là ***nhúng "mẩu tin"*** thường là bí mật, vào môi trường phủ, sau đó có thể lấy ra (tách lại) tin mật từ môi trường phủ.

+ *"**Giấu tin**"* không cho phép nhìn thấy (bằng mắt) "**tin giấu**" trên vật mang tin.

Mục đích của "***Giấu tin***" là ***bảo vệ tin được giấu***.

* ***Về đánh giá hiệu quả***: Theo tiêu chí hay chỉ tiêu nào ?

+ Chỉ tiêu quan trọng nhất của "***Ẩn tin***" là ***tính bền vững*** của tin được giấu.

+ Chỉ tiêu quan trọng nhất của "***Giấu tin***" là ***dung lượng*** của tin được giấu.

5.1.5. Các tính chất của "Ẩn - Giấu tin" trong Ảnh.

Hiện nay có nhiều phương pháp **"**Ẩn - Giấu tin**"** trong ảnh. Để đánh giá chất lượng của một phương pháp **"**Ẩn - Giấu tin**"**, người ta dựa vào một số tiêu chí sau:

1) Bảo đảm Tính "vô hình" (Tính bí mật)

"*Ẩn - Giấu tin*" trong ảnh sẽ làm biến đổi ảnh mang tin. Tính "vô hình" thể hiện mức độ biến đổi ảnh mang. Phương pháp **"*Ẩn - Giấu tin*"** tốt, sẽ làm cho thông tin mật trở nên "***vô hình***" (bí mật) trên ảnh mang, người ta "khó thể" nhận ra trong ảnh có ẩn chứa thông tin mật.

Riêng với **"Ẩn tin"** thì trong thực tế không phải khi nào cũng cố gắng để đạt được tính vô hình cao nhất, ví dụ trong truyền hình, người ta gắn ***hình ảnh mờ***, gọi là "***thuỷ ấn***" để bảo vệ bản quyền bản tin.

2) Khả năng chống giả mạo (Tính toàn vẹn)

Mục đích của "Giấu tin" là để truyền đi thông tin mật. Nếu không thể do thám tin mật, thì kẻ địch cũng cố tìm cách làm sai lạc tin mật, làm giả mạo tin mật để gây bất lợi cho đối phương. Phương pháp "Giấu tin" tốt phải đảm bảo tin mật không bị tấn công một cách chủ động trên cơ sở những hiểu biết về thuật toán nhúng tin và có ảnh mang (nhưng không biết khoá "Giấu tin"). Đối với "Ẩn tin" thì khả năng chống giả mạo là yêu cầu vô cùng quan trọng, vì có như vậy mới bảo vệ được bản quyền, minh chứng tính pháp lý của sản phẩm. Tóm lại, "Giấu tin" hay "Ẩn tin" đều cần yêu cầu "khả năng chống giả mạo".

3) Tính bền vững

Sau khi "Ẩn - giấu tin" vào ảnh mang, bản thân ảnh mang có thể phải qua các biến đổi khác nhau như lọc (tuyến tính, phi tuyến), thêm nhiễu, làm sắc nét, mờ nhạt, quay, nén mất dữ liệu, ... ***Tính bền vững*** là thước đo "***sự nguyên vẹn***" của ***tin mật*** sau những biến đổi như vậy.

4) Dung lượng tin giấu

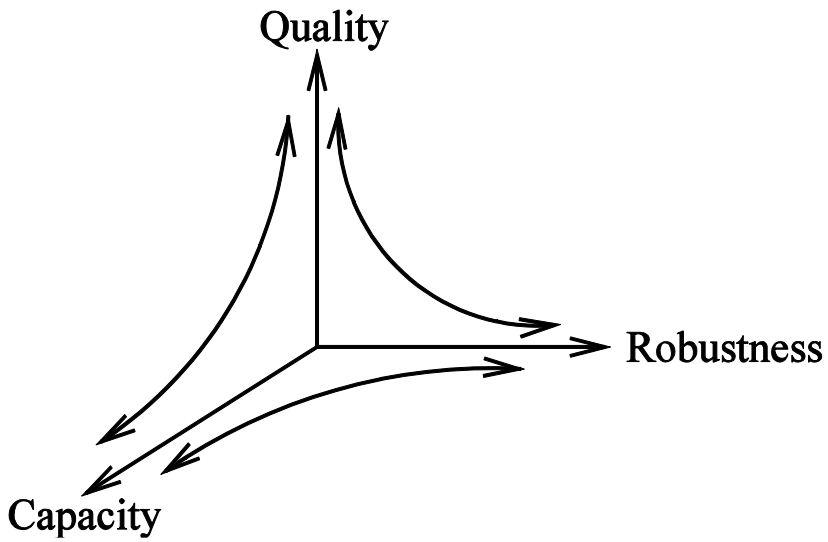
Dung lượng tin giấu được tính bằng tỷ lệ của lượng tin cần giấu so với kích thước ảnh mang tin. Các phương pháp đều cố gắng giấu được nhiều tin trong ảnh nhưng vẫn giữ được bí mật.

5) Độ phức tạp tính toán

"Độ phức tạp" của thuật toán "Ẩn - giấu tin" và "Giải tin" (tách tin) cũng là một chỉ tiêu quan trọng để đánh giá một phương pháp "Ẩn - giấu tin" trong ảnh. Chỉ tiêu này cho chúng ta biết "tài nguyên" (thời gian và bộ nhớ) tốn bao nhiêu dùng cho một phương pháp "Ẩn - giấu tin". Với chủ nhân "Ẩn - giấu tin" thì thời gian thực hiện phải "nhanh", nhưng với kẻ thám tin thì "Tách tin" phải là bài toán "khó". Ví dụ bài toán "Tách tin" từ "Thuỷ ấn" để đánh dấu bản quyền cần phải là bài toán "khó", thì mới chịu được sự tấn công của tin tặc nhằm phá huỷ "Thuỷ ấn".

Chú ý:

Trong thực tế "Ẩn - giấu tin", người ta luôn phải cân nhắc giữa *chất lượng* **(tính bí mật, tính toàn vẹn, tính bền vững) và** *dung lượng* **tin cần giấu, độ phức tạp của việc "Ẩn - giấu tin" và độ phức tạp của việc phá hoại "Ẩn - giấu tin".**



***Hình 4*: Cân nhắc giữa chất lượng, dung lượng, tính bền vững**

5.1.6. Vấn đề tấn công Hệ thống "Ẩn - Giấu tin".

Tấn công một hệ **"**Ẩn - Giấu tin**"** được gọi là "***Steganalysis***". Đó là các phương pháp để phát hiện, phá huỷ, trích rút hay sửa đổi tin mật. Nghiên cứu các biện pháp của kẻ tấn công, sẽ hữu ích cho việc thiết kế một hệ **"**Ẩn - Giấu tin**"** tốt.

Việc tấn công được coi là thành công hay không tuỳ theo ứng dụng. Đối với liên lạc bí mật, việc phát hiện và chứng minh được một ảnh có chứa tin mật được coi là thành công. Đối với bảo vệ bản quyền hay chống giả mạo, thì việc tấn công được coi là thành

công nếu không chỉ phát hiện ra "thuỷ ấn", mà còn phá huỷ hay sửa đổi nó, nhưng không làm giảm chất lượng của ảnh mang.

Có điểm giống nhau giữa "mã hoá" và "giấu tin" là người ta giả thiết ***thám tin biết trước phương pháp mã hoá hay giấu tin***. Như vậy việc thám tin theo một phương pháp cụ thể (mã hoá hay giấu tin) phụ thuộc vào "***khoá***", chứ không phải phụ thuộc vào độ phức tạp của phương pháp này (Nguyên lý Kerkhoff)

***Thám tin trong* "Ẩn - Giấu tin"**:

Tương tự như thám mã trong mã hóa, các kỹ thuật thám tin trong giấu tin cũng được chia thành làm năm nhóm:

- Biết ảnh mang tin.
- Biết ảnh gốc (ảnh sẽ mang tin) và ảnh mang tin.
- Biết có tin giấu trong ảnh mang tin.
- Biết thuật toán giấu tin.
- Biết thuật toán trích (tách) tin mật.

Thám tin phát hiện "thuỷ ấn" hay "tin mật" có thể thực hiện bằng cách phân tích vùng ***nhiễu quá mức*** trên ảnh. Tin tặc kinh nghiệm có thể nhận thấy các vùng nhiễu này bằng mắt thường. Nếu biết được ảnh gốc thì việc thám tin còn đơn giản hơn nữa, vì khi đó có thể so sánh ảnh mang tin với ảnh gốc để tách nhiễu.

Nếu thám tin biết được có tin ẩn giấu, người ta có thể tạo ra các cặp ảnh gốc và ảnh mang để phân tích và xét xem liệu ảnh đang tìm hiểu có mang dấu ấn của chữ ký hay tin mật không.

Việc phá tin mật có thể đơn giản hay phức tạp tuỳ thuộc vào phương pháp "Ẩn - giấu tin". Đối với phương pháp nhúng tin vào "bit có trọng số thấp", thì việc phá tin mật chỉ đơn thuần là thay đổi lại các bit này, như vậy ảnh mang tin trở về trạng thái ban đầu.

Phá tin mật đối với các phương pháp "***Ẩn tin***", mà vẫn giữ nguyên ảnh mang là một việc khó. Vì mục tiêu của "thuỷ ấn" là phải đạt được độ bền vững sao cho nếu có ai đó phá "thuỷ ấn", thì cũng làm hỏng ngay cả ảnh gốc.

Thông thường người ta tìm cách áp dụng nhiều phép biến đổi ảnh với hy vọng rằng: tuỳ từng phép biến đổi không có tác dụng, nhưng tổ hợp của chúng có thể giúp cho việc phá huỷ "thuỷ ấn" mà vẫn giữ được ảnh mang.

Nếu biết tin mật và ảnh mang tin, thì cơ hội phá tin mật sẽ cao hơn.

Nếu biết thuật toán "Ẩn - giấu tin", thám tin có thể dùng nó thử "Ẩn - giấu tin" lên nhiều ảnh khác nhau, qua đó dùng phương pháp thống kê để tìm ra các quy luật gây nhiễu, cũng như dùng nó để kiểm thử một ảnh có mang tin mật hay không.

Việc thám tin khó nhất đó là sửa đổi tin trong ảnh mang và suy ra được "***khoá viết mật***" (Stego-key) dùng để nhúng tin. Nếu biết khoá viết mật, kẻ thám tin có thể làm giả các tin khác giống như nó được gửi đi từ chính chủ.

Phương pháp thám tin để biết thuật toán "Ẩn - giấu tin" và thuật toán "tách tin" hay được dùng trong các hệ thám tin. Nhiều kỹ thuật thám tin trong "Ẩn - giấu tin" được chuyển sang từ kỹ thuật thám mã (trong mã hóa).

5.1.7. Các ứng dụng của "Ẩn - Giấu tin".

5.1.7. 1. Liên lạc bí mật.

Bản mã của tin mật có thể gây ra sự chú ý của tin tặc, nhưng tin mật được giấu vào trong môi trường nào đó, rồi gửi đi trên mạng máy tính, thì ít gây ra sự chú ý của tin tặc. Đó là một ứng dụng của **"**Giấu tin**"**.

Hiện nay người ta phối hợp đồng thời nhiều giải pháp để truyền tin mật trên mạng công khai: Đầu tiên tin mật được nén tin, sau đó mã hóa bản tin nén, cuối cùng giấu bản mã vào trong môi trường nào đó.

5.1.7. 2. Bảo vệ bản quyền.

1) "***Thuỷ ấn***" (Watermark)

+ Một biểu tượng bí mật gọi là "***Thuỷ ấn***" (Watermark) được "***nhúng***" vào trong một tài liệu (hình ảnh, âm thanh, …) để xác nhận quyền sở hữu về tài liệu.

+ "*Thuỷ ấn*" được đính lên tranh ảnh khi bán hoặc phân phối, thêm vào đó có thể gán một "nhãn thời gian" (Time stamp) để chống giả mạo.

+ "*Thuỷ ấn*" cũng được dùng để phát hiện xem các ảnh có bị sửa đổi hay không. Việc phát hiện "*Thuỷ ấn*" được thực hiện bằng thống kê, so sánh độ tương quan hoặc bằng cách đo đạc xác định chất lượng của "*Thuỷ ấn*" trong ảnh mang. [iv].

2) "*Điểm chỉ số*": Điểm chỉ số tương tự như số Seri của phần mềm [v].

"*Điểm chỉ số*" dùng để chuyển thông tin về người nhận "sản phẩm số" (không phải chủ sở hữu), nhằm chứng thực bản sao duy nhất của sản phẩm.

3) "*Gán nhãn*":

Tiêu đề, chú giải, nhãn thời gian … có thể được "nhúng" vào "sản phẩm số", Gắn tên người lên ảnh của họ, gắn tên địa phương lên bản đồ. Khi đó nếu sao chép ảnh thì cũng sẽ sao chép cả thông tin đã "nhúng" vào nó. Chủ sở hữu của sản phẩm, người có "khoá viết mật" (Stego-Key) có thể tách ra và xem các chú giải.

Trong một cơ sở dữ liệu ảnh, người ta có thể "nhúng" các ***từ khoá***, để các động cơ tìm kiếm có thể tìm nhanh một bức ảnh nào đó. Nếu là một khung ảnh cho cả một đoạn phim, người ta có thể gán cả thời điểm diễn ra sự kiện (timing) để đồng bộ hình ảnh với âm thanh. Người ta cũng có thể gán số lần mà hình ảnh được xem, để tính tiền thanh toán.

5.1.8. Một số chương trình "Ẩn - Giấu tin".

1) Chương trình Hide and Seek v4.1

Chương trình này của Colin Maroney, chạy dưới hệ điều hành DOS, để giấu tin vào các ảnh GIF. Nó thực hiện giấu tin vào ảnh mang một cách ngẫu nhiên, do đó nếu lượng tin cần giấu nhỏ thì tin sẽ được rải đều khắp ảnh mang. Nếu lượng tin nhiều, thì các vùng thay đổi dầy hơn, vì vậy dễ bị phát hiện.

2) Chương trình StegoDos

Chương trình chạy dưới hệ điều hành DOS, sử dụng ảnh mang 320 x 200 điểm ảnh và 256 mầu.

3) Chương trình White Noise Storm

Chương trình này của Ray (Arsen) Arachelian, dễ dùng hơn và nhúng được nhiều tin hơn các chương trình trước. Ảnh mang không cần có kích thước cố định, tính vô hình cao.

4) Chương trình S-Tools for Windows

Một chương trình giấu ảnh tốt. Có thể giấu tin trong ảnh BMP, GIF, tệp âm thanh WAV, các vùng chưa dùng đến của đĩa mềm. Giao diện đồ hoạ kéo thả. Để giấu tin chỉ cần kéo biểu tượng tệp tin cần giấu và thả lên ảnh.

Một yếu tố khác mà các hệ giấu tin nhắm tới và khai thác, đó là những điểm yếu trong hệ thống thị giác con người. Một trong những phương pháp giấu tin là tạo ra các

mặt nạ giác quan để đánh lừa mắt người. Vậy nên các nghiên cứu về phương pháp giấu tin trong ảnh có liên quan mật thiết với lĩnh vực xử lý ảnh, lý thuyết mật mã và các kiến thức về hệ thống thị giác.

5.2. Phương pháp giấu tin trong ảnh

5.2.1. Giấu tin trong ảnh đen trắng

Ảnh đen trắng số được thể hiện như một ***ma trận*** điểm ảnh gồm số 0 hay 1. Giấu tin trong ảnh đen trắng là việc khó khăn, vì dễ bị nhận biết bằng mắt thường. Số lượng tin giấu là hạn chế. Ảnh đen trắng ngày càng ít được dùng, do đó việc nghiên cứu giấu tin trong loại ảnh này là ít thực tế, tuy nhiên ta cũng cần phải biết.

5.2.1.1. Thuật toán giấu tin sử dụng tính "chẵn lẻ" của tổng số bit 1.

1) Ý tưởng:

Chia ảnh mang thành các khối nhỏ. Mỗi khối nhỏ sẽ được "***gài***" 1 bit của tin cần giấu. Dựa vào tính "chẵn lẻ" của tổng số các bit 1 trong khối để qui định giấu bit 1 hay bit 0.

Cụ thể là sau khi giấu, thì tổng số các bit 1 trong khối và bit cần giấu sẽ có cùng tính "chẵn lẻ".

2) Thuật toán "giấu tin":

Input: **FF** là File ảnh Bitmap đen trắng (sẽ mang tin giấu); **Fb** là File tin cần giấu;

K là khóa bí mật, đó là kích thước của khối nhỏ sẽ được tách từ **FF**.

Output: **FF** * là File ảnh đã được giấu file tin mật **Fb**, (**FF** *: ảnh đã mang tin giấu)

Bước 1: **Tiền xử lý.**

+ Chuyển File tin cần giấu **Fb** sang xâu bit nhị phân **b**.

+ Đọc Header của ảnh, đọc bảng màu, để lấy thông tin về ảnh.

Đọc phần dữ liệu ảnh vào mảng 2 chiều (ma trận) **F**.

Bước 2: **Giấu tin.**

Input: **F** là ma trận ảnh mang, **b** là xâu bit bí mật cần giấu.

K là khóa bí mật, đó là kích thước của khối nhỏ (được xác định trước)

Output: **F** * là ma trận ảnh đã được giấu xâu bit bí mật **b**.

B1: Chia ảnh mang **F** thành các khối nhỏ với kích thước **K**.

B2: Theo một thứ tự xác định trước, xét từng khối nhỏ:

+ Nếu muốn giấu bit **1** vào một khối thì phải thỏa mãn điều kiện:

(**L**): Tổng số các bit **1** trong khối đó là số "***lẻ***" (tức là cùng tính "***lẻ***" như **1**)

+ Nếu muốn giấu bit **1** vào một khối, nhưng không thỏa mãn điều kiện (**L**) (Tức là tổng số các bit **1** trong khối đó là số "***chẵn***"), thì trong khối đó chọn ngẫu nhiên một bit và thay đổi giá trị của nó (từ 0 đổi sang 1 hay từ 1 đổi sang 0)

Bằng cách đó, khối mang tin sẽ thỏa mãn điều kiện (**L**)

+ Nếu muốn giấu bit **0** vào một khối thì phải thỏa mãn điều kiện:

(**C**): Tổng số các bit **1** trong khối đó là số "***chẵn***" (tức là cùng tính "***chẵn***" như **0**)

+ Nếu muốn giấu bit **0** vào một khối, nhưng không thỏa mãn điều kiện (**C**) (Tức là tổng số các bit **1** trong khối đó là số "***lẻ***"), thì trong khối đó chọn ngẫu nhiên một bit và thay đổi giá trị của nó (từ 0 đổi sang 1 hay từ 1 đổi sang 0)

Bằng cách đó, khối mang tin sẽ thỏa mãn điều kiện (**C**)

3) Thuật toán tách "tin giấu":

Input: $\mathbf{F}^*$ là ảnh đã được giấu dãy bit bí mật **b**.

K là khóa bí mật, đó là kích thước của khối nhỏ (được xác định trước)

Output: **F** là ảnh mang (ảnh trước khi giấu tin mật) **b** là dãy bit bí mật cần giấu.

B0: Đọc Header của ảnh, đọc bảng màu, để lấy thông tin về ảnh.

Đọc phần dữ liệu ảnh vào mảng 2 chiều **F**.

B1: Chia ảnh mang **F** thành các khối nhỏ với kích thước **K**.

B2: Theo một thứ tự xác định trước, xét từng khối nhỏ:

Nếu tổng số bít **1** là "***lẻ***" thì ta thu được bit giấu là **1**.

Nếu tổng số bít **1** là "***chẵn***" thì ta thu được bit giấu là **0**.

Chú ý:

Độ an toàn của thuật toán giấu tin không cao, vì chỉ cần biết khóa **K** (tức là kích thước các khối giấu tin), là có thể dễ dàng tách được tin mật.

3) Ví dụ: Giấu bit **1** vào khối **V** sau:

Vì **V** có 6 bit 1 (một số chẵn các bit 1), nên để giấu bit 1 vào **V**, ta phải chọn ngẫu nhiên 1 bit và đổi giá trị.

$\mathbf{v_{13}} = \mathbf{1}$

1	0	**1**	1

0	0	1	0
1	1	0	0
0	0	0	0

Ví dụ chọn phần tử $\mathbf{v_{13}} = \mathbf{1}$, thay bằng $\mathbf{v_{13}} = \mathbf{0}$, ta có khối giấu tin là $\mathbf{V}^*$.

Bit **1** đã được giấu, $\mathbf{V}^*$ có 5 bit 1 (một số lẻ các bit 1)

$\mathbf{v_{13}} = \mathbf{0}$

1	0	**0**	1
0	0	1	0
1	1	0	0
0	0	0	0

5.2.1.2. Thuật toán giấu tin trên ảnh "đen trắng" của M.Y.Wu - J.H.Lee

1) Ý tưởng:

Thuật toán giấu tin kinh điển trong ảnh đen trắng của M.Y.Wu-.H.Lee[vi], với mục tiêu là giấu được càng nhiều tin vào trong ảnh càng tốt. Ý tưởng chính của thuật toán là chia ảnh thành các khối bằng nhau, ***tìm khối*** nào ***ít bị phát hiện nhất***, (tức là vùng thứ yếu trên ảnh), giấu một thông tin vào khối đó.

2) Thuật toán:

Input: **F** là ảnh mang; **b** là bit bí mật cần giấu; **K** là khóa bí mật (ma trận m x n)

Output: **F** là ảnh đã được giấu bit **b** bí mật.

Ký hiệu SUM(**F**) là số bit **1** có trong ma trận **F**.

B1: Chia **F** thành các khối nhỏ $\mathbf{F_i}$ có kích thước m x n (như ma trận **K**)

B2: Với mỗi khối nhỏ $\mathbf{F_i}$, kiểm tra điều kiện: $0 < \text{SUM}(\mathbf{F_i} \wedge \mathbf{K}) < \text{SUM}(\mathbf{K})$

Nếu đúng thì chuyển B3, giấu bit **b** vào $\mathbf{F_i}$; Nếu không đúng thì giữ nguyên $\mathbf{F_i}$.

B3: Khi giấu bit **b** vào $\mathbf{F_i}$, thay đổi $\mathbf{F_i}$ như sau:

IF (SUM ($\mathbf{F_i} \wedge \mathbf{K}$) mod 2 = **b**) **THEN** giữ nguyên $\mathbf{F_i}$

ELSE

If (SUM($\mathbf{F_i} \wedge \mathbf{K}$) = **1**) **Then**

Chọn ngẫu nhiên 1 bít thoả mãn ($[\mathbf{F_i}]_{j\,k} = \mathbf{0}$ & $[\mathbf{K}]_{j\,k} = \mathbf{1}$), lật $[\mathbf{F_i}]_{j\,k}$ thành **1**;

Else

if (SUM($\mathbf{F_i} \wedge \mathbf{K}$) = SUM(**K**) – 1) **then**

Chọn ngẫu nhiên 1 bít thoả mãn ($[\mathbf{F_i}]_{j\,k} = \mathbf{1}$ & $[\mathbf{K}]_{j\,k} = \mathbf{1}$), lật $[\mathbf{F_i}]_{j\,k}$ thành **0**;

else Chọn ngẫu nhiên 1 bít thoả mãn $[\mathbf{K}]_{j\,k} = 1$,

lật bít $[\mathbf{F_i}]_{j\,k}$ từ 0 thành 1, hay từ 1 thành 0;

Chú ý

Nếu m và n đủ lớn, thì sự thay đổi trên ảnh mang không dễ gì bị phát hiện bằng mặt thường. Có một số hướng cải tiến cho các thuật toán trên nhằm mục đích giấu được nhiều bit hơn vào khối ảnh.

5.2.2. Giấu tin trong ảnh màu

Có nhiều phương pháp giấu tin trong ảnh mầu hơn trong ảnh đen trắng. Tin giấu trong ảnh mầu khó bị phát hiện hơn trong ảnh đen trắng.

Ảnh mầu "số" là một mảng các bit thể hiện cường độ sáng tại mỗi điểm ảnh (**pixel**), mỗi điểm ảnh thể hiện bằng **8** bits. Một ảnh 640 x 480 pixels, sử dụng 256 mầu là phổ biến. Một ảnh như vậy có thể chứa chừng 300 kilobits dữ liệu.

5.2.2.1. Các yêu cầu kỹ thuật

Các kỹ thuật giấu tin trong ảnh màu phải đáp ứng các yêu cầu sau:

1/ Chất lượng ảnh mang vẫn bảo đảm, tin giấu không nhìn được bằng mắt thường.

2/ Tin giấu phải được mã hoá trực tiếp vào ảnh mang, chứ không vào phần khác, như vậy mới giữ được cho nhiều dạng tệp ảnh khác nhau.

3/ Tin giấu phải bền vững với các sửa đổi và tấn công từ bên ngoài. Ví dụ, nhiễu trên đường truyền, lọc, lấy mẫu, cắt xén, mã hoá, nén dữ liệu, in, quét, biến đổi số sang "tương tự" và ngược lại, tác động đến tin giấu là ít nhất.

4/ Đảm bảo toàn vẹn dữ liệu, vì điều khó tránh khỏi là tin giấu cũng sẽ bị thay đổi, nếu biến đổi ảnh mang.

5/ Chú ý các phương pháp giấu tin cho phép phục hồi tin giấu, không cần ảnh gốc.

5.2.2.2. Phương pháp giấu tin trong ảnh màu

a) Phân nhóm phương pháp giấu tin theo "kỹ thuật".

Theo phương pháp này, các phương pháp giấu tin trong ảnh hiện nay đều thuộc một trong ba nhóm sau.

1/ Giấu tin mật vào các "bít có trọng số thấp" (**LSB**: Least Significant Bit)

Nhóm phương pháp nhúng tin giấu vào các "***bít có trọng số thấp***" của ảnh hay được áp dụng trên các ảnh ***Bitmap*** không nén và các ảnh dùng bảng mầu (như GIF, TIF) Ý tưởng chính của phương pháp này là lấy từng bít của mẩu tin mật, rải nó lên ảnh mang, gài vào các bít có trọng số thấp.

2/ Giấu tin dựa vào kỹ thuật "***biến đổi ảnh***".

Nhóm phương pháp dựa vào kỹ thuật "***biến đổi ảnh***", lợi dụng việc biến đổi ảnh từ miền biểu diễn này sang miền biểu diễn khác, để giấu các bít tin mật.

Ví dụ biến đổi miền không gian sang miền tần số.

Một ví dụ về hệ thống dùng phương pháp này là "**Jpeg-Jsteg**" [4]. Hệ thống này nhúng tin bằng cách điều chế các hệ số của phép biến đổi "***Cosin rời rạc***", theo các bít tin cần giấu và sự làm tròn lỗi khi lượng hoá.

Một số phương pháp khác thuộc nhóm này sử dụng ảnh như mô hình vật lý với các dải phổ thể hiện mức năng lượng. Khi đó giấu tin giống như việc điều chế một tín hiệu dải hẹp vào một dải tần rộng (ảnh phủ, ảnh mang)

3/ Giấu tin sử dụng "***mặt nạ***" ***giác quan***.

Nhóm phương pháp dùng "***mặt nạ***" giác quan, dựa trên nguyên lý "đánh lừa" hệ thống giác quan con người. Một số điểm yếu của hệ thống giác quan là:

+ Hiệu ứng "mặt nạ" của các cạnh.

+ Sự nhậy cảm đối với độ tương phản là một hàm của miền tần số.

+ Khả năng nhạy cảm kém đối với các thay đổi nhỏ trong độ chói trên các mảng ảnh có cấu tạo ngẫu nhiên.

+ Sự nhạy cảm kém đối với các tần số miền không gian thấp, ví dụ như sự thay đổi liên tục của độ sáng trên ảnh.

"***Mặt nạ***" ở đây ám chỉ hiện tượng mắt người "***không cảm nhận***" được một tín hiệu, nếu nó ở bên cạnh một tín hiệu nhất định nào đó.

b) Phân nhóm phương pháp giấu tin theo "***định dạng ảnh***"**.**

1/ Nhóm phương pháp " phụ thuộc định dạng ảnh".

Hạn chế của nhóm phương pháp này là thông tin giấu dễ bị "tổn thương" bởi các phép biến đổi ảnh.

Trong nhóm này lại chia ra theo dạng ảnh, có các phương pháp cho: ảnh dựa vào bảng mầu, ảnh JPEG.

2/ Nhóm phương pháp "***độc lập với định dạng ảnh***".

Đặc trưng của nhóm phương pháp này là lợi dụng vào việc biến đổi ảnh để giấu tin vào trong đó, ví dụ giấu vào các hệ số biến đổi. Như vậy có bao nhiêu phép biến đổi ảnh thì cũng có thể có bấy nhiêu phương pháp giấu ảnh.

Một số phép "***biến đổi ảnh***":

+ Phương pháp biến đổi theo miền không gian (Spacial domain)

+ Phương pháp biến đổi theo miền tần số (DCT, DFT, Wavelet)

+ Phương pháp biến đổi hình học.

Phương pháp nhóm thứ hai có nhiều ưu điểm hơn về tính bền vững, tuy nhiên lượng thông tin giấu được sẽ ít hơn và cài đặt cũng sẽ phức tạp hơn.

c) Phân nhóm phương pháp giấu tin theo "*đặc điểm kỹ thuật*".

1/ Phương pháp thay thế.

+ Thay thế các bit dữ liệu trong bản đồ bit (bit plane)

+ Thay thế bảng mầu (palette)

2/ Phương pháp xử lý tín hiệu.

+ Các phương pháp biến đổi ảnh (Transform)

+ Các kỹ thuật điều chế dải phổ.

3/ Phương pháp mã hoá (coding)

+ Lượng hoá, dithering.

+ Mã hoá sửa lỗi.

4/Phương pháp thống kê - kiểm thử giả thuyết.

5/ Phương pháp sinh "mặt nạ" (Fractal)

5.2.3. Giấu tin sử dụng tính chẵn lẻ của tổng số bit 1

Đối với ảnh màu hay ảnh đa cấp xám, cũng có thể áp dụng thuật toán "Giấu tin sử dụng tính chẵn lẻ của tổng số bit 1" cho ảnh đen trắng (Xem 5.2.1)

Với các loại ảnh này, mỗi điểm ảnh được biểu diễn bằng nhiều bit, trong đó có những bit ít quan trọng (LSB: Least Significant bit) Từ mỗi điểm ảnh, ta chọn ra một bit LSB, lưu nó vào ma trận 2 chiều **F** gồm các bit 0, 1.

Trên ma trận **F** ta áp dụng thuật toán "Giấu tin sử dụng tính chẵn lẻ của tổng số bit 1" cho ảnh đen trắng.

5.2.4. Giấu tin vào các bít có trọng số thấp (LSB)

5.2.4.1. Cơ sở kỹ thuật

1/ Cách thể hiện mầu:

Khi chuyển ảnh tương tự sang ảnh số, người ta chọn 3 cách thể hiện mầu:

**** 24-bit mầu:***

Mỗi điểm ảnh có thể nhận một trong 2 ^ 24 mầu, mỗi mầu được tạo từ 3 mầu cơ bản: red (R), green (G), blue (B), mỗi mầu nhận một giá trị từ 0 đến 255 (8 bit)

**** 8-bit mầu***:

Mỗi điểm ảnh có thể nhận một trong 256 mầu, chọn từ một bảng mầu (Palette)

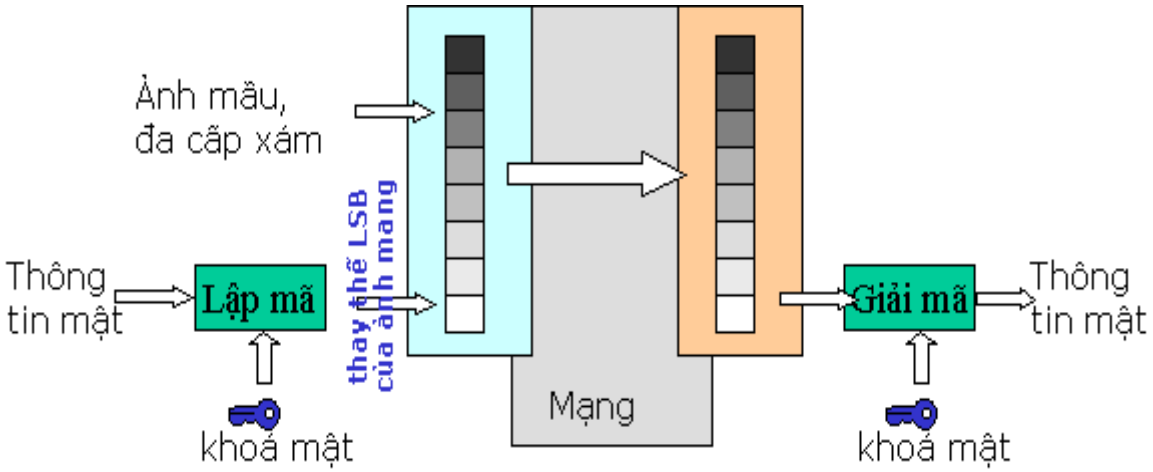
**** 8-bit dải xám***:

Mỗi điểm ảnh có thể nhận một trong 256 (2^8) sắc thái xám.

2/ Gài tin mật vào bít có trọng số thấp:

Phương pháp LSB sửa bít hay các bít có trọng số thấp nhất (ít quan trọng nhất để tạo nên mầu điểm ảnh), gài các thông tin mật vào đó. Các thông tin được giấu sẽ "lẫn" vào đâu đó giống như nhiễu ảnh.

Áp dụng kỹ thuật LSB, một điểm ảnh 24-bit có thể giấu được 3 bit thông tin (vì mỗi điểm được thể hiện bằng 3 byte) Mọi sự thay đổi trên điểm ảnh có trọng số thấp đều không gây nên sự chú ý của mắt người.



Hình : Giấu tin vào các bít ít quan trọng của điểm ảnh

5.2.4.2. Ví dụ về phương pháp LSB

Ảnh trước khi giấu có 3 điểm ảnh. Bit ít quan trọng là bit cuối cùng của mỗi byte.

00100111 11101001 11001000

00100111 11001000 11101001

11001000 00100111 11101001

Cần giấu chữ **A** (mã ASCII là 65, mã nhị phân là **0100 0001**)

Để giấu chữ **A** cần 3 điểm ảnh liên tiếp.

Chèn giá trị nhị phân của chữ **A** vào ba điểm ảnh trên, bắt đầu từ byte dưới cùng bên phải, chèn vào bit "***bít ít quan trọng***". Ảnh mang sau khi giấu chữ **A** là:

00100110 11101000 11001001

00100110 11001000 11101000

11001000 00100110 11101001

Các bit được gạch chân là các bit bị lật. Có thể dùng hai bit có trọng số thấp để giấu tin mà chất lượng không thay đổi nhiều đối với mắt thường.

Từ ví dụ trên ta có thể suy ra rằng nếu dùng 1 LSB, thì xác suất phải lật bít là 50%, vậy nên lượng nhiễu gây ra cho ảnh là rất ít.

Đối với ảnh mầu 24 bit, đôi khi người ta có thể dùng đến 2 hoặc thậm chí 3 bít thấp mà vẫn không để lộ thông tin mật.

Đối với ảnh 8 bit thì điều này là không thể, và người ta chỉ dùng 1 bít thấp nhất để giấu tin.

5.2.4.3. Dung lượng tin giấu

Phương pháp **LSB** giấu được nhiều thông tin.

Với ảnh 24 bit / điểm ảnh, dùng một bít có trọng số thấp có thể giấu được:

3 bít ẩn / 1 điểm ảnh (24 bít dữ liệu) = 1/8 bít ẩn / bít dữ liệu

Nếu dùng 2 bít có trọng số thấp

6 bít ẩn / 1 điểm ảnh (24 bít dữ liệu) = 1/4 bít ẩn / bít dữ liệu

Trong các ảnh sặc sỡ chúng ta có thể dùng thậm chí 3 bit LSB, khi đó thu được tỷ lệ bít ẩn / bit dữ liệu là 3/8.

Đôi khi người ta hỏi ngược lại là cần bao nhiêu byte ảnh để có thể giấu 1 byte tin mật. Nếu chỉ dùng 1 bit thấp ta cần 8 byte, nếu dùng đến 2 bit, ta chỉ cần 4 byte dữ liệu là đã giấu được 1 byte thông tin.

Nếu áp dụng kỹ thuật LSB lên ảnh 8-bit, cần phải chú ý hơn vì ảnh 8-bit không để chấp nhận thay đổi như ảnh 24-bit. Nên tránh dùng các ảnh vẽ phức tạp (như *Mona Lisa*) Các ảnh đơn giản như ảnh động vật, ví dụ chó, mèo phù hợp hơn. Khi sửa bit trọng số thấp trong ảnh 8-bit, các con trỏ chỉ đến bảng mầu cũng bị thay đổi theo. Chú ý rằng đôi khi chỉ cần thay đổi 1 bit có thể dẫn đến sự khác biệt về dải Red và dải Blue. Các thay đổi như vậy sẽ bị nhận ra ngay. Vì vậy các chuyên gia về giấu tin trong ảnh khuyên nên dùng bảng mầu xám vì sự khác biệt giữa các cấp mầu không dễ thấy.

5.2.4.4. Tính bền vững

Phương pháp LSB rất dễ bị "tổn thương" bởi một loạt các phép biến đổi ảnh, ngay cả phép biến đổi ảnh đơn giản và thông dụng nhất.

Nén ảnh mất dữ liệu như JPEG rất dễ dàng phá huỷ tin mật. Vấn đề là ở chỗ, những "lỗ hổng" trong hệ thống thị giác con người - ít nhậy cảm với các nhiễu bổ sung - mà phương pháp chèn bít LSB khai thác lại cũng chính là yếu tố mà phương pháp nén mất dữ liệu dựa lên đó để giảm mức dữ liệu của một ảnh.

Các phép biến đổi hình học như dịch chuyển hay xoay cũng dễ làm mất dữ liệu mật vì khi đó vị trí của các bít giấu sẽ bị thay đổi. Chỉ có một phép dịch chuyển đơn giản thì mới có thể phục hồi lại dữ liệu mật.

Các phép xử lý ảnh khác như làm mờ ảnh cũng sẽ làm mất dữ liệu hoàn toàn.

Tóm lại phương pháp LSB là phương pháp có tính ổn định kém nhất.

BÀI TẬP CHƯƠNG 5. ẨN GIẤU TIN.

Để hiểu cách thức Giấu tin và Tách tin đã giấu từ môi trường giấi tin với từng phương pháp cụ thể, bài tập chương 5 tập trung vào việc lập chương trình Giấu tin và Tách tin đã giấu.

Bài tập

Viết chương trình thực hiện Giấu tin sau:

1/ Thuật toán giấu tin sử dụng tính chẵn lẻ của tổng số bit 1.

2/ Thuật toán giấu tin M.Y.Wu - J.H.Lee

3/ Giấu tin vào các bít có trọng số thấp (LSB)

Mẫu Chương trình

* Mỗi chương trình giấu tin thực hiện các công việc theo thực đơn sau:

Thực đơn chính.

G. Giấu tin.

T. Tách tin.

K. Kết thúc.

Chương 6. BẢO TOÀN VÀ XÁC THỰC DỮ LIỆU

6.1. Bảo toàn dữ liệu

6.1.1. Tổng quan về Bảo toàn dữ liệu

6.1.1.1. Bài toán Bảo toàn dữ liệu hay Bảo đảm "toàn vẹn" dữ liệu

Người gửi G cần chuyển tài liệu **x** tới người nhận N trên mạng công khai.

N nhận được tài liệu **y** ghi rõ địa chỉ của G.

Bài toán 1: Bài toán "***bảo toàn dữ liệu*** " (bảo đảm toàn vẹn dữ liệu)

Kiểm tra để khẳng định được rằng tài liệu **y** (ghi địa chỉ của G), chính là **x**.

Yêu cầu quan trọng nhất của bài toán 1 là khẳng định được $\mathbf{y} \equiv \mathbf{x}$? Chưa quan tâm tới **y** là của G hay không ! Tức là chưa quan tâm tới nguồn gốc của **y**.

Câu hỏi: Hiện nay có các phương pháp nào để kiểm tra $\mathbf{y} \equiv \mathbf{x}$?

Điều đó có nghĩa là trên đường truyền tin, nếu có kẻ gian chặn bắt được **x**, nếu nó thay đổi nội dung của **x**, hay thay **x** bằng **y**, thì

1/ N có thể ***phát hiện được*** sự thay đổi nội dung của **x**.

2/ G có thể ***không cho phép*** kẻ gian thay đổi nội dung của **x**.

Bài toán 2: Bài toán "***xác thực nguồn gốc dữ liệu***" (chứng thực nguồn gốc dữ liệu)

Kiểm tra để khẳng định được rằng tài liệu **y** (ghi địa chỉ của G), đúng là của G.

Chưa quan tâm tới $\mathbf{y} \equiv \mathbf{x}$ hay không ! Các phương pháp kiểm tra ?

Câu hỏi: Hiện nay có các phương pháp nào để kiểm tra nguồn gốc của **y** ?

6.1.1.2. Phương pháp Bảo toàn dữ liệu

1) Dùng mã hóa hay giấu tin để bảo toàn dữ liệu.

Phương pháp mã hóa hay giấu tin dùng để "che giấu" tài liệu **x**, kẻ gian có chặn bắt được nó, thì cũng "khó" hiểu được nó, do đó "khó thể" thay đổi được nội dung của **x**. Như vậy phương pháp này thực hiện được khả năng 2/

(2 G có thể ***không cho phép*** kẻ gian thay đổi nội dung của **x**)

Chú ý rằng nếu tài liệu **x** được mã hóa, kẻ gian "khó" giải được mã, nếu tài liệu **x** được "giấu" trong tài liệu khác, kẻ gian "khó" tách ra được tài liệu gốc.

Nhưng nếu kẻ gian vẫn thay **x** bằng **y**, thì phương pháp mã hóa hay giấu tin không xác minh được ! Tức là phương pháp này không có khả năng 1)

(1/ N có thể ***phát hiện được*** sự thay đổi nội dung của **x**)

2) Dùng "chữ ký số" đề bảo toàn dữ liệu.

Người gửi G cần chuyển tài liệu **x** tới người nhận N trên mạng công khai.

Nếu dùng "chữ ký số" để bảo toàn **x**, thì G phải chuyển **x** và cả chữ ký trên **x** là **z** cho N. Như vậy N sẽ nhận được cặp tin (***tài liệu***, ***chữ ký***) = (**x**, **z**), **z** = Sig (**x**)

Như vậy phương pháp này chỉ thực hiện được khả năng 1/

(1/ N có thể ***phát hiện được*** sự thay đổi nội dung của **x**)

Nếu kẻ gian thay đổi nội dung của **x**, hay dùng **y** thay cho **x**, thì khi Nkiểm tra chữ ký của G, chắc chắn chữ ký **z** là sai, vì **x** cũ đã bị thay đổi.

Chú ý rằng phương pháp này không thực hiện được khả năng 2/

Tức là kẻ gian có thể thay được đổi nội dung của **x**.

(2/ G có thể ***không cho phép*** kẻ gian thay đổi nội dung của **x**)

3) Dùng "thủy vân ký" đề bảo toàn dữ liệu.

Người gửi G cần chuyển tài liệu **x** tới người nhận N trên mạng công khai.

Nếu dùng "thủy vân ký" để bảo toàn **x**, thì G phải cho "ẩn giấu" vào **x** một "dấu hiệu đặc trưng" **C** của mình, như vậy **x** đã trở thành **x**' (vật mang tin **C**) Sau đó

G chuyển **x**' cho N trên mạng.

N tách **C** ra khỏi **x**' và nhận được tài liệu gốc là **x**.

Như vậy phương pháp này chỉ thực hiện được khả năng 1/

(1/ N có thể ***phát hiện được*** sự thay đổi nội dung của **x**)

Lý do:

Kẻ gian không biết được tài liệu gốc là **x**, vì nó chỉ chặn bắt được **x**'.

Nó tìm cách tách một dấu hiệu khả nghi **C**' từ **x**'.

Khi đó tài liệu mà N nhận được không phải là **x**', mà là **x**".

N tách **C** ra khỏi **x**" , sẽ không nhận được "dấu hiệu" **C**.

Tức là N có thể phát hiện được sự thay đổi nội dung của **x**.

Chú ý rằng phương pháp này không thực hiện được khả năng 2/ (như PP Ký)

Tức là kẻ gian có thể thay được đổi nội dung của **x**.

(2/ G có thể ***không cho phép*** kẻ gian thay đổi nội dung của **x**)

4) Dùng "hàm băm" đề bảo toàn dữ liệu.

* ***Đặc điểm của hàm băm***:

Hàm băm **h** là hàm một chiều (One-way Hash) với các đặc tính sau:

1) Với tài liệu đầu vào (bản tin gốc) **x**, chỉ thu được giá trị băm duy nhất **z** = **h**(**x**)

2) Nếu dữ liệu trong bản tin **x** bị thay đổi hay bị thay hoàn toàn để thành bản tin **x**', thì giá trị băm **h**(**x**') ≠ **h**(**x**)

Cho dù chỉ là một sự thay đổi nhỏ, ví dụ chỉ thay đổi 1 bit dữ liệu của bản tin gốc **x**, thì giá trị băm **h**(**x**) của nó cũng vẫn thay đổi. Điều này có nghĩa là: hai thông điệp khác nhau, thì giá trị băm của chúng cũng khác nhau.

Dựa vào đặc điểm trên của hàm băm, người ta bảo toàn dữ liệu như sau.

**** Bảo toàn dữ liệu dùng "hàm băm"***:

Người gửi G cần chuyển tài liệu **x** tới người nhận N trên mạng công khai.

Nếu dùng "hàm băm" để bảo toàn **x**, thì G phải chuyển **x** và cả giá trị băm trên **x** là **z** cho N. Như vậy N sẽ nhận được cặp tin (***tài liệu***, ***đại diện TL***) = (**x**, **z**), **z** = **h**(**x**)

(Chú ý **z** là giá trị băm trên **x**, còn được gọi là ***đại diện tài liệu***)

N sẽ băm lại **x**, và nhận được giá trị băm là **z**'. Nếu **z**' ≠ **z**, thì chắc chắn **x** đã bị thay đổi trên đường truyền tin. Nếu **z**' ≡ **z**, thì **x** được bảo toàn.

Như vậy phương pháp này chỉ thực hiện được khả năng 1/

Tức là N có thể phát hiện được sự thay đổi nội dung của **x**.

(1/ N có thể ***phát hiện được*** sự thay đổi nội dung của **x**)

Chú ý rằng phương pháp này không thực hiện được khả năng 2/(như PP Ký)

Tức là kẻ gian có thể thay được đổi nội dung của **x**.

(2/ G có thể ***không cho phép*** kẻ gian thay đổi nội dung của **x**)

6.1.2. Bảo toàn dữ liệu bằng kết hợp các phương pháp

6.1.2.1. Kết hợp Mã hóa hay Giấu tin với ký số hay thủy vân ký.

Trong mục trước ta đã biết rằng mỗi công cụ có mặt mạnh và mặt yếu.

Kết hợp các công cụ lại, sẽ nhận được các mặt mạnh của chúng, cụ thể như sau:

+ Phương pháp mã hóa hay giấu tin chỉ thực hiện được khả năng 2/ của bảo toàn dữ liệu **x**, tức là không cho phép kẻ gian thay đổi nội dung của **x** !

Nhưng không thực hiện được khả năng 1/, tức là người nhận N không phát hiện được sự thay đổi nội dung của **x**.

+ Ngược lại, phương pháp ký số hay thủy vân ký chỉ thực hiện được khả năng 1/ của bảo toàn dữ liệu **x**, tức là có thể phát hiện được sự thay đổi nội dung của **x** !

Nhưng không thực hiện được khả năng 2/, tức là kẻ gian vẫn có thể thay đổi được nội dung của **x**.

Kết hợp hai phương pháp trên, sẽ có được phương pháp thực hiện được cả khả năng 1/ và khả năng 2/ của việc bảo toàn dữ liệu **x**.

(1/ N có thể ***phát hiện được*** sự thay đổi nội dung của **x**)

(2/ G có thể ***không cho phép*** kẻ gian thay đổi nội dung của **x**)

**** Kết hợp Mã hóa và Ký số:***

Người gửi G cần chuyển tài liệu **x** tới người nhận N trên mạng công khai.

Nếu dùng "chữ ký số" để bảo toàn **x**, thì G phải chuyển **x** và cả chữ ký trên **x** là **z** cho N. Trước khi gửi cặp tin **T**= (***tài liệu***, ***chữ ký***) = (**x**, **z**), (**z** = Sig (**x**)), G mã hóa **T**, sau đó mới gửi cho N.

Tương tự ta kết hợp Giấu tin với ký số, Mã hóa và với thủy vân ký, …

6.1.2.2. Kết hợp Mã hóa hay Giấu tin với Hàm băm.

+ Phương pháp mã hóa hay giấu tin chỉ thực hiện được khả năng 2/ của bảo toàn dữ liệu **x**, tức là không cho phép kẻ gian thay đổi nội dung của **x** !

Nhưng không thực hiện được khả năng 1/, tức là người nhận N không phát hiện được sự thay đổi nội dung của **x**.

+ Ngược lại, phương pháp hàm băm chỉ thực hiện được khả năng 1/ của bảo toàn dữ liệu **x**, tức là có thể phát hiện được sự thay đổi nội dung của **x** !

Nhưng không thực hiện được khả năng 2/, tức là kẻ gian vẫn có thể thay đổi được nội dung của **x**.

Kết hợp hai phương pháp trên, sẽ có được phương pháp thực hiện được cả khả năng 1/ và khả năng 2/ của việc bảo toàn dữ liệu **x**.

**** Kết hợp Mã hóa và hàm băm:***

Người gửi G cần chuyển tài liệu **x** tới người nhận N trên mạng công khai.

Nếu dùng "hàm băm" để bảo toàn **x**, thì G phải chuyển **x** và cả giá trị băm của **x** là **z** cho N. Trước khi gửi cặp tin **T**= (***tài liệu***, ***đại diện TL***) = (**x**, **z**), **z** = **h**(**x**), G mã hóa **T**, sau đó mới gửi cho N.

Tương tự ta kết hợp Giấu tin với hàm băm.

6.2. Bảo đảm xác thực

6.2.1. Phân loại xác thực điện tử

Hiện nay có một số cách phân loại "xác thực" như sau:

Cách 1: **Phân loại theo đối tượng cần xác thực**.

Có 2 loại đối tượng chính: Dữ liệu và Thực thể.

1) Xác thực Dữ liệu: Văn bản, hình ảnh, âm thanh,..

+ Xác thực Thông điệp (Message Authentication)

+ Xác thực Giao dịch (Transaction Authentication)

+ Xác thực Khóa (Key Authentication)

2) Xác thực Thực thể: Người dùng, thiết bị đầu cuối,..

+ Xác thực Thực thể (Entity Authentication)

Cách 2: **Phân loại theo công việc cần xác thực**.

1) Xác thực Thông điệp (Message Authentication)

2) Xác thực Giao dịch (Transaction Authentication)

3) Xác thực Thực thể (Entity Authentication)

4) Xác thực Khóa (Key Authentication)

Cách 3: **Phân loại theo đặc điểm xác thực**.

1) Xác thực bảo đảm định danh nguồn gốc **(**Identification of Source)

2) Xác thực bảo đảm toàn vẹn dữ liệu (Data Integrity)

3) Xác thực bảo đảm tính duy nhất (Uniqueness)

4) Xác thực bảo đảm tính phù hợp về thời gian (Timeliness)

Tổng kết các loại xác thực

Type of Authentication	*Identification of Source*	*Data Integrity*	*Timeliness or Uniqueness*
Message authentication	yes	yes	-
Transaction authentication	yes	yes	yes
Entity authentication	yes	-	yes
Key authentication	yes	yes	-

Properties of Various Types of Authentication

6.2.2. Xác thực dữ liệu (Data Authentication)

6.2.2. 1. Xác thực thông điệp (Message Authentication)

Xác thực thông điệp hay ***Xác thực tính nguyên bản*** của dữ liệu

(Data Origin Authentication) là một kiểu ***xác thực đảm bảo*** **một thực thể** được chứng thực là **nguồn gốc thực sự** tạo ra dữ liệu này ở một thời điểm nào đó.

Xác thực thông điệp bao hàm cả **tính toàn vẹn dữ liệu**, nhưng ***không đảm bảo*** tính duy nhất và phù hợp về thời gian của nó.

6.2.2. 2. Xác thực giao dịch (Transaction Authentication)

Xác thực giao dịch là ***Xác thực thông điệp*** cộng thêm việc **đảm bảo tính duy nhất** (Uniqueness) và phù hợp về **thời gian** (Timeliness) của nó.

Xác thực giao dịch liên quan đến việc sử dụng các tham số thời gian (**TVP** – Time Variant Parameters)

Transaction Authentication = Message Authentication + TVP

Xác thực giao dịch "mạnh hơn" Xác thực thông điệp.

Chú ý

Một thông điệp gửi đi có thể đã bị chặn và phát lại (tương tự như việc sử dụng lại nhiều lần một đồng tiền "số" (Double spending) Để ngăn chặn tình huống này, người gửi và người nhận có thể gắn vào thông điệp ***nhãn thời gian*** hay ***mã thông điệp***.

Mã thông điệp là con số được gắn vào thông điệp. Nó có thể chỉ dùng một lần duy nhất, giá trị không lặp lại, hoặc dùng một dãy số (Sequence Numbers)

Thám mã không thể biết được các bit của con số này nằm ở vị trí nào trong thông điệp, hay không thể biết cách thay đổi các bit để tạo ra dạng mã hoá của số tiếp theo, hoặc không thể biết cách thay đổi các bit này mà không làm gián đoạn việc giải mã phần còn lại của thông báo.

Các số thông báo này không thể bị thay thế, thay đổi hoặc giả mạo. Người nhận phải duy trì việc đếm các số thông báo đã nhận được. Nếu 2 người sử dụng một tập các số thì người nhận có thể ngay lập tức biết được liệu có thông báo nào trước thông báo hiện thời đã bị mất hoặc bị chậm trễ, vì số được mã hoá của thông báo hiện thời phải lớn hơn số được mã hoá của thông báo trước.

Nếu người gửi có nhiều thông báo thì có thể số thông báo sẽ quá dài. Vì thế, người ta thường đặt lại bộ đếm số thông báo khi nó đạt tới giá trị lớn nào đó. Lúc này tất cả các bên thu phải được thông báo rằng, số thông báo được gửi tiếp theo sẽ được đặt lại về một số nhỏ (chẳng hạn là 0)

Nhãn thời gian (TimeStamp) là các dấu hiệu về thời gian và ngày tháng lấy từ đồng hồ hệ thống hoặc đồng hồ địa phương. Bên gửi: gửi dữ liệu gắn TimeStamp đi. Bên nhận: nhận được dữ liệu, tiến hành lấy TimeStamp tại thời điểm hiện thời, trừ đi TimeStamp nhận được. Dữ liệu nhận được sẽ được chấp nhận nếu:

Độ lệch giữa 2 TimeStamp nằm trong khoảng chấp nhận được.

Không có thông báo nào có cùng TimeStamp được nhận trước đó từ cùng một người gửi. Điều này được thực hiện bằng cách bên nhận lưu giữ danh sách các TimeStamp từ người gửi để kiểm tra hoặc ghi lại TimeStamp gần nhất và chỉ chấp nhận TimeStamp có giá trị lớn hơn.

Như vậy, bên nhận phải đồng bộ và bảo mật về thời gian rất chặt chẽ với bên gửi, ngoài ra phải lưu trữ các TimeStamp.

6.2.2.3. Xác thực khoá (Key Authentication)

***+ Xác thực không tường minh khóa (Implicit Key Authentication)*:**

Một bên được đảm bảo (khẳng định) rằng chỉ có bên thứ hai (và có thể có thêm các bên tin cậy –Trusted Parties) là ***có thể*** truy cập được khoá mật.

***+ Khẳng định* (Xác nhận) *khóa (Key Confirmation)*:**

Một bên được đảm bảo (khẳng định) rằng bên thứ hai ***chắc chắn*** đã sở hữu khoá mật.

***+ Xác thực tường minh khóa (Explicit Key Authentication)*:**

Bao gồm cả 2 yếu tố trên.

Xác định được chắc chắn định danh của bên sở hữu khoá đã cho.

Chú ý

Xác thực khoá tập trung vào định danh bên thứ hai có thể truy cập khoá hơn là giá trị của khoá. Khẳng định khoá thì lại tập trung vào giá trị của khoá.

Ta gọi ngắn gọn Explicit Key authentication là Key authentication.

Chú ý

Xác thực dữ liệu đã bao gồm tính toàn vẹn dữ liệu. Ngược lại thì không. Tức là:

+ Đảm bảo xác thực nguồn gốc dữ liệu ➔ phải đảm bảo tính toàn vẹn dữ liệu.

+ Đảm bảo tính toàn vẹn dữ liệu // ➔ đảm bảo xác thực nguồn gốc dữ liệu.

6.2.2. 4. Xác thực nguồn gốc dữ liệu.

Dùng chữ ký số, hàm băm, chữ ký số, Thủy vân ký.

6.2.2. 5. Xác thực tính toàn vẹn của dữ liệu.

Dùng chữ ký số, hàm băm, chữ ký số, Thủy vân ký.

6.2.3. Xác thực thực thể (Entity Authentication)

6.2.3.1. Khái niệm Xác thực thực thể (Định danh thực thể)

Xác thực thực thể (hay Định danh thực thể) là xác thực định danh của một đối tượng tham gia giao thức truyền tin.

Thực thể hay đối tượng có thể là người dùng, thiết bị đầu cuối,...

Tức là: Một thực thể được xác thực bằng định danh của nó đối với thực thể thứ hai trong một giao thức, và bên thứ hai đã thực sự tham gia vào giao thức.

6.2.3.2. Các phương pháp Xác thực thực thể.

1) Xác thực dựa vào thực thể: Biết cái gì (Something Known):

Ví dụ như biết **mật khẩu** Password, định danh cá nhân (PIN), Giao thức định danh, để truy nhập vào hệ thống nào đó.

Định danh cá nhân (PIN - Personal Indentifier Number) thường gắn với

Something Possessed để tăng tính bảo mật.

Chú ý

"Biết cái gì" được dùng trong ***Giao thức định danh***, đó là ***cơ chế hỏi - đáp*** (Challenge-Response):

Một thực thể (Claimant) chứng tỏ định danh của nó đối với thực thể khác (Verifier) bằng cách biểu lộ hiểu biết về một thông tin mật liên quan nào đó cho Verifier, mà không bộc lộ bí mật của nó cho Verifier trong suốt giao thức.

Cơ chế đó gọi là "***Chứng minh không tiết lộ thông tin***".

2) Xác thực dựa vào thực thể: Sở hữu cái gì (Something Possessed):

Ví dụ như sở hữu **khóa bí mật** để ký điện tử.

Ví dụ như sở hữu Magnetic - striped Card, Credit Card, Smart Card,... có thể truy nhập được vào các hệ thống tự động.

3) Xác thực dựa vào thực thể: Thừa hưởng cái gì (Something Inherent):

Ví dụ như thừa hưởng chữ ký viết tay, **dấu vân tay**, giọng nói, mống mắt,..

Chú ý: **TVP** (Time Variant Parameter) có thể được sử dụng trong các giao thức định danh để đảm bảo Uniqueness và Timeliness.

Trong cơ chế hỏi - đáp thường dùng một người được uỷ quyền có tín nhiệm TA (Trusted Autherity) để tạo các tham số chung, các thuật toán ký, kiểm tra chữ ký và các chuỗi định danh, dấu xác nhận cho các bên tham gia,...

BÀI TẬP CHƯƠNG 6. BẢO ĐẢM XÁC THỰC VÀ TOÀN VẸN.

Bài tập

1) Viết chương trình thực hiện Xác thực:

1/ Xác thực dữ liệu bằng các phương pháp sau:

Mã hóa, Chữ ký, Hàm băm.

2/ Xác thực thực thể bằng các phương pháp sau:

Mật khẩu, Chữ ký, Giao thức định danh.

2) Viết chương trình thực hiện bảo đảm toàn vẹn dữ liệu:

Bảo đảm toàn vẹn dữ liệu bằng các phương pháp sau:

Mã hóa, Giấu tin, Chữ ký, Hàm băm và kết hợp giữa các phương pháp.

Chương 7. QUẢN LÝ KHÓA

7.1. Tổng quan về quản lý khoá

7.1.1. Vấn đề quản lý khóa bí mật

Với ***hệ mã hóa khóa đối xứng***, nếu biết được khóa mã hóa thì có thể "dễ" tính được khóa giải mã và ngược lại. Một số hệ mã hóa khóa đối xứng có khóa mã hóa và giải mã trùng nhau. Chính vì vậy, hệ mã hóa khóa đối xứng còn được gọi là hệ mã hóa "***khóa riêng***". Hai đối tác muốn liên lạc bí mật với nhau bằng hệ mã hóa khóa đối xứng, phải thỏa thuận trước 1 "***khóa riêng***", tức là 1 "***khóa bí mật***".

Một người giữ 1 bí mật đã khó, đằng này hai người cùng giữ 1 bí mật: càng khó giữ, bí mật rất dễ bị lộ. Chưa kể rằng nếu trên một mạng có **n** người dùng, thì theo phương pháp thông thường, mỗi người dùng phải quản lý (**n-1**) khóa, và tổng số khóa riêng giữa 2 người dùng nhiều nhất là (n-1) + (n-2) + (n-3) + … + 2 +1 = **n (n-1)/2**. Nếu **n** lớn thì giải pháp này không thực tế, vì lượng thông tin rất lớn cần phải truyền đi, khó bảo đảm an toàn.

Như vậy, điều cần quan tâm là cố gắng ***giảm được lượng tin cần truyền đi*** và ***cất giữ***, trong khi vẫn cho phép mỗi cặp người dùng U và V có chung khoá mật $\mathbf{K_{u,v}}$.

Ở đây xuất hiện nhu cầu "***quản lý khóa bí mật***" cho người dùng, bao gồm các việc:

"***Phân phối khóa mật*** ", "***Thỏa thuận khóa mật***", "***Bảo vệ khóa mật***".

+ "***Bảo vệ khóa mật*** " bằng "Mã hóa" khóa, "Băm" khóa, "Giấu" khóa, "Chia sẻ" khóa.

+ "***Phân phối khóa mật***" là ***cơ chế*** để một tổ chức ***chọn khóa mật***, sau đó truyền ***khóa mật***, hay chỉ truyền "***vật liệu công khai***" và "***cách thức***" tạo ***khóa mật*** đến cặp người dùng muốn có chung ***khóa mật***.

+ "***Thỏa thuận khóa mật***" là ***giao thức*** để cặp người dùng (hoặc nhiều hơn) liên kết với nhau ***cùng thiết lập khóa mật***, bằng cách liên lạc trên kênh công khai.

Cho đơn giản, từ nay về sau nói "***Phân phối khóa hay thỏa thuận khóa***", ta hiểu là "***Phân phối khóa hay thỏa thuận khóa***" "***bí mật***".

Mục tiêu của "phân phối khoá hay thoả thuận khoá" là tại thời điểm kết thúc thủ tục, cặp người dùng đều có khoá **K**, nhưng người dùng khác thì không biết được.

7.1.2. Vấn đề quản lý khóa công khai

1/ Hệ mã hóa khóa công khai:

Hệ mã hóa khóa công khai có ưu điểm hơn hệ mã hóa khóa riêng ở chỗ: có thể công khai thuật toán mã hoá và khóa mã hóa (khóa công khai) cho nhiều người sử dụng, "***khóa bí mật***" ("***khóa riêng***") chỉ do một người quản lý, cho nên không cần kênh an toàn để "**thống nhất**" khóa mật (bằng phân phối khóa hay thỏa thuận khóa) Tuy nhiên, hầu hết các hệ mã hóa khóa công khai đều chậm hơn hệ mã hóa khóa riêng (VD DES) Vì thế ***Hệ mã hóa khóa riêng*** được sử dụng để mã hóa các bản tin dài, ***Hệ mã hóa khóa công khai*** được dùng để **thống nhất** ***khóa riêng***.

Hơn thế nữa, ***Hệ mã hóa khóa công khai*** còn được dùng để tạo ra sơ đồ ký số hay các giao thức phục vụ bảo đảm an toàn thông tin.

Khác với hệ mã hóa khóa bí mật, với hệ mã hóa khóa công khai, hai đối tác truyền tin an toàn không phải "**thống nhất**" khóa mật, do đó không có nỗi lo chung để quản lý khóa mật (tất nhiên từng người phải lo bảo vệ khóa mật của mình), nhưng họ phải có nỗi lo chung để quản lý "***khóa công khai***".

2/ Tại sao phải quản lý tốt "***khóa công khai***" ?

Tại sao phải quản lý tốt "***khóa công khai***" ? Để hiểu rõ điều này ta lấy ví dụ.

Ví dụ 1:

+ Người dùng A có khóa bí mật **a**, bị lộ với người dùng B, như vậy B có khoá mật **a**.

+ Nếu đối tác của A vẫn dùng khóa công khai **b** (tương ứng với **a**) để mã hóa bản tin gửi cho A, thì B có thể xem được bản tin này (vì B đã có khóa mật **a** để giải mã) Thật là tai hại !

Trong trường hợp trên, người dùng A phải báo với các đối tác của mình rằng khóa mật **a** đã bị lộ, không dùng khóa công khai **b** để mã hóa nữa, vì người B có khoá **a** sẽ xem được các bản tin mật, đã mã hóa bởi khóa **b**. Người dùng A phải chọn cặp khóa (**a**, **b**) mới và công bố khóa công khai mới **b**.

Ví dụ 2:

+ Người dùng A có khóa bí mật **a**, bị lộ với người dùng B, như vậy B có khoá mật **a**.

+ Nếu A không thông báo với các đối tác của mình, thì B sẽ dùng **a** làm "***khóa ký***" ký lên các thông điệp giả mạo. Tuy nhiên nhờ khóa công khai **b** (tương ứng với **a**), các đối tác của A vẫn kiểm thử được rằng đó chính là chữ ký của A.

Thật tai hại cho A !

Trong trường hợp trên, người dùng A phải báo với các đối tác của mình rằng khóa mật **a** đã bị lộ, không dùng khóa công khai **b** để kiểm tra chữ ký của A nữa. Người dùng A cũng phải chọn cặp khóa (**a**, **b**) mới và công bố khóa công khai mới **b**.

3/ Ai lo quản lý "***khóa công khai***" ?

Vấn đề tiếp theo đặt ra là từng người dùng phải lo quản lý "***khóa công khai***" của riêng họ, hay có cơ quan chung để quản lý "***khóa công khai***" của mọi người dùng.

Câu trả lời: Người dùng phải lo quản lý khoá riêng (khoá mật) là điều tự nhiên. Nhưng từng người dùng không đủ sức để quản lý "***khóa công khai***" của mình, vì nó đã đượccông khai cho nhiều người biết. Trên thực tế có một cơ quan chuyên lo cung cấp và quản lý một "giấy chứng nhận" (có cơ sở pháp lý) để chứng thực "***khóa công khai***" nào đó hiện thời do ai sở hữu. "Giấy chứng nhận" này được gọi là "***chứng chỉ số***" hay "***chứng thư số***" (Digital Certificate)

Khi người dùng bị lộ khóa mật, họ phải báo cho cơ quan này biết, để xin được cấp "***chứng chỉ số***" cho "***khóa công khai***" mới. Mọi người dùng xem chứng chỉ số, sẽ biết được khóa công khai nào còn hiệu lực, nhờ đó tránh được các tình huống tương tự như hai ví dụ trên.

Hiện nay trên thế giới người ta theo xu hướng tổ chức các cơ quan như trên để cung cấp và quản lý các "***khóa công khai***" của người dùng. Tổ chức này được gọi là ***cơ quan chứng thực*** "***khóa công khai***" (**CA**: Certificate Authority)

7.2. Giao thức phân phối khoá

7.2.1. Phương pháp phân phối khóa.

Trong mục trên chúng ta đã biết ý nghĩa của việc "***Phân phối khóa bí mật***".

Đó là ***cơ chế*** để một tổ chức ***chọn khóa mật***, sau đó truyền nó đến cặp người dùng, hay chỉ truyền "***vật liệu công khai***" và "***cách thức***" tạo ***khóa mật*** đến cho họ.

Hơn thế nữa bảo đảm rằng thám mã khó thể ***khám phá*** hay ***tráo đổi khoá mật*** của họ.

Phương pháp thiết lập khóa chung này phải nhờ một Tổ chức tin cậy (TT) điều phối.

Vấn đề đặt ra là bằng cách nào để trung tâm được uỷ quyền (**TT**) có thể chuyển một cách an toàn ***khóa mật*** đến cặp người dùng **U** và **V** muốn có chung ***khóa mật*** $\mathbf{K_{u,v}}$ **?** hay chỉ chuyển "***vật liệu công khai***" và "***cách thức***" tạo ***khóa mật*** cho họ.

Mặt khác ***giảm được lượng thông tin cần truyền đi*** và ***cất giữ*** của mỗi cặp người dùng.

Hơn thế nữa bảo đảm rằng kẻ thám mã khó thể ***khám phá*** hay ***tráo đổi khoá mật*** của cặp người dùng. Hiện nay có hai phương pháp chính:

***+ Phương pháp thông thường*:**

Trung tâm được uỷ quyền (**TT**) chuyển từng ***khóa mật*** cho cặp người dùng **U**.

Phương pháp này phải dùng nhiều thông tin ***truyền đi*** và ***cất giữ***, mặt khác ***độ an toàn thấp*** khi truyền khóa trên mạng công khai. Mặt khác TT cũng biết được ***khóa mật*** !

***+ Phương pháp hiệu quả*:**

Trung tâm được uỷ quyền (**TT**) chỉ chuyển "***vật liệu công khai*** " và "***cách thức***" tạo ***khóa mật*** đến cặp người dùng **U** và **V**, trong khi mỗi người dùng vẫn giữ gìn "***vật liệu riêng*** " (bí mật) để thiết lập khóa.

Phương pháp này không phải dùng nhiều thông tin ***truyền đi*** và ***cất giữ***, mặt khác ***độ an toàn cao***, vì TT chỉ truyền trên mạng "***vật liệu công khai*** " và "***cách thức***" tạo ***khóa mật***, chứ không truyền trực tiếp khóa mật.

1) Phân phối khóa theo phương pháp thông thường.

Giả sử, có một mạng không an toàn gồm **n** người dùng, Trung tâm được uỷ quyền (TT) phân phối khóa riêng cho mỗi "cặp" người dùng.

Theo phương pháp thông thường, tổng số khóa riêng giữa 2 người dùng nhiều nhất là $(n-1) + (n-2) + (n-3) + \ldots + 2 + 1 = n(n-1)/2$

Như vậy mỗi người dùng phải lưu trữ (n-1) khóa. TT phải tạo ra n(n-1)/2 khóa và chuyển mỗi khoá cho duy nhất một cặp người dùng.

Phương pháp này chỉ nên sử dụng khi số người dùng không nhiều. Nếu **n** lớn thì giải pháp này không thực tế, vì lượng thông tin rất lớn cần phải truyền đi khó bảo đảm an toàn, mặt khác vì mỗi người dùng phải cất giữ nhiều khóa mật. Đó là các khoá mật của (n – 1) người dùng khác.

2) Phân phối khóa theo phương pháp hiệu quả.

Phương pháp phân phối khóa hiệu quả phải đạt được hai tiêu chí chính sau:

+ Bảo đảm an toàn các thông tin về ***khóa mật***:

Tức là bảo đảm rằng thám mã "khó" thể ***khám phá*** hay ***tráo đổi khoá mật***.

+ ***Giảm được lượng thông tin cần truyền đi*** và ***cất giữ***, trong khi vẫn cho phép mỗi cặp người dùng tính toán được khoá mật.

Hiện nay có nhiều phương pháp phân phối khóa hiệu quả, trung tâm được uỷ quyền (**TT**) chỉ chuyển "***vật liệu công khai***" và "***cách thức***" tạo ***khóa mật*** đến cặp người dùng.

Mỗi người dùng ***tự tính*** khóa chung của họ.

Thám mã có trộm được tin trên đường truyền, cũng khó tính được ***khóa mật*** vì không biết "***vật liệu bí mật***" của từng người dùng.

Sau đây sẽ giới thiệu một số phương pháp phân phối khóa hiệu quả:

Sơ đồ phân phối khoá Blom, Diffie-Hellman, Kerboros, ...

7.2.2. Giao thức phân phối khoá Blom.

7.2.2.1. Ý tưởng chính

Giả thiết có một mạng gồm **n** người dùng.

Giả sử rằng các khoá được chọn trên trường hữu hạn $\mathbf{Z_p}$ (số nguyên tố $\mathbf{p} \geq \mathbf{n}$)

Chọn số nguyên **k**, ($1 < \mathbf{k} < n-2$), giá trị **k** để hạn chế kích thước lớn nhất, mà sơ đồ vẫn duy trì được độ mật.

Trung tâm được uỷ quyền (**TT**) phải thiết kế một ***sơ đồ phân phối khóa*** để thực hiện được các yêu cầu sau:

+ Truyền đi (**k**+1) phần tử của $\mathbf{Z_p}$, cho mỗi người dùng trên kênh an toàn.

(Theo phương pháp phân phối thông thường, TT phải truyền đi **n**-1 phần tử)

+ Mỗi cặp người dùng U và V phải có khả năng tính được khoá chung $\mathbf{K_{u,v}} = \mathbf{K_{v,u}}$.

+ Bảo đảm điều kiện an toàn sau: tập bất kỳ gồm ***nhiều nhất*** **k** người dùng không liên kết với U hay V, thì "***khó***" thể xác định được bất kì thông tin nào về $\mathbf{K_{u,v}}$.

7.2.2.2. Giao thức phân phối khóa Blom với k=1.

Sơ đồ

1/ TT chọn số nguyên tố **p** công khai (**p** ≥ **n**), mỗi người dùng U chọn phần tử $\mathbf{r_u} \in \mathbf{Z_p}$ công khai, khác nhau.

2/ TT chọn 3 phần tử ngẫu nhiên bí mật **a**, **b**, **c** $\in \mathbf{Z_p}$ (không cần khác biệt) và chọn đa thức **f**(x, y) = (**a** + **b** * (x + y) + **c** * x * y) mod p

(Ở đây k = 1, nên chọn đa thức có lũy thừa bậc 1 (tuyến tính) đối với x và y)

3/ Với mỗi người dùng U, TT tính đa thức: $\mathbf{g_u}(x) = \mathbf{f}(x, \mathbf{r_u}) \bmod p$ và truyền $\mathbf{g_u}(x)$ đến U trên kênh an toàn (bí mật)

$\mathbf{g_u}(x) = f(x, r_u) \bmod p = (a + b.(x + r_u) + c.x.\ r_u \bmod p) \bmod p$, hay

$\mathbf{g_u}(x) = a_u + b_u * x$, với $a_u = a + b * r_u \bmod p$, $b_u = b + c * r_u \bmod p$

4/ Nếu U và V muốn liên lạc với nhau, mỗi người hãy tự tính khoá chung:

U tính $\mathbf{K_{u,v}} = \mathbf{g_u}(\mathbf{r_v}) = f(\mathbf{r_v}, r_u) = (a + b*(\mathbf{r_v} + r_u) + c.\ \mathbf{r_v}\ .\ r_u) \bmod p$.

V tính $\mathbf{K_{v,u}} = \mathbf{g_v}(r_u) = f(r_u, \mathbf{r_v}) = (a + b*(r_u + \mathbf{r_v}) + c.\ r_u\ .\ r_v) \bmod p$.

Do tính chất đối xứng của đa thức f(x, y), nên $\mathbf{K_{u,v}} = \mathbf{K_{v,u}}$

Ví dụ 1:

1/ Giả sử có 3 người dùng là U,V và W. Chọn công khai số nguyên tố p =17.

Các phần tử công khai của họ tương ứng là $r_u = 12$, $r_v = 7$, $r_w = 1$.

2/ TT chọn ngẫu nhiên, bí mật **a** = 8, **b** = 7, **c** = 2, chọn đa thức f là

f(x, y) = (**8** + **7** * (x + y) + **2** * x * y) mod 17.

3/ TT tính các đa thức và gửi cho U, V, W tương ứng là:

$\mathbf{g_u}(x) = f(x, 12) = (\mathbf{8} + \mathbf{7}*(x + 12) + \mathbf{2}*x*12) \bmod 17 = 7 + 14*x$.

$\mathbf{g_v}(x) = f(x, 7) = (\mathbf{8} + \mathbf{7}*(x + 7) + \mathbf{2}*x*7) \bmod 17 = 6 + 4*x$.

$\mathbf{g_w}(x) = f(x, 1) = (\mathbf{8} + \mathbf{7}*(x + 1) + \mathbf{2}*x*1) \bmod 17 = 15 + 9*x$.

4/ Khi U và V muốn liên lạc với nhau, mỗi người tự tính khoá chung:

U tính $\mathbf{K_{u,v}} = \mathbf{g_u}(\mathbf{r_v}) = f(\mathbf{r_v}, r_u) = (a + b*(\mathbf{r_v} + r_u) + c.\ \mathbf{r_v}\ .\ r_u) \bmod p$.

$= 7 + 14*\mathbf{7} \bmod 17 = \mathbf{3}$.

V tính $\mathbf{K_{v,u}} = \mathbf{g_v}(r_u) = f(r_u, \mathbf{r_v}) = (a + b*(r_u + \mathbf{r_v}) + c.\ r_u\ .\ r_v) \bmod p$.

= 6 + 4* **12** mod 17 =**3**.

* 3 khoá tương ứng với 3 cặp người dùng là:

$\mathbf{K_{u,v}} = f(r_u, r_v) = (8 + 7*(12 + 7) + 2*12*7) \bmod 17 = \mathbf{3}$.

$\mathbf{K_{u,w}} = f(r_u, r_w) = (8 + 7*(12 + 1) + 2*12*1) \bmod 17 = \mathbf{4}$.

$\mathbf{K_{v,w}} = f(r_v, r_w) = (8 + 7*(7 + 1) + 2* 7*1) \bmod 17 = \mathbf{10}$.

Ví dụ 2:

1/ Giả sử có 3 người dùng là U,V và W. Chọn công khai số nguyên tố p =83.

Các phần tử công khai của họ tương ứng là $r_u = 42$, $r_v = 31$, $r_w = 53$.

2/ TT chọn ngẫu nhiên, bí mật **a** = 10, **b** = 20, **c** = 30, chọn đa thức f là

$f(x, y) = (\mathbf{10} + \mathbf{20} * (x + y) + \mathbf{30} * x * y) \bmod 17$.

3/ TT tính các đa thức và gửi cho U, V, W tương ứng là:

$g_u(x) = f(x, 12) = (10 + 20*(x + 42) + 30*x* 42) \bmod 83 = 20 + 35*x$.

$g_v(x) = f(x, 7) = (10 + 20*(x + 31) + 30*x* 31) \bmod 83 = 49 + 37*x$.

$g_w(x) = f(x, 1) = (10 + 20*(x + 53) + 30*x* 53) \bmod 83 = 74 + 33*x$.

4/ Nếu U và V muốn liên lạc với nhau, mỗi người tự tính khoá chung:

U tính $\mathbf{K_{u,v}} = \mathbf{g_u}(\mathbf{r_v}) = f(\mathbf{r_v}, r_u) = (a + b*(\mathbf{r_v} + r_u) + c.\ \mathbf{r_v} . r_u) \bmod p$.

$= 20 + 35 * \mathbf{31} \bmod 83 = \mathbf{26}$.

V tính $\mathbf{K_{v,u}} = \mathbf{g_v}(r_u) = f(r_u, \mathbf{r_v}) = (a + b*(r_u + \mathbf{r_v}) + c.\ r_u . r_v) \bmod p$.

$= 49 + 37* \mathbf{42} \bmod 83 = \mathbf{26}$.

* 3 khoá tương ứng với 3 cặp người dùng là:

$\mathbf{K_{u,v}} = f(r_u, r_v) = (10 + 20* (42 + 31) + 30*42*31) \bmod 83 = 26$.

$\mathbf{K_{u,w}} = f(r_u, r_w) = (10 + 20* (42 + 53) + 30*42*53) \bmod 83 = 49$.

$\mathbf{K_{v,w}} = f(r_v, r_w) = (10 + 20* (31 + 53) + 30*31*53) \bmod 83 = 18$.

<u>Mức an toàn</u>

a) Sơ đồ Blom với k=1 an toàn với 1 đối thủ.

<u>Định lý</u>

Theo sơ đồ phân phối khóa Blom với k =1, ***khoá chung*** của ***cặp đối tác*** là ***an toàn*** không điều kiện trước bất kỳ người dùng thứ ba. (Không người dùng nào có thể xác định được thông tin về khoá của 2 người dùng khác)

Chứng minh

+ Giả sử người dùng thứ ba là W muốn thử tính khoá chung của U, V:

$\mathbf{K_{u,v}} = (a + b * (r_u + r_v) + c * r_u * r_v) \bmod p$

Trong đó các giá trị r_u, r_v là công khai, còn a, b, c là bí mật.

W biết được các giá trị sau do TT gửi đến:

$\mathbf{g_w}(x) = a_w + b_w * x$, với $a_w = a + b * r_w \bmod p$, $b_w = b + c * r_w \bmod p$

+ Ta chỉ ra rằng thông tin mà W biết, phù hợp với ***giá trị tùy ý*** $\mathbf{t} \in \mathbf{Z_p}$ của khoá $\mathbf{K_{u,v}}$.

Xét phương trình ma trận sau: (Chú ý Các phép số học được thực hiện trong $\mathbf{Z_p}$)

$$\begin{matrix} 1 & r_u + r_v & r_u r_v \\ 1 & r_w & 0 \\ 0 & 1 & r_w \end{matrix} \begin{matrix} a \\ b \\ c \end{matrix} = \begin{matrix} \mathbf{t} \\ a_w \\ b_w \end{matrix}$$

Tức là hệ các phương trình:

$\mathbf{K_{u,v}} = (a + b * (r_u + r_v) + c * r_u * r_v) \bmod p = \mathbf{t}$ (1)

$a + b * r_w \bmod p = a_w$ (2)

$b + c * r_w \bmod p = b_w$ (3)

(1) thể hiện giả thiết rằng $\mathbf{K_{u,v}} = \mathbf{t}$. (2), (3) cho thấy W biết a, b, c từ $\mathbf{g_w}(x)$

Định thức của ma trận hệ số là: $\{(1*r_w * r_w) + (1 * 1 * r_u r_v) + (0 *(r_u + r_v) * 0)\} -$

$$\{(0* r_w * r_u r_v) + (1*1*0) + (1 * (r_u + r_v) * r_w)\} =$$

$$= \{r_w^2 + r_u r_v\} - \{(r_u + r_v) * r_w\} = (r_w - r_u)(r_w - r_v)$$

Vì $r_w \neq r_u$ và $r_w \neq r_v$ nên định thức ma trận hệ số khác không (Det $\neq 0$) Do đó phương trình ma trận có ***nghiệm duy nhất*** cho a, b, c.

Nói cách khác, bất kì giá trị $\mathbf{t} \in \mathbf{Z_p}$ cũng có thể nhận là khoá $\mathbf{K_{u,v}}$.

b) Sơ đồ Blom với k=1 không an toàn với liên minh 2 đối thủ.

Định lý

Liên minh 2 người dùng {W, X} (không phải là cặp người dùng {U, V})có khả năng xác định khoá mật $\mathbf{K_{u,v}}$ bất kỳ của U và V.

Chứng minh

Hai người W và X cùng biết các đẳng thức sau:

$a_w = a + b*r_w$. $b_w = b + c*r_w$.

$a_x = a + b*r_x$. $b_x = b + c*r_x$.

Như vậy, họ có 4 phương trình với 3 ẩn a, b, c chưa biết, họ dễ dàng tính ra nghiệm duy nhất a, b, c. Từ đó, họ có thể thiết lập đa thức f(x, y) và tính khoá chung bất kỳ của cặp người dùng nào đó.

7.2.2.3. Giao thức phân phối khóa Blom với k > 1.

<u>Sơ đồ</u>

Để tạo lập sơ đồ phân phối khoá chống lại được liên minh **k** đối thủ, TT dùng đa thức f(x, y) dạng sau:

$$\mathbf{f}(x, y) = \sum_{i=0}^{k} \sum_{j=o}^{k} a_{i,j}\, x^i y^j \mod p.$$

Trong đó $\mathbf{a_{i,j}} \in \mathbf{Z_p}$ $(0 \le i \le \mathbf{k}, 0 \le j \le \mathbf{k})$ và $a_{i,j} = a_{j,i}$ với mọi i, j .

Các phần còn lại của giao thức như với sơ đồ phân phối khoá Blom với k=1.

7.2.3. Giao thức phân phối khoá Diffie-Hellman.

<u>Sơ đồ</u>

1/ Chọn số nguyên tố **p** sao cho bài toán logarit rời rạc trong $\mathbf{Z_p}$ là "khó" giải.

Chọn α là phần tử nguyên thuỷ của $\mathbf{Z_p}^*$.

Giá trị **p** và α là công khai (Người dùng hoặc TT chọn)

Mỗi người dùng U chọn số mũ bí mật $\mathbf{a_u}$ $(0 \le \mathbf{a_u} \le p - 2)$ và tính giá trị công khai t-ương ứng: $\mathbf{b_u} = \alpha^{a_u} \mod p$.

Mỗi người dùng **U** có dấu xác nhận của TT về ID(U) và $\mathbf{b_u}$:

$\mathbf{C}(\mathbf{U}) = (ID(U), \mathbf{b_u}, sig_{TT}(ID(U), \mathbf{b_u}))$.

2/ Để có khoá chung với V, người dùng U (có $\mathbf{a_u}$) tính:

$$\mathbf{K_{u,v}} = b_v^{a_u} \mod p = \alpha^{a_u a_v} \mod p$$

3/ Để có khoá chung với U, người dùng V (có $\mathbf{a_v}$) tính:

$$\mathbf{K_{v,u}} = b_u^{a_v} \mod p = \alpha^{a_u a_v} \mod p$$

Rõ ràng 2 khoá là như nhau và bằng $\alpha^{a_u a_v} \mod p$

<u>Chú ý</u>

- ID(U) là thông tin định danh của người dùng U trên mạng, như tên gọi, địa chỉ hòm thư điện tử, số điện thoại,…

- Trung tâm được uỷ quyền phân phối khoá (TT) có sơ đồ chữ ký với thuật toán ký sig_{TT} (với khóa bí mật) và thuật toán xác minh ver_{TT} (với khóa công khai)

Giả thiết rằng mỗi thông tin đều được thu gọn thành đại diện và ký trên đại diện đó.

- Sơ đồ này "an toàn" về mặt tính toán, vì nó liên quan đến bài toán logarit rời rạc "***khó*** " giải. Cụ thể là "***khó***" tính được $\mathbf{a_u}$ từ phần tử công khai $\mathbf{b_u} = \alpha^{a_u} \bmod p$.

Ví dụ

1/ Chọn số nguyên tố **p** = 25307, sao cho bài toán logarit rời rạc trong $\mathbf{Z_p}$ là "khó" giải. Chọn phần tử nguyên thuỷ $\alpha = \mathbf{2} \in \mathbf{Z_p}^*$.

Giá trị **p** và α là công khai (Người dùng hoặc TT chọn)

Người dùng U chọn số mũ bí mật $\mathbf{a_u}$ = 3578 ($0 \le \mathbf{a_u} \le p - 2$) và tính giá trị công khai tương ứng: $\mathbf{b_u} = \alpha^{a_u} \bmod p = \mathbf{2}^{\mathbf{3578}} \bmod 25307 = 6113$.

Người dùng V chọn số mũ bí mật $\mathbf{a_v}$ = 19956 ($0 \le \mathbf{a_v} \le p - 2$) và tính giá trị công khai tương ứng: $\mathbf{b_v} = \alpha^{a_v} \bmod p = \mathbf{2}^{\mathbf{19956}} \bmod 25307 = 7984$.

2/ U (có $\mathbf{a_u}$) tính khoá chung: $K_{u,v} = b_v^{a_u} \bmod p = 7984^{\mathbf{3578}} \bmod 25307 = 3694$.

3/ V (có $\mathbf{a_v}$) tính khoá chung: $K_{v,u} = b_u^{a_v} \bmod p = 6113^{\mathbf{19956}} \bmod 25307 = 3694$.

Hai giá trị khoá trên là bằng nhau.

Mức an toàn

1) Với loại tấn công chủ động, không cần lo lắng nhiều, vì

Người dùng U có dấu xác nhận **C**(**U**) của trung tâm được uỷ quyền TT, điều này ngăn chặn người dùng khác U có thể biến đổi thông tin nào đó trong dấu xác nhận.

$\mathbf{C}(\mathbf{U}) = (ID(U), \mathbf{b_u}, sig_{TT}(ID(U), \mathbf{b_u}))$.

2) Với loại tấn công thụ động, cũng không cần lo lắng nhiều, vì

Người dùng W (khác U,V) "***khó***" có thể tính được khoá chung $\mathbf{K_{u,v}}$ của U, V.

Cụ thể khi biết $\mathbf{b_u} = \alpha^{a_u} \bmod p$ và $\mathbf{b_v} = \alpha^{a_v} \bmod p$, thì cũng "***khó***" có thể tính được khóa chung của U và V là $\mathbf{K_{u,v}} = \alpha^{a_u a_v} \bmod p$ (1)

Muốn tính được (1), W phải tính được $\mathbf{a_u}$ từ $\mathbf{b_u}$ và $\mathbf{a_v}$ từ $\mathbf{b_v}$. Nhưng đó là các trường hợp riêng của bài toán Logarit rời rạc. Như vậy chỉ cần bài toán Logarit rời rạc là

"***khó***" giải thì **sơ** đồ phân phối khoá Diffie-Hellman sẽ "***an toàn***" trước kiểu tấn công loại này (Trong sơ đồ đã giả thiết điều đó)

Bài toán Diffie-Hellman: Đó là Vấn đề trên.

Cho trước số nguyên tố p, phần tử nguyên thuỷ $\alpha \in \mathbf{Z_p}^*$, phần tử $\beta, \gamma \in \mathbf{Z_p}^*$

Yêu cầu: Tính $\beta^{\log_\alpha \gamma} \bmod p \ (= \gamma^{\log_\alpha \beta} \bmod p)$?

Chú ý

- Giả định cho rằng thuật toán bất kỳ giải được bài toán Diffie-Hellman thì cũng có thể giải được bài toán logarith rời rạc. Hiện nay giả định này vẫn chưa được chứng minh.

- Điều này cũng tương tự như tình huống với RSA:

Giả định cho rằng việc phá RSA tương đương "đa thức" với bài toán phân tích số. Hiện nay giả định này cũng chưa được chứng minh.

Theo nhận xét trên, bài toán Diffie-Hellman không khó hơn bài toán logarit rời rạc. Mặc dù không thể nói chính xác bài toán này khó như thế nào, song ta có thể nói rằng độ an toàn của nó tương đương với độ an toàn của hệ mã hoá Elgamal.

7.2.4. Giao thức phân phối khoá "tươi" Kerboros.

7.2.4.1. Ý tưởng chính

Trong các phương pháp phân phối khoá đã trình bày, mỗi cặp người dùng phải tính một khoá cố định dùng chung ***trong thời gian dài***, như vậy dễ bị "***tổn thương***". Có cách phân phối khoá khác, trong đó khoá của phiên làm việc chỉ cần tạo ra mỗi khi cặp người dùng cần liên lạc với nhau. Khoá này gọi là khoá "**tươi**" hay khoá "**phiên**" (Session)

Với cách phân phối khoá như vậy, người dùng không cần phải lưu giữ các khoá chung khi muốn liên lạc với những người dùng khác. Nhưng lưu ý rằng mỗi người dùng đều phải "***chia sẻ***" bí mật của khoá với Trung tâm phân phối khóa (TT)

Kerboros là hệ thống dịch phân phối khoá "**phiên**" theo nghĩa trên.

Sơ đồ

Kerboros là hệ phân phối khoá dựa trên hệ mã hóa ***khoá riêng*** (khoá đối xứng) Mỗi người dùng U có định danh công khai ID(U), U và TT phải có chung khoá bí mật $\mathbf{K_u}$ cho Hệ mã hóa ***khoá riêng*** (vd DES) Mọi thông báo cần truyền được mã hoá theo chế độ xích khối (CBC)

Khi cặp người dùng (U, V) có yêu cầu khoá "phiên", TT tạo ngẫu nhiên khoá "phiên" mới **K**, ghi lại thời gian hệ thống **T** khi có yêu cầu và chỉ ra thời gian tồn tại của **K** là **L**. Nghĩa là khoá **K** chỉ có hiệu lực trong thời gian từ T đến T+**L**. Tất cả các thông tin này đều được mã hoá và truyền đến U và V.

1) U yêu cầu TT khoá "phiên" **K** để liên lạc với V.

2) TT chọn ngẫu nhiên khoá "phiên" **K**, thời gian hệ thống **T** và thời gian tồn tại **L**.

3) TT tính $\mathbf{m_1} = \mathbf{E}_{\mathbf{Ku}}(\mathbf{K}, ID(V), \mathbf{T}, \mathbf{L})$, $\mathbf{m_2} = \mathbf{E}_{\mathbf{Kv}}(\mathbf{K}, ID(U), \mathbf{T}, \mathbf{L})$, gửi tới U.

4) U dùng hàm giải mã $\mathbf{D}_{\mathbf{Ku}}$ để tìm ra **K**, **T**, **L**, ID(V) từ $\mathbf{m_1}$.

U tính $\mathbf{m_3} = \mathbf{E}_{\mathbf{K}}(ID(U), \mathbf{T})$, và gửi đến V cùng với $\mathbf{m_2}$ nhận được từ TT.

(Chú ý : U dùng ngay khóa **K** để mã hóa)

5) V dùng hàm giải mã $\mathbf{D}_{\mathbf{Kv}}$ để tìm ra **K**, **T**, **L**, ID(U) từ $\mathbf{m_2}$. (1)

V dùng ngay khóa **K** để giải mã $\mathbf{m_3}$, tìm ra ID(U), **T**. (2)

V kiểm tra 2 giá trị của **T** và 2 giá trị của ID(U) (từ (1), (2)) có bằng nhau không ?

Nếu đúng thì V chấp nhận **K** là "khóa phiên" với U.

V mã hóa (**T** + 1) thành $\mathbf{m_4} = \mathbf{E}_{\mathbf{K}}(\mathbf{T} + 1)$ và gửi đến U.

6) U dùng khóa **K** để giải mã $\mathbf{m_4}$.

Nếu kết quả giải mã là **T** + 1, thì U chấp nhận **K** là "khóa phiên" với V.

<u>Thực hiện giao thức</u>

1) và 2) U yêu cầu TT khoá "phiên" **K** để liên lạc với V. TT tạo ra **K**, T, **L**.

Thông tin truyền đi trong giao thức được minh hoạ như sau:

$m_1 = \mathbf{E}_{\mathbf{Ku}}(K, ID(V), T, L)$ $m_3 = \mathbf{E}_{\mathbf{K}}(ID(U), T)$

$m_2 = \mathbf{E}_{\mathbf{Kv}}(K, ID(U), T, L)$ $m_2 = \mathbf{E}_{\mathbf{Kv}}(K, ID(U), T, L)$

TT ----------→ U ----------→ V

$m_4 = \mathbf{E}_{\mathbf{K}}(T + 1)$

<----------

3) **K**, T, **L** và ID(V) được mã hoá bằng khoá $\mathbf{K}_{\mathbf{u}}$ để tạo ra $\mathbf{m_1}$.

(Chú ý : U và TT có khóa chung là $\mathbf{K}_{\mathbf{u}}$)

K, T, **L** và ID(U) được mã hoá bằng khoá $\mathbf{K}_{\mathbf{v}}$ để tạo ra $\mathbf{m_2}$.

(Chú ý : V và TT có khóa chung là $\mathbf{K}_v$)

Cả hai bức điện $\mathbf{m_1}$ và $\mathbf{m_2}$ đều được gửi đến U.

4) U dùng khoá $\mathbf{K_u}$ (của mình) giải mã $\mathbf{m_1}$, nhận được **K**, T, **L**, ID(V)

U xác minh thời gian hiện tại của khoá **K** có thuộc khoảng (T, T + **L)** ?

U kiểm tra khoá **K** (để liên lạc giữa U, V), bằng cách xác minh thông tin ID(V) ?

U làm trễ thời gian gửi $\mathbf{m_2}$ đến V.

U dùng khoá **K** để mã hoá T và ID(U) thành $\mathbf{m_3}$ và gửi nó tới V.

5) V dùng khoá $\mathbf{K_v}$ (của mình) giải mã $\mathbf{m_2}$, thu được **K**, T, **L**, ID(U), dùng **K** giải mã $\mathbf{m_3}$.

V xác minh T, ID(U) nhận được từ $\mathbf{m_2}$ và $\mathbf{m_3}$ có giống nhau không ?

Điều này đảm bảo cho V rằng khoá **K** được mã hoá trong $\mathbf{m_2}$. Và **K** cũng là khoá đã được U dùng để mã hoá T, ID(U) thành $\mathbf{m_3}$.

V dùng khoá **K** để mã hóa T + 1 thành $\mathbf{m_4}$ và gửi nó cho U.

6) U dùng **K** để giải mã $\mathbf{m_4}$ và xác minh xem kết quả có bằng T + 1 không ?

Điều này đảm bảo cho U rằng khoá **K** đã được truyền thành công đến V, vì **K** đã được dùng để tạo ra $\mathbf{m_4}$.

Mức an toàn

1) Chú ý các chức năng khác nhau của những thông báo trong giao thức:

+ $\mathbf{m_1}$, $\mathbf{m_2}$ dùng để đảm bảo an toàn trong việc truyền khoá phiên **K**.

+ $\mathbf{m_3}$, $\mathbf{m_4}$ dùng để ***khẳng định*** khoá **K**, nghĩa là cho phép U và V có thể ***thuyết phục với nhau*** rằng họ sở hữu cùng một khoá phiên **K**.

+ Thời gian T và thời hạn **L** để ngăn kẻ phá hoại khỏi "***lưu***" **khóa cũ**, nhằm tái sử dụng lại sau này, vì các khoá không được chấp nhận khi chúng quá hạn.

2) Mọi người dùng trong mạng đều phải có đồng hồ đồng bộ với nhau, vì cần có thời gian hiện tại để xác định khoá **K** còn hợp lệ không.

Chú ý rằng trong thực tế, khó có được sự đồng bộ hoàn hảo, nên phải cho phép có một khoảng thay đổi nào đó về thời gian.

7.3. Giao thức thoả thuận khoá

7.3.1. Phương pháp thỏa thuận khóa.

Nếu không muốn dùng dịch vụ phân phối khoá qua trung tâm được uỷ quyền TT, cặp người dùng phải tự thoả thuận (trao đổi) "***khóa bí mật***".

"***Thỏa thuận khóa mật***" là ***giao thức*** để cặp người dùng (hoặc nhiều hơn) liên kết với nhau ***cùng thiết lập khóa mật***, bằng cách liên lạc trên kênh công khai.

Phương pháp thiết lập khóa chung kiểu này không nhờ Tổ chức tin cậy TT điều phối, cặp người dùng tự "***Thỏa thuận khóa mật***".

Hiện nay có hai phương pháp chính để "***Thỏa thuận khóa mật***":

***+ Phương pháp thông thường*:**

Khi cặp người dùng đã thống nhất có một ***khóa bí mật* chung**, thì một trong hai người chọn khóa ngẫu nhiên **K**, sau đó truyền nó một cách an toàn đến người kia bằng phương pháp nào đó, ví dụ bằng hệ mã hóa khóa công khai hay phương pháp "giấu tin".

Phương pháp này phải dùng nhiều thông tin ***truyền đi*** và ***cất giữ***, mặt khác ***độ an toàn thấp*** vì phải truyền đi "***trọn vẹn***" một ***khóa*** trên mạng công khai.

***+ Phương pháp hiệu quả*:**

Phương pháp hiệu quả để thỏa thuận khóa phải đạt được hai tiêu chí chính sau:

+ Bảo đảm an toàn các thông tin về ***khóa mật***:

Tức là bảo đảm rằng thám mã khó thể ***khám phá*** hay ***tráo đổi khoá mật***.

+ ***Giảm được lượng thông tin cần truyền đi*** và ***cất giữ***, trong khi vẫn cho phép mỗi cặp người dùng tính toán được khoá mật.

Theo phương pháp hiệu quả, người dùng không truyền cho nhau trên mạng "***trọn vẹn***" một ***khóa* K**, mà chỉ truyền "***vật liệu công khai***" và "***cách thức***" tạo ***khóa* K** đến cặp người dùng **U** và **V**.

Phương pháp này không phải dùng nhiều thông tin ***truyền đi*** và ***cất giữ***, mặt khác ***độ an toàn cao***, vì người dùng chỉ truyền trên mạng "***vật liệu công khai***" và "***cách thức***" tạo ***khóa mật***, chứ không truyền trực tiếp khóa mật.

Thám mã có trộm được tin trên đường truyền, cũng khó tính được ***khóa mật*** vì không biết "***vật liệu bí mật***" của từng người dùng.

7.3.2. Giao thức thoả thuận khoá Diffie-Hellman.

Sơ đồ

Chuẩn bị:

Người dùng chọn số nguyên tố **p** rất lớn sao cho bài toán logarit rời rạc trong $\mathbf{Z_p}^*$ là "khó" giải; chọn α là phần tử nguyên thuỷ $\in \mathbf{Z_p}^*$. Phần tử **p**, α là công khai.

1) Người dùng U chọn $\mathbf{a_u}$ ngẫu nhiên, bí mật ($0 \le a_u \le p - 2$)

Tính $\mathbf{b_u} = \alpha^{a_u} \bmod p$ và gửi nó đến V.

2) Người dùng V chọn $\mathbf{a_v}$ ngẫu nhiên, bí mật ($0 \le a_v \le p - 2$)

Tính $\mathbf{b_v} = \alpha^{a_v} \bmod p$ và gửi nó đến U.

3) U tính khoá chung $\mathbf{K_{u,v}} = (\alpha^{a_v})^{a_u} \bmod p$.

4) V tính khoá chung $\mathbf{K_{v,u}} = (\alpha^{a_u})^{a_v} \bmod p$.

Hai giá trị khoá đó là bằng nhau !

Chú ý

1) Giao thức thoả thuận khoá DH tương tự như giao thức phân phối khoá Diffie-Hellman.

Sự khác nhau ở chỗ số mũ bí mật $\mathbf{a_u}$, $\mathbf{a_v}$ (của U, V), đều được chọn lại trước mỗi lần thực hiện giao thức này, thay vì cố định.

2) Người dùng U và V đều được đảm bảo "***khoá tươi***" , vì ***khoá phiên*** phụ thuộc vào cả hai số ngẫu nhiên bí mật $\mathbf{a_u}$ và $\mathbf{a_v}$.

3) Vì $\mathbf{b_u}$, $\mathbf{b_v}$ trên đường truyền không được bảo vệ bởi tổ chức tin cậy TT, nên "kẻ xâm nhập giữa cuộc" có thể lợi dụng "lỗ hổng" này để phá hoại U, V.

Mức an toàn

Thông tin trao đổi trong giao thức được mô tả như sau:

α^{a_u}

U ----------> **V**

<----------

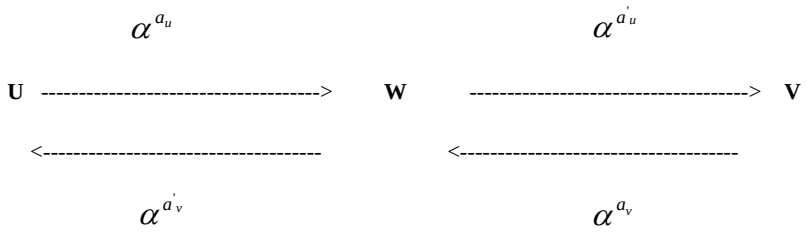
α^{a_v}

1) Hạn chế: Không có xác thực danh tính U và ***V.***

Giao thức này dễ bị tổn thương trước đối phương tích cực: "***kẻ xâm nhập giữa cuộc***".

Đó là tình tiết trong vở kịch "**The Lucy show**", trong đó Vivian Vance đang dùng bữa tối với người bạn, còn ***Lucille Ball*** đang trốn dưới bàn. Vivian và người bạn của cô đang cầm tay nhau dưới bàn. ***Lucille Ball*** cố tránh bị phát hiện, đã nắm lấy tay của cả hai người, còn hai người trên bàn vẫn nghĩ rằng họ đang cầm tay nhau.

Tấn công kiểu "***kẻ xâm nhập giữa cuộc***" trên giao thức trao đổi khoá Diffie-Hellman hoạt động cũng như vậy. ***Kẻ xâm nhập*** **W** sẽ chặn lại các bức điện trao đổi giữa U và V và thay thế bằng các bức điện của anh ta.



Cuối giao thức, U thiết lập thực sự khoá mật $\alpha^{a_u a'_v}$ với **W**, còn V thiết lập khoá mật $\alpha^{a'_u a_v}$ với **W**.

Khi U cố mã hóa bản tin để gửi cho V, và W có khả năng giải mã nó, nhưng V thì không thể giải mã được bản tin của U, vì anh ta không có khóa chung với U.

2) Cải tiến: Bổ sung xác thực danh tính U và ***V.***

Điều cơ bản với U, V là bảo đảm rằng, họ đang trao đổi khoá cho nhau mà không có **W**. Vì vậy trước khi trao đổi khoá, U và V phải thực hiện các giao thức tách bạch để thông báo ***danh tính*** của nhau, nhờ đó họ sẽ nhận ra kẻ không phải là U hay V.

7.3.3. Giao thức thoả thuận khoá "Trạm tới Trạm".

Giao thức thoả thuận khoá "Trạm tới Trạm" (**STS**) là cải tiến của giao thức phân phối khoá Diffie-Hellman, trong đó bổ sung phần xác thực danh tính của người dùng.

STS được gọi là giao thức ***thoả thuận khoá có xác thực***, nhờ trung tâm tin cậy **TT**.

<u>Sơ đồ</u>

Chuẩn bị:

Trung tâm tin cậy **TT** chọn số nguyên tố **p** rất lớn sao cho bài toán logarit rời rạc trong $\mathbf{Z_p}^*$ là "khó" giải. Chọn α là phần tử nguyên thuỷ trong $\mathbf{Z_p}^*$.

Giá trị **p**, α công khai, có dấu xác nhận của TT.

Mỗi người dùng U có chữ ký với thuật toán xác minh $\mathbf{ver_u}$.

TT có chữ ký với thuật toán xác minh $\mathbf{ver_{TT}}$.

Mỗi người dùng U có dấu xác nhận định danh ID(U) là:

$$\mathbf{C}(\mathbf{U}) = (ID(U), \mathbf{ver_u}, sig_{TT}(ID(U), \mathbf{ver_u}))$$

1) **U** chọn $\mathbf{a_u}$ ngẫu nhiên, bí mật ($0 \le \mathbf{a_u} \le p - 2$), tính $\alpha^{a_u} \bmod p$, gửi tới **V**.

2) **V** chọn $\mathbf{a_v}$ ngẫu nhiên, bí mật ($0 \le \mathbf{a_v} \le p - 2$)

Tính $\alpha^{a_v} \bmod p$, $\mathbf{y_v} = \mathbf{sig}_{\mathbf{v}}(\alpha^{a_v}, \alpha^{a_u})$. Gửi $(\mathbf{C}(V), \alpha^{a_v} \bmod p, \mathbf{y_v})$ tới **U**.

V tính khóa chung $\mathbf{K_{v,u}} = (\alpha^{a_u})^{a_v} \bmod p$

3) U tính khóa chung $\mathbf{K_{u,v}} = (\alpha^{a_v})^{a_u} \bmod p$

Dùng $\mathbf{ver_v}$ để xác minh $\mathbf{y_v}$ và xác minh $\mathbf{C}(V)$ nhờ $\mathbf{ver_{TT}}$.

Tính $\mathbf{y_u} = \mathbf{sig}_{\mathbf{u}}(\alpha^{a_u}, \alpha^{a_v})$ Gửi $(\mathbf{C}(U), \mathbf{y_u})$ tới V.

4) Dùng $\mathbf{ver_u}$ để xác minh $\mathbf{y_u}$ và xác minh $\mathbf{C}(U)$ nhờ $\mathbf{ver_{TT}}$.

Mức an toàn

STS là giao thức "***3 lần***" truyền tin. Thông tin trao đổi trong giao thức như sau:

α^{a_u}

---------->

α^{a_v}, $\mathbf{sig}_{\mathbf{v}}(\alpha^{a_v}, \alpha^{a_u})$

U <---------- **V**

$\mathbf{sig}_{\mathbf{u}}(\alpha^{a_u}, \alpha^{a_v})$

---------->

1/ Giao thức có thể bảo vệ trước tấn công của "kẻ xâm nhập giữa cuộc".

+ Giống như giao thức phân phối khoá Diffie-Hellman (PP DH), kẻ tấn công **W** chặn bắt α^{a_u} của U và thay nó bằng $\alpha^{a'_u}$. Tiếp đó **W** chặn bắt α^{a_v}, $\mathbf{sig}_{\mathbf{v}}(\alpha^{a_v}, \alpha^{a_u})$ từ V.

+ **W** cũng muốn thay α^{a_v} của V bằng $\alpha^{a'_v}$. Điều này có nghĩa là anh ta cũng phải thay $\mathbf{y_v} = \mathbf{sig}_{\mathbf{v}}(\alpha^{a_v}, \alpha^{a_u})$ bằng $\mathbf{sig}_{\mathbf{v}}(\alpha^{a'_v}, \alpha^{a_u})$

Đáng tiếc **W *khó thể*** tính **sig** $_{\mathbf{v}}(\alpha^{a'_v}, \alpha^{a_u})$, vì không biết thuật toán ký **sig** $_{\mathbf{v}}$ của V.

+ Tương tự, **W *khó thể*** thay thế **sig** $_{\mathbf{u}}(\alpha^{a_u}, \alpha^{a_v})$ bằng **sig** $_{\mathbf{u}}(\alpha^{a'_u}, \alpha^{a_v})$, do anh ta không biết thuật toán ký **sig** $_{\mathbf{u}}$ của U.

α^{a_u} $\alpha^{a'_u}$

U ----------> **W** ----------> **V**

α^{a_v}, **sig** $_{\mathbf{v}}(\alpha^{a_v}, \alpha^{a_u})$

<----------

W muốn thay bằng $\alpha^{a'_v}$ để gửi cho U, nhưng ***khó thể*** tính **sig** $_{\mathbf{v}}(\alpha^{a'_v}, \alpha^{a_u})$ = **?**

sig $_{\mathbf{u}}(\alpha^{a_u}, \alpha^{a_v})$

---------->

W muốn thay bằng $\alpha^{a'_u}$ để gửi cho V, nhưng ***khó thể*** tính **sig** $_{\mathbf{u}}(\alpha^{a'_u}, \alpha^{a_v})$ = **?**

2/ Giao thức **STS** không đưa ra được ***"sự khẳng định khoá công khai"***.

Tức là trong bước 2), α^{a_v} và $\mathbf{y_v}$ được gửi tới U, nhưng chưa bảo đảm thật an toàn.

Trong bước 1), 3), α^{a_u} và $\mathbf{y_u}$ được gửi tới V, nhưng chưa bảo đảm thật an toàn.

Có thể bảo đảm an toàn y_v và y_u bằng cách:

Trong bước 2): mã hoá y_v bằng khoá session K:

$z_v = \mathbf{e_K}(\text{sig}_v(\alpha^{a_v}, \alpha^{a_u})) = e_K(y_v)$

Trong bước 3): mã hoá y_u bằng khoá session K:

$z_u = \mathbf{e_K}(\text{sig}_u(\alpha^{a_u}, \alpha^{a_v})) = e_K(y_u)$

7.3.4. Giao thức thoả thuận khoá MTI.

Matsumoto, Takashima và Imai (MTI) đã xây dựng giao thức thoả thuận khoá bằng cách cải biên giao thức trao đổi khoá **STS**. Giao thức không đòi hỏi U và V phải tính bất kỳ chữ ký nào. Đó là giao thức "***2 lần***", vì chỉ có 2 lần truyền tin riêng biệt (một từ U đến V và một từ V đến U)

<u>Sơ đồ</u>

Chuẩn bị:

+ Chọn số nguyên tố **p** rất lớn sao cho bài toán logarit rời rạc trong $\mathbf{Z_p}^*$ là "khó" giải.

Chọn α là phần tử nguyên thuỷ $\in \mathbf{Z_p}^*$.

Giá trị **p**, α công khai, với dấu xác nhận của trung tâm tin cậy **TT**.

Mỗi người dùng U có số mũ bí mật $\mathbf{a_u}$ ($0 \le a_u \le p-2$) và giá trị công khai tương ứng:

$\mathbf{b_u} = \alpha^{a_u} \bmod p$.

TT có sơ đồ chữ ký với thuật toán ký bí mật sig_{TT} và thuật toán xác minh ver_{TT}. Mỗi người dùng U có định danh ID(U) và dấu xác nhận của **TT**:

$\mathbf{C}(U) = (ID(U), \mathbf{b_u}, \mathbf{sig}_{TT}(ID(U), \mathbf{b_u}))$

1) **U** chọn ngẫu nhiên bí mật $\mathbf{r_u}$, $0 \le \mathbf{r_u} \le p-2$, tính $\mathbf{s_u} = \alpha^{r_u} \bmod p$ và gửi $(\mathbf{C}(U), \mathbf{s_u})$ đến V.

2) **V** chọn ngẫu nhiên bí mật $\mathbf{r_v}$, $0 \le \mathbf{r_v} \le p-2$, tính $\mathbf{s_v} = \alpha^{r_v} \bmod p$ và gửi $(\mathbf{C}(V), \mathbf{s_v})$ đến U.

3) U tính khoá $\mathbf{K} = s_v^{a_u} * b_v^{r_u} \bmod p$, trong đó $\mathbf{b_v}$ nhận từ $\mathbf{C}(V)$

4) V tính khoá $\mathbf{K} = s_u^{a_v} * b_u^{r_v} \bmod p$, trong đó $\mathbf{b_u}$ nhận từ $\mathbf{C}(U)$

Cuối giao thức, U và V đều tính được cùng một khoá $\mathbf{K} = \alpha^{r_u * a_v + r_v * a_u} \bmod p$.

Mức an toàn

Thông tin được truyền trong giao thức "***2 lần***":

$\mathbf{C}(U), \mathbf{s_u} = \alpha^{r_u} \bmod p$

U ----------> **v**

$\mathbf{C}(V), \mathbf{s_v} = \alpha^{r_v} \bmod p$

<----------

1/ Trước tấn công thụ động, độ an toàn của giao thức MTI như bài toán Diffie-Hellman.

Trước tấn công chủ động, việc chứng minh tính an toàn không đơn giản.

2/ Nếu trong giao thức ***không dùng chữ ký***, có thể xuất hiện tình huống ***không có sự bảo vệ*** nào trước "***kẻ xâm nhập giữa cuộc***".

Ví dụ **W** có thể tráo đổi các giá trị mà U và V gửi cho nhau:

C(U), α^{r_u} C(U), $\alpha^{r'_u}$

U ----------> **W** ----------> **V**

C(V), $\alpha^{r'_v}$ C(V), α^{r_v}

<---------- <----------

Trong trường hợp này, U và V sẽ tính ra các khoá khác nhau !

U tính khoá $\mathbf{K_u} = \alpha^{r_u a_v + r'_v a_u} \bmod p$. V tính khoá $\mathbf{K_v} = \alpha^{r'_u a_v + r_v a_u} \bmod p$.

Tất nhiên, **W** không thể tính ra khoá đúng của U và V, vì phải biết mũ mật $\mathbf{a_u}$ và $\mathbf{a_v}$ tương ứng.

Nói cách khác, cả U và V đều được đảm bảo rằng, người dùng khác trên mạng chỉ có thể tính được khoá "***rởm***".

Tính chất này còn được gọi là ***xác thực khoá ẩn*** (implicit key authentication)

Ví dụ Giao thức thoả thuận khoá MTI

Chuẩn bị:

Chọn số nguyên tố **p** = 27 803, α = 5 là phần tử nguyên thuỷ $\in \mathbf{Z_p}^*$.

U chọn bí mật $\mathbf{a_u}$ = 21 131, tính $\mathbf{b_u} = 5^{21131} \bmod 27803 = 21\,420$.

Có dấu xác nhận **C**(U) = (ID(U), $\mathbf{b_u}$, **sig** $_{TT}$(ID(U), $\mathbf{b_u}$))

V chọn bí mật $\mathbf{a_v}$ = 17555, tính $\mathbf{b_v} = 5^{17555} \bmod 27803 = 17\,100$.

Có dấu xác nhận **C**(V) = (ID(V), $\mathbf{b_v}$, **sig** $_{TT}$(ID(V), $\mathbf{b_v}$))

1) U chọn $\mathbf{r_u}$ = 169, tính $\mathbf{s_u} = 5^{169} \bmod 27803 = 6268$, gửi $\mathbf{s_u}$ đến V.

2) V chọn $\mathbf{r_v}$ = 23456, tính $\mathbf{s_v} = 5^{23456} \bmod 27803 = 26759$, gửi $\mathbf{s_v}$ đến U.

3) U tính khoá $\mathbf{K_{u,v}} = s_v^{a_u} * b_v^{r_u} \bmod p$

$= 26759^{21131} * 17100^{169} \bmod 27803 =$ **21 600**.

4) V tính khoá $\mathbf{K_{v,u}} = s_u^{a_v} * b_u^{r_v} \bmod p$

$= 6268^{17555} * 21420^{23456} \bmod 27803 =$ **21 600**.

Như vậy U và V đã tính cùng một khoá chung.

7.3. Giao thức Chia sẻ bí mật

7.3.1. Tổng quan về "Chia sẻ bí mật"

7.3.1.1. Khái niệm "Chia sẻ bí mật".

Thông tin quan trọng (Tin gốc: **TG**) cần giữ bí mật, không nên trao cho một người nắm giữ, mà phải "***chia***" Thông tin đó thành nhiều mảnh (**TM**), và trao cho mỗi người giữ một hay một số mảnh. Thông tin gốc chỉ có thể được xem lại, khi mọi người giữ các mảnh tin đều nhất trí. Các mảnh **TM** được "***khớp lại***" để được Tin gốc **TG**.

Ví dụ ngân hàng có một két bạc phải mở hàng ngày. Ngân hàng có 3 thủ quỹ kinh nghiệm, nhưng họ không tin người nào. Bởi vậy họ cần xây dựng một hệ thống sao cho bất kỳ 2 thủ quỹ nào cũng có thể mở được két bạc, nhưng riêng từng người một thì khó thể mở được. Vấn đề này có thể được giải quyết bằng phương pháp "***Chia sẻ bí mật***". Đó là cơ chế "Chia sẻ bí mật" "2 trong 3".

Phương pháp "***Chia sẻ bí mật***" được thể hiện bằng "***sơ đồ***" chỉ rõ cách thức "***Chia sẻ bí mật***" và cách thức "***khớp lại***" các mảnh bí mật.

Sơ đồ "***Chia sẻ bí mật***" dùng để chia sẻ "***tin mật***" cho **n** thành viên, sao cho chỉ những tập con "***hợp thức***" các thành viên mới có thể khôi phục lại "***tin mật***", còn lại, bất kỳ tập con "***không hợp thức***" các thành viên thì ***khó thể*** làm được điều đó.

7.3.1.2. Ví dụ về "Chia sẻ bí mật".

Các sơ đồ chia sẻ bí mật có ứng dụng rất rộng rãi trong thực tế, như là chia sẻ ***khoá*** để mở két bạc, chia sẻ ***khoá*** để ký số văn bản. Các sơ đồ chia sẻ bí mật đầu tiên được đưa ra độc lập bởi Shamir và Blakley.

Ví dụ 1:

Ví dụ, "***chìa khóa***" mở két bạc là "chìa khóa số" (chìa khóa gốc), được chia thành 3 mảnh khóa. Mỗi thủ quĩ chỉ được giữ 1 mảnh khóa, mảnh khóa này không thể mở được két bạc. Khi 2 trong 3 người nhất trí mở két bạc, họ khớp 2 mảnh khóa của họ với nhau, sẽ nhận được chìa khóa gốc để mở két bạc.

Ví dụ 2:

Một ví dụ thực tế khác về nhu cầu "chia sẻ bí mật" là việc điều khiển vũ khí hạt nhân tại một nước, họ chỉ có một "***chìa khóa***" ấn nút dùng vũ khí. Với tổng thống, thủ tướng, bộ trưởng bộ quốc phòng, họ qui định 2 trong 3 thành viên trên cùng nhất trí thì mới được dùng vũ khí. Và họ cũng sử dụng cơ chế "Chia sẻ bí mật" "2 trong 3".

Ví dụ 3:

Ví dụ cần sử dụng phương pháp "chia sẻ bí mật" là chia nhỏ "***lá phiếu"*** trong bỏ phiếu điện tử. Trên thực tế nhiều khi người ta chưa thật tin vào một nhóm ít người trong Ban kiểm phiếu (BKP), người bỏ phiếu được phép chia "***lá phiếu***" thành nhiều "***mảnh phiếu***", sau đó gửi cho mỗi người trong BKP một "mảnh". Như vậy từng thành viên trong BKP khó thể đọc được nội dung "***lá phiếu***".

Khi mọi thành viên trong BKP nhất trí xem nội dung lá phiếu, thì các "***mảnh phiếu***" mới được khớp lại để có được "***lá phiếu***" ban đầu của người bỏ phiếu.

Chia nhỏ "***lá phiếu***" có thể hiểu theo nhiều nghĩa: Có thể là chia ***khóa bí mật*** để giải mã nội dung lá phiếu; Có thể chính ***nội dung lá phiếu*** (tên, mã số ứng cử viên, ý kiến đồng ý hay không đồng ý, …) được chia thành các "mảnh tin" (trên thực tế bản thân mỗi mảnh tin đều không có nghĩa)

Ví dụ 4:

Sơ đồ "***chia sẻ bí mật***" là một trường hợp đặc biệt của sơ đồ "***phân phối khoá***". Khi một cá nhân đáng tin cậy hoặc một trung tâm được uỷ quyền phân phối khoá cho một tập các thành viên, họ sẽ phải đáp ứng yêu cầu: có "***một khoá bí mật***", làm sao để mỗi người giữ "***một bộ phận của khoá***", khi những tập con thành viên được chỉ định trước thì có thể tính ra "***khoá***", những tập con thành viên khác thì ***khó thể*** làm đươc việc đó. Một sơ đồ "phân phối khoá" thoả mãn yêu cầu trên cũng gọi là một sơ đồ "chia sẻ bí mật".

7.3.2. Giao thức "Chia sẻ bí mật" Sharmir

7.3.2.1. Khái niệm Sơ đồ ngưỡng A(t, m)

Cho **t**, **m** là các số nguyên dương, **t** ≤ **m**. ***Sơ đồ ngưỡng*** **A(t, m)** là phương pháp phân chia khoá **K** cho một tập **P** gồm **m** thành viên $\mathbf{P_i}$, sao cho **t** thành viên bất kỳ có thể tính được giá trị **K**, nhưng không một nhóm gồm (**t-1**) thành viên nào có thể làm được điều đó.

Ví dụ có **m** =3 thủ quĩ giữ két bạc. Hãy xây dựng sơ đồ "***chia sẻ bí mật***", sao cho **t** = 2 thủ quĩ nào cũng có thể mở được két bạc, nhưng từng người một riêng rẽ thì khó thể. Đó là sơ đồ ngưỡng **A**(2, 3)

Cho đơn giản ta xem người phân phối khóa D không thuộc nhóm P. Khi D muốn phân chia ***khoá*** **K** cho các thành viên trong **P** , anh ta sẽ cho mỗi thành viên một thông tin "***cục bộ***" nào đó về **K**, được gọi là các "***mảnh khóa***". Các mảnh khóa được phân phát

một cách bí mật, để không một thành viên nào biết được mảnh khóa của thành viên khác. Sơ đồ ngưỡng của Sharmir được đưa ra năm 1979.

***Sơ đồ ngưỡng* A(t, t) *trong* Z_m (*Trường hợp đặc biệt khi* t = m):**

**** Chia sẻ khóa bí mật* K.**

1) D chọn một cách bí mật (độc lập và ngẫu nhiên) (**t -1**) phần tử của Z_m:

$y_1, \ldots, y_{t-1}$

2) D tính $y_i = K - \sum_{j=1}^{t} y_j \pmod m$ với $1 \le i \le t$, trao mảnh y_i cho P_i

**** Khôi phục khóa bí mật* K.**

t thành viên cùng hợp tác, có thể tính **K** theo công thức:

$$K = y_i - \sum_{j=1}^{t} y_j \pmod m$$

***7.3.2.2. Chia sẻ khóa bí mật* K.**

1) D chọn số nguyên tố **p**, và m phần tử x_i ($1 \le i \le m$) khác nhau - khác **0** trong Z_p

x_i là công khai. D trao giá trị x_i cho mỗi thành viên P_i .

2) Để chia mảnh khoá $K \in Z_p$, D chọn bí mật, ngẫu nhiên **t-1** phần tử của Z_p là $a_1, \ldots, a_{t-1}$, lập đa thức trong Z_p là $P(x) = K + \sum_{j=1}^{t-1} a_j x^j \bmod p$

Với $1 \le i \le m$, D tính $y_i = P(x_i)$, và trao mảnh y_i cho P_i .

Chú ý đa thức **P**(x) có bậc tối đa là **t-1**, thành phần hằng số là khoá bí mật **K**. Mỗi thành viên P_i sẽ có một điểm (x_i, y_i) trên đa thức.

***Ví dụ 1: Chia sẻ khóa bí mật* K = 13**

Khóa **K=13** cần chia thành 3 mảnh cho 3 người P1 , P3 , P5.

1) Chọn số nguyên tố **p** = **17**, chọn m = 5 phần tử $x_i = i$. trong Z_p, i =1, 2, 3, 4, 5.

D trao giá trị công khai x_i cho P_i .

2) D chọn bí mật **t - 1** = 2 phần tử trong Z_p : $a_1 = 10$, $a_2 = 2$. Lập đa thức $P(x) = K + \sum_{j=1}^{t-1} a_j x_j^j \pmod p = 13 + a_1 x + a_2 x^2 \pmod{17}$

D tính $y_i = P(x_i)$, $1 \le i \le m$, trao mảnh y_i cho P_i :

$y_1 = P(x1) = P(1) = 13 + a_1 .1 + a_2 .1^2 = 13 + 10 .1 + 2 .1^2 = 8$

$\mathbf{y_3} = \mathbf{P}(x3) = P(3) = \mathbf{13} + a_1 .3 + a_2 .3^2 = 13 + 10 .3 + 2 .3^2 = \mathbf{10}$

$\mathbf{y_5} = \mathbf{P}(x5) = P(5) = \mathbf{13} + a_1 .5 + a_2 .5^2 = 13 + 10 .5 + 2 .5^2 = \mathbf{11}$

***7.3.2.3. Khôi phục khóa bí mật* K.**

Một tập con gồm **t** thành viên có thể khôi phục lại khoá như thế nào. Có hai phương pháp: Giải hệ phương trình tuyến tính và dùng công thức nội suy Lagrangre.

***1) Phương pháp 1:* Giải hệ phương trình tuyến tính khôi phục K.**

Giải hệ phương trình tuyến tính **t** ẩn số, **t** phương trình.

Giả sử rằng các thành viên $P_{i1},\ldots, P_{it}$ muốn xác định khoá **K**. Họ biết rằng

$\mathbf{y_{ij}} = \mathbf{P}(\mathbf{x_{ij}})$, $1 \le j \le \mathbf{t}$, trong đó $P(x) \in Z_p[x]$ là đa thức do phân phối khóa D chọn.

Vì **P**(x) có bậc lớn nhất là (**t-1**) nên có thể viết $\mathbf{P}(x) = \mathbf{a_0} + \mathbf{a_1}\,\mathbf{x} + \ldots + \mathbf{a_{t-1}}\,\mathbf{x}^{\mathbf{t-1}}$

trong đó các hệ số $\mathbf{a_0}, \mathbf{a_1}, \ldots \mathbf{a_{t-1}}$ là chưa biết của $\mathbf{Z_p}$, còn $\mathbf{a_0} = \mathbf{K}$ là khoá.

Vì $\mathbf{y_{ij}} = \mathbf{P}(\mathbf{x_{ij}})$, nên ta có hệ phương trình tuyến tính **t** ẩn số, **t** phương trình.

Chú ý ở đây các phép tính số học đều thực hiện trên $\mathbf{Z_p}$. Nếu các phương trình độc lập tuyến tính, thì sẽ có một nghiệm duy nhất, trong đó giá trị khoá $\mathbf{a_0} = \mathbf{K}$.

***Ví dụ 2: Khôi phục khóa bí mật* K = 13 (*Bằng phương pháp 1*)**

Trong ví dụ 1, ta đã biết D chọn số nguyên tố **p** = **17**, chọn m = 5 phần tử

$\mathbf{x_i = i}$. trong $\mathbf{Z_p}$, i =1, 2, 3, 4, 5. D trao giá trị công khai $\mathbf{x_i}$ cho $\mathbf{P_i}$

Giả sử nhóm thành viên giữ các "mảnh khóa" **B** = {P_1, P_3, P_5} sẽ kết hợp các mảnh của họ tương ứng là $\mathbf{y_1} = \mathbf{8}$, $\mathbf{y_3} = \mathbf{10}$, $\mathbf{y_5} = \mathbf{11}$.

B biết đa thức $\mathbf{P}(x) = \mathbf{K} + \sum_{j=1}^{t-1} \mathbf{a_j}\,\mathbf{x_j}^{\mathbf{j}} \pmod{\mathbf{p}} = \mathbf{a_0} + \mathbf{a_1}\,\mathbf{x} + \mathbf{a_2}\,\mathbf{x^2} \pmod{\mathbf{17}}$

Các phương trình cụ thể trong $\mathbf{Z_{17}}$ là

$a_0 + a_1 + a_2 = \mathbf{8} \pmod{\mathbf{17}}$

$a_0 + 3\,a_1 + 9\,a_2 = \mathbf{10} \pmod{\mathbf{17}}$

$a_0 + 5\,a_1 + 8\,a_2 = \mathbf{11} \pmod{\mathbf{17}}$

Hệ này có nghiệm duy nhất trong Z_{17} là $a_0 = 13$, $a_1 = 10$, $a_2 = 2$.

Như vậy khoá được khôi phục lại là : $\mathbf{K} = \mathbf{a_0} = \mathbf{13}$

***2) Phương pháp 2:* Dùng công thức nội suy Lagrangre khôi phục K.**

$\mathbf{P}(x) = \sum_{\mathbf{j}=1}^{\mathbf{t}} \mathbf{y_{ij}} \prod_{1 \le k \le \mathbf{t},\ k \ne j} (\mathbf{x} - \mathbf{x_{ik}}) / (\mathbf{x_{ij}} - \mathbf{x_{ik}}) \pmod{\mathbf{p}}$

Có các tính chất như đa thức **P**(x) ở trên: $\mathbf{y_{ij}} = \mathbf{P}(\mathbf{x_{ij}})$, $1 \le j \le t$.

$\mathbf{K} = \mathbf{P}(0) = \sum \mathbf{t}_{\mathbf{j}=1} \mathbf{y}_{i\,j} \prod_{1 \le k \le \mathbf{t},\, k \ne j} (0 - \mathbf{x}_{i\,k}) / (\mathbf{x}_{i\,j} - \mathbf{x}_{i\,k}) \pmod{\mathbf{p}}$

Nếu ta đặt $b_j = \prod_{1 \le k \le \mathbf{t},\ k \ne j} (- \mathbf{x}_{i\,k}) / (\mathbf{x}_{i\,j} - \mathbf{x}_{i\,k}) \pmod{\mathbf{p}}$

Khi đó $K = \sum \mathbf{t}_{\mathbf{j}=1} \mathbf{y}_{i\,j}\ b_j$

Ví dụ 3: Khôi phục khóa bí mật* K = 13 (Bằng phương pháp 2***)**

B ={P1, P3, P5} cần kết hợp các mảnh khóa của họ y1 =8, y3 = 10, y5 = 11, để khôi phục lại khóa K.

B ={P1, P3, P5} cần kết hợp các mảnh khóa của họ y1 =8, y3 = 10, y5 = 11,
để khôi phục lại khóa K. Tính $\mathbf{b_j} = \prod_{1 \le k \le \mathbf{t},\ k \ne \mathbf{j}} (- \mathbf{x}_{i\,k}) / (\mathbf{x}_{i\,\mathbf{j}} - \mathbf{x}_{i\,k}) \pmod{\mathbf{p}}$.

$$\mathbf{b1} = \frac{-x3}{(x1 - x3)} * \frac{-x5}{(x1 - x5)} = \frac{-3}{1-3} * \frac{-5}{1-5} = \frac{3*5}{2*4} =$$

$= 3 . 5 * (2)^{-1} . (4)^{-1} \pmod{17} = 3 . 5 * 9 . 13 \pmod{17} = \mathbf{4}$

$$\mathbf{b3} = \frac{-x1}{(x3 - x1)} * \frac{-x5}{(x3 - x5)} = \frac{-1}{3-1} * \frac{-5}{3-5} = \frac{1*5}{2*(-2)} =$$

$= 1. 5 * (2)^{-1} . (-2)^{-1} \pmod{17} = 1 . 5 * 9 . (-9) \pmod{17} = \mathbf{3}$

$$\mathbf{b5} = \frac{-x1}{(x5 - x1)} * \frac{-x3}{(x5 - x3)} = \frac{-1}{5-1} * \frac{-3}{5-3} = \frac{1*3}{4*2} =$$

$= 1 . 3 * (4)^{-1} . (2)^{-1} \pmod{17} = 1 . 3 * 13 . 9 \pmod{17} = \mathbf{11}$

Theo công thức trên ta có: $\mathbf{K} = \mathbf{8} * \mathbf{4} + \mathbf{10} * \mathbf{3} + \mathbf{11} * \mathbf{11} \pmod{17} = \mathbf{13}$

7.3.3. Giao thức "Chia sẻ bí mật" bằng "Mạch đơn điệu".

7.3.3.1. Các khái niệm

Vấn đề:

Giả sử ta không muốn tất cả các tập con **t** thành viên bất kỳ đều ***có khả năng tính được khoá*** **K** như trong sơ đồ ngưỡng Shamir, mà ***chỉ một số tập con*** thành viên chỉ định trước có thể làm được điều đó.

Cấu trúc mạch đơn điệu (Benaloh, Leichter) là một giải pháp cho yêu cầu trên.

***Ký hiệu*:**

P là Tập gồm **m** thành viên được chia mảnh công khai $\mathbf{x_i}$.

Tập con các thành viên có thể tính được khoá **K** gọi là "***tập con hợp thức***".

Γ là một họ các "***tập con hợp thức***". **Γ** được gọi là một "***cấu trúc truy nhập***".

(**Γ** là họ các tập con của **P**, mà mỗi tập con các thành viên này có khả năng tính **K**)

Ví dụ:

Nếu trong tập các thành viên $\mathbf{\{P_1, P_2, P_3, P_4\}}$ chỉ có các tập con sau có thể mở khoá:

$\mathbf{\{P_1, P_2, P_4\}, \{P_1, P_3, P_4\}, \{P_2, P_3\}}$, thì đó là các "***tập con hợp thức***".

Ở đây **Γ** là $\mathbf{\{ \{P_1, P_2, P_4\}, \{P_1, P_3, P_4\}, \{P_2, P_3\} \}}$

Định nghĩa

Một sơ đồ chia sẻ bí mật được gọi là "***hoàn thiện***" nếu thỏa mãn điều kiện:

1) Nếu một "***tập con hợp thức***" các thành viên $\mathbf{B \subseteq P}$ góp chung các "***mảnh khóa***" của họ, thì có thể xác định được khoá **K**.

2) Nếu một "***tập con không hợp thức***" các thành viên $\mathbf{B \subseteq P}$ góp chung các "***mảnh khóa***" của họ, thì cũng ***khó thể*** xác định được khoá **K**.

Ví dụ

Sơ đồ Shamir **A(t, m)** thể hiện "***cấu trúc truy nhập***" sau:

$\mathbf{\Gamma = \{B \subseteq P : /B/ \geq t\}}$

Như vậy Sơ đồ Shamir là sơ đồ chia sẻ bí mật "***hoàn thiện***".

Chú ý:

"***Tập trên***" (Superset) của "***tập hợp thức***" sẽ lại là "***tập hợp thức***".

TL: Nếu $\mathbf{B \in \Gamma}$, và $\mathbf{B \subseteq C \subseteq P}$, thì $\mathbf{C \in \Gamma}$.

Điều trên nói rằng "***cấu trúc truy nhập***" **Γ** phải thoả mãn tính chất ***đơn điệu***.

Định nghĩa

$\mathbf{B} \in \Gamma$ được gọi là "***tập hợp thức***" **tối thiểu** nếu $\mathbf{C} \subset \mathbf{B}$, $\mathbf{C} \neq \mathbf{B}$ thì $\mathbf{C} \notin \Gamma$.

Nói cách khác **B** là "***tập hợp thức***" ***nhỏ nhất*** trong Γ.

Tập mọi tập con "tập hợp thức" tối thiểu của Γ đ.g.l ***Tập cơ sở***, ký hiệu là Γ_0.

Như vậy có thể biểu diễn $\Gamma = \{\mathbf{B} \subseteq \mathbf{P} / \ \mathbf{C} \subset \mathbf{B}, \ \mathbf{C} \in \Gamma_0 \}$.

Ví dụ:

Nếu trong tập các thành viên $\{P_1, P_2, P_3, P_4\}$ có các "***tập hợp thức***" là:

$\{P_1, P_2, P_4\}, \{P_1, P_3, P_4\}, \{P_2, P_3\},$ thì đó là ***Tập cơ sở*** Γ_0.

B_1 B_2 B_3

Như vậy $\Gamma = \Gamma_0 \cup \{ \{P_1, P_2, P_3, P_4\}, \{P_1, P_2, P_3\}, \{P_2, P_3, P_4\} \}$.

C_1 C_2 C_3

Ta thấy $B_1, B_2 \subset C_1$ và $B_3 \subset C_2$, $B_3 \subset C_3$

Ví dụ:

Trong sơ đồ **A(t, m),** ***Tập cơ sở*** gồm tất cả các tập con chứa đúng **t** thành viên.

Định nghĩa **Mạch đơn điệu**

Cho **C** là ***mạch*** Boolean (Công thức logic) với **m** đầu vào (biến logic) $X_1, X_2, \ldots, X_m$ (tương ứng với **m** thành viên $P_1, P_2, \ldots, P_m$) và **1** đầu ra (biến logic) **Y**.

Mạch này chỉ gồm các cổng "hoặc" và các cổng "và".

Mạch **C** như trên được gọi là ***Mạch đơn điệu***.

7.3.3.2. Xây dựng Mạch đơn điệu

Cho Γ là tập đơn điệu các tập con của **P**, trong đó Γ_0 là tập cơ sở của Γ.

Khi đó ta xây dựng công thức Boolean dạng tuyển hội chuẩn sau:

$$\mathbf{C} = \bigvee_{\mathbf{B} \in \Gamma_0} \left(\bigwedge_{P_i \in \mathbf{B}} P_i \right) \quad (\mathbf{B} \text{ là tập hợp thức tối thiểu})$$

Đó là **Mạch đơn điệu**.

Ví dụ:

Nếu trong tập các thành viên $\{P_1, P_2, P_3, P_4\}$ có tập cơ sở

$\Gamma_0 = \{ \{P_1, P_2, P_4\}, \{P_1, P_3, P_4\}, \{P_2, P_3\} \}$.

Theo cách xây dựng trên, công thức Boolean sau là **Mạch đơn điệu**:

$\mathbf{C = (P_1 \wedge P_2 \wedge P_4) \vee (P_1 \wedge P_3 \wedge P_4) \vee (P_2 \wedge P_3)}$

Mỗi mệnh đề trong công thức với một cổng "và" của mạch đơn điệu tương ứng, phép tuyển cuối cùng ứng với cổng "hoặc".

7.3.3.3. " Chia sẻ" khóa bí mật K dựa vào "mạch đơn điệu"

Thuật toán thực hiện phép gán một giá trị **f**(**W**) ∈ **K** cho mỗi ***dây*** **W** trong mạch **C**.

+ Đầu tiên, ***dây ra*** $\mathbf{W_{out}}$ của mạch sẽ được gán giá trị khoá **K**.

+ Thuật toán sẽ lặp lại một số lần cho đến khi ***mỗi dây*** có một giá trị gán vào nó.

+ Cuối cùng, mỗi thành viên $\mathbf{P_i}$ sẽ được một danh sách các giá trị **f**(**W**) sao cho **W** là một dây vào của mạch có đầu vào $\mathbf{x_i}$.

Thuật toán "chia sẻ" ***khoá K***

1) $\mathbf{f(W_{out}) = K}$;

2) **WHILE** tồn tại một ***dây*** **W** sao cho **f**(**W**) không được xác định **DO**

Begin

3) Tìm cổng **G** của **C** sao cho $\mathbf{f(W_G)}$ được xác định, $\mathbf{W_G}$ là dây ra của **G** nhưng **f**(**W**) không được xác định với bất kỳ dây nào của **G**.

4) **If G** là cổng "**hoặc**" **Then f**(**W**) := $\mathbf{f(W_G)}$ với mỗi dây vào **W** của **G**

Else (tl: **G** là cổng "**và**")

Cho các dây vào của **G** là $\mathbf{W_1}$, …, $\mathbf{W_t}$

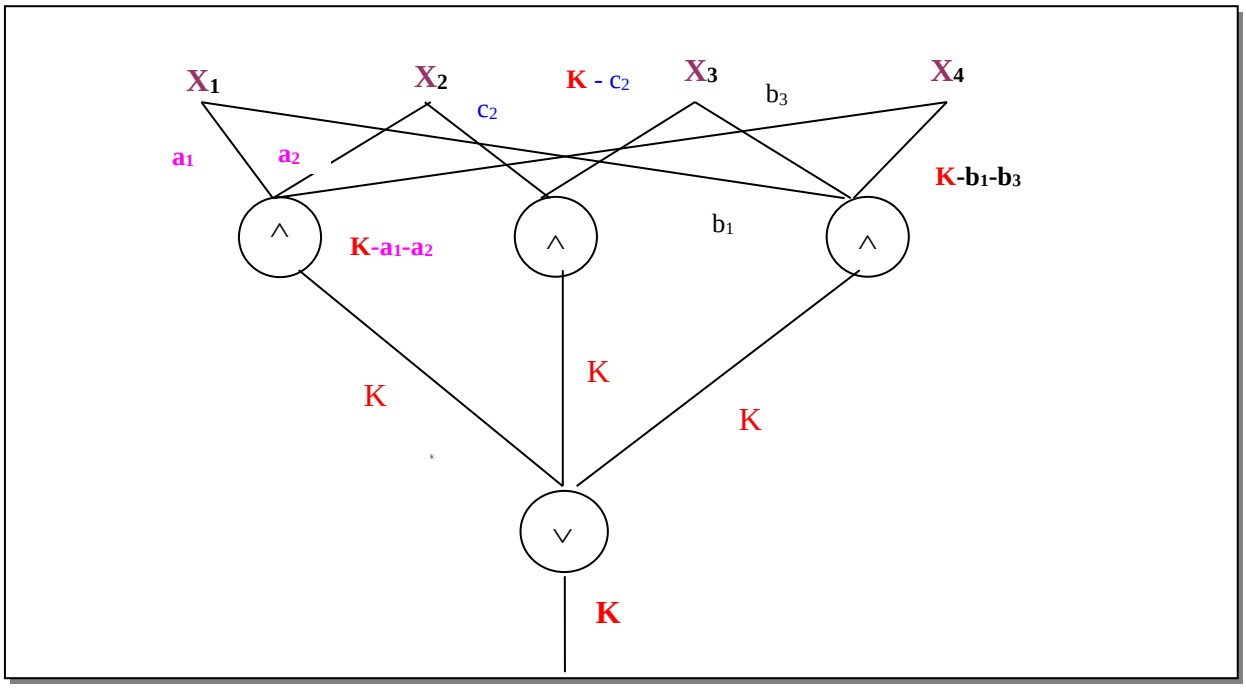
Chọn độc lập, ngẫu nhiên **t-1** phần tử của $\mathbf{Z_m}$ và ký hiệu chúng là:

$\mathbf{y_{G,1}}$, …, $\mathbf{y_{G,t-1}}$.

Tính $\mathbf{y_{G,t}} = \mathbf{f(W_G)} - \sum_{i=1}^{t-1} \mathbf{y_{G,i}} \pmod{\mathbf{m}}$;

End;

5) **For** $1 \le i \le m$ **Do** $\mathbf{f(W_i) := y_{G,i}}$



Ví dụ:

Căn cứ vào mạch đơn điệu trên ta có:

P_1 nhận **a_1, b_1** (Ứng với **X_1**)

P_2 nhận **a_2, c_2** (Ứng với **X_2**)

P_3 nhận **b_3, K - c_2** (Ứng với **X_3**)

P_4 nhận **K - a_1 - a_2**, **K - b_1 - b_3** (Ứng với **X_4**)

Như vậy, mỗi thành viên nhận 2 phần tử trong **Z_p** làm "***mảnh khóa***" của mình.

Nhận xét: Sơ đồ trong ví dụ trên là "***hoàn thiện***".

+ Ta thấy mỗi tập con hợp thức có thể tính được **K**:

Tập con hợp thức $\{P_1, P_2, P_4\}$ có thể tính $K = a_1 + a_2 + (K - a_1 - a_2) \pmod m$

$\{P_1, P_3, P_4\}$ có thể tính $K = b_1 + b_3 + (K - b_1 - b_3) \pmod m$

$\{P_2, P_3\}$ có thể tính $K = c_2 + (K - c_2) \pmod m$

+ Trong khi tập con không hợp thức tối đa không tính được **K**:

Đó là các tập $\{P_1, P_2\}$, $\{P_1, P_3\}$, $\{P_1, P_4\}$, $\{P_2, P_4\}$, $\{P_3, P_4\}$

BÀI TẬP CHƯƠNG 7. QUẢN LÝ KHÓA

Viết các chương trình Phân phối khoá, Thỏa thuận khóa, Chia sẻ khóa sau:

1) Giao thức phân phối khoá Blom với k = 1.

2) Giao thức phân phối khoá Diffie-Hellman.

3) Giao thức phân phối khoá Kerboros.

4) Giao thức thỏa thuận khoá Diffie-Hellman.

5) Giao thức thỏa thuận khoá Trạm tới Trạm.

6) Giao thức thỏa thuận khoá MTI.

7) Giao thức "Chia sẻ bí mật" Sharmir.

8) Giao thức "Chia sẻ bí mật" bằng "Mạch đơn điệu".

TÀI LIỆU THAM KHẢO

Tiếng Việt

[i] Lương Chi Mai, Huỳnh Thị Thanh Bình (2000), *"Nhập Môn Đồ Hoạ Máy Tính"*, nhà xuất bản Khoa học và Kỹ thuật, tr 65-75.

[ii] Lương Mạnh Bá, Nguyễn Thanh Thuỷ (1999), *" Nhập Môn Xử Lý Ảnh Số"*, nhà xuất bản Khoa học và Kỹ thuật, tr 212-213.

[iii] Gs. Phan Đình Diệu (1999), *"Lý thuyết mật mã và an toàn thông tin",* Đại Học Quốc Gia Hà Nội - Khoa Công Nghệ, pp 44.

Tiếng Anh

[iv] R. B. Wolfgang and E. J. Delp (January 1999), *"Fragile watermarking using the VW2D watermark*," *Proceedings of the SPIE/IS&T Conference on Security and Watermarking of Multimedia Contents*, SPIE Vol. 3657, San Jose, CA, pp 124 .

[v] Onur Mutlu (December 2001) *"An Overview ofImage Watermarking Algorithms"* , Project Report EE 731R Digital Image Processing, pp 1.
www.watermarkingworld.org/WMMLArchive

[vi] Yu-Yuan chen, Hsiang-Kuang Pan, Yu Chee Tseng (1999), *"A secure Data hiding scheme for two – color images"*. Department of Computer Science and Information Engineering, National Central University, Chung-li, Taiwan, pp 1-5